Carl-Fredrik Sørensen

Adaptive Mobile Work Processes in Context-Rich, Heterogeneous Environments

Doctoral thesis
for the degree of doktor ingeniør

Trondheim, 2005

Norwegian University of Science and Technology
Faculty of Information Technology,
Mathematics and Electrical Engineering
Department of Computer and Information Science

NTNU
Norwegian University of Science and Technology
Doctoral thesis
for the degree of doktor ingeniør
Faculty of Information Technology,
Mathematics and Electrical Engineering
Department of Computer and Information Science

ISBN 82-471-7360-3 (printed ver.)
ISBN 82-471-7358-1 (electronic ver.)
ISSN 1503-8181

Doctoral theses at NTNU, 2005:229

Printed by NTNU-trykk

TO MY WIFE KRISTIN AND MY CHILDREN,
ELIAS, JAKOB, AND ANNA KAROLINE.

Abstract

The rapid evolution in ubiquitous technologies has increased the demand and enabled for computational support of work *anywhere, anytime*. Mobility implies migrating computer systems from relatively static physical environments to highly dynamic and heterogeneous environments. The physical environment (including humans) is increasingly being instrumented with sensors, actuators, and other computing devices. These capture, contain, or represent different types of information and knowledge that can be used to monitor and infer different properties of the environment, both static and dynamic. The properties can be represented as explicit context information that can help support, monitor, manage, and coordinate work processes performed within that environment.

This work concentrates on how to support work processes in mobile and dynamic environments. Dynamic context information has normally not been an integrated part of process support software. Such software has traditionally been built for static, well-defined work processes and environments with a priori known resources, states, and working conditions. We introduce the notion of **"smart" work processes**, enabling adaptation of work processes to dynamic working environments.

The contributions of this thesis can be divided into three main themes:

T1 Exploration and investigation of mobile workers.

T2 Investigation and development of concepts and frameworks to capture and describe properties of mobile work processes.

T3 Development of enabling technology to support "smart" work processes in pervasive and mobile, ad hoc computing environments.

The main contributions are:

C1 A characterisation framework for mobile work to understand and find requirements for mobile work processes.

C2 An architecture to support context-aware, mobile (smart) work processes.

C3 Enabling technologies to support mobile work.

- **C3.1 A context-aware middleware for mobile ad hoc environments to support context-aware, mobile (smart) work processes**.
- **C3.2 The NIDAROS framework for location-aware support on mobile devices**.

The contributions are presented as a paper collection.

Preface

This dissertation is submitted to the Norwegian University of Science and Technology (NTNU) in partial fulfilment of the requirements for the degree Doktor Ingeniør.

The work contained herein has been performed at the Department of Computer and Information Science, NTNU, Trondheim, under supervision of Professor Reidar Conradi.

The thesis is part of the MOWAHS project (MObile Work Across Heterogeneous Systems) and has been supported by grant 140439/431 from the Norwegian Research Council through the IKT'2010 Programme.

Acknowledgements

Working on a PhD is not possible without cooperation with other people. I have been in a favourable working environment with many talented and experienced colleagues. I will first of all thank my supervisor, Professor Reidar Conradi, for giving valuable feedback and advice during my work. Reidar has given me good opportunities to explore different research fields, not all leading to a PhD, but nevertheless been important for my professional development. Alf Inge Wang has also been invaluable for my work and life as a PhD student; a good colleague as well as friend. We have collaborated on a number of papers; many of these are included in this thesis. I would also like to thank the MOWAHS project: Mads Nygård, Heri Ramampiaro, and Hien Nam Le for their cooperation and involvement.

I have appreciated the discussions and feedback from Monica Divitini, Letizia Jaccheri, and Tor Stålhane. They have all helped me with insights and advice to my research. I would like to thank all current and former colleagues in the software engineering group, the administration, and other people at my department, who all have been important to provide a good working environment. Magne Syrstad has been especially helpful by proof-reading my thesis.

Several master students have participated to make this work possible by working on prototypes and by conducting parts of the user studies. I would like to thank all the students that have co-authored the different papers and participated in MOWAHS student projects.

I would thank Lancaster University, England, for a friendly welcome and the cooperation during my stay there. I would especially express my gratitude to Gordon Blair and the CORTEX project: Adrian Friday, Hector Duran-Limon, Paul Okanda, Thirunavukkarasu Sivaharan, and Maomao Wu. I would also like to thank Gerd Kortuem and Ian Som-

merville for interesting discussions and feedback to my research.

Lastly, I would thank my dear family, Kristin, Elias, Jakob, and Anna Karoline for their love and support; my parents, Terje and Anna-Grete, for always believing in me; my siblings with families, Ann-Torill, Asgeir, Gudmund, and Marina; and, finally, my parents-in-law, Reidun and Karsten for helping to cope with the daily life and for sharing their house at Hølonda.

Contents

List of Tables

List of Figures

Part I

Context

CHAPTER 1

Introduction

This chapter outlines the context and motivation for the research presented in this thesis. Research questions and claimed contributions are briefly presented together with a list of the papers. Finally, a thesis outline is presented.

1.1 Motivation

The rapid evolution in wireless and mobile technologies, and the expected evolution in ubiquitous computing including sensor and actuator technologies, open new possibilities to provide computational support for both professional and private activities, ”anytime” and ”anywhere”. Despite of this evolution, people tend to travel more and more, both locally and globally, to perform work related activities. Some of the reasons for this increased travelling are the emergence of information-based service work as the dominating profession in the post-industrial society, the increased importance of cooperation both within and across organisations, as well as the extensive adoption of mobile phones [WL01b]. Mobile phones enable mobility because people can be mobile and still accessible. They will probably in the near future replace the fixed phone because of a cheaper communication infrastructure and thus cheaper use. New ways of working have emerged as a result of the increased accessibility independent of location. To distinguish mobile workers as *moving* from terms as distributed and co-located, new terms have emerged like, e.g., ”road warrior” or ”nomad”. Kleinrock [Kle96] states that ”*Most of us are ”nomads” when it comes to computing and communications. As nomads, we own computers*

and communication devices that we carry about with us in our travels. Moreover, even without carrying portable computers or communications, there are many of us who travel to numerous locations in our business and personal lives, and who require access to computers and communications when we arrive at our destinations."

The concept of mobile or nomadic work is not new, but the development in mobile and wireless technology has changed how activities can be performed and supported by giving opportunities to automate and support work processes in real-time irrespective of time and location of the worker.

The environment we live in is more and more instrumented with embedded interactive systems that provide basic services like sensing of physical conditions like temperature, air pressure, and humidity. Also more advanced services like automatic recognition of people, and self-aware, smart physical artefacts [SGKK04] have been proposed to enhance and optimise the services provided by and in the work environment. The technological evolution means that more and more of software and computing devices become a normal and integral part of our lives by fading into the background. Weiser introduced the notion of **ubiquitous computing** [Wei02] in 1991:

> "*The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.*"

Software is vital to all parts of the new ubiquitous computing environment, but the nature and type of software differs from hardware platform to hardware platform. This makes integration of different systems a challenging task. The communication networks are also heterogeneous, offering different connection protocols, quality of service, and cost. The ubiquitous communication backbone is a mixture of wired and wireless networks using different communication paradigms.

The heterogeneity of networks, hardware, software, services, and information makes it a challenging task to provide a transparent computing system from the user point of view. Mobility means that some of the assumptions of how to create distributed systems are challenged. Wireless network connections are intermittent with varying bandwidth and quality. The mobile devices are resource-weak to make them wearable, and to allow them to operate on battery power. Sensors and actuators may be wireless and thus need to be inoperative in periods to save power.

To find technologies to support mobile work and work processes, we have looked into the traditional workflow and process-centred environments, and into technologies made to support collaboration in a distributed setting. Most of these technologies assume a stable execution environment with available and accessible resources, and partially pre-defined process goals and plans. This will often not be the case when mobile. A mobile environment is characterised to be very dynamic in terms of available resources and artefacts.

Different mobile actors appear and disappear in a partly non-deterministic fashion, and the physical environment itself naturally changes as we move within it.

Work support systems have traditionally been focusing on providing detailed work plans spanning from a few seconds to last for several months. These plans force and guide people to work and cooperate accordingly by providing strict control/data flow. Aspects related to how actions influence the environment, and how the environment can change the actions performed within it, has little coverage in traditional workflow or process-centred environments. The omission of these aspects stems from the difficulty to incorporate the physical environment into computational models, and the missing awareness of how actions and the environment mutually influence each other.

This thesis addresses the challenges of incorporating, reacting, and adapting to dynamic context information in systems to support work processes in a mobile, dynamic setting. The papers in this thesis cover a quite broad scope, but the work has eventually lead to the notion of the *"smart" work process*. Smart work processes use available context information to support planned and/or ongoing activities, and to help re-plan to cope with the environmental state.

Situational information is also regarded as important to truly provide on-the-spot support for mobile activities. Situated planning [Bar97] and situated actions [Suc87] are two conceptual terms that both incorporate context as important to plan, coordinate, and perform activities within dynamic environments. Situated planning is based on *activity theory* and tries to integrate situated actions into activity theory.

1.2 Research Context: The MOWAHS Project

This work has been performed as a part of the Mobile Work Across Heterogeneous Systems (MOWAHS) project that started in March 2001. MOWAHS is a multi-disciplinary research project supported by the Norwegian Research Council where two research groups at the Department of Computer and Information Science at NTNU have worked together. The MOWAHS project team consisted of people from the database and software engineering groups.

The project goals of the MOWAHS project were threefold [CN00]:

G1 Helping to understand and to continuously assess and improve work processes in virtual organizations.

G2 Providing a flexible, common work environment to execute and share real work processes and their artefacts, applicable on a variety of electronic devices (from big servers to small PDAs).

G3 Disseminating the results to colleagues, students, companies, and the community at large.

The approach of the MOWAHS project was to iteratively:

- Define a flexible work environment for virtual organizations using heterogeneous devices, with support for processes, their artefacts and transactions.
- Implement a test bed for process support for virtual organizations, using XML-based and mobile agents.
- Use real scenarios to evaluate the environment, e.g., for software development and remote education. 25% of the work will be spent on formulating requirements and success criteria, and to perform empirical studies. In all this, MSc students will be actively used for implementation and evaluation.

The project results are mainly PhD theses, papers, reports, software, and Web material covering the following:

The work within the MOWAHS project has resulted in several internationally published papers and student projects. This thesis partly addresses all three goals, but with emphasis on support for mobile work processes – goal G2. Our work has partly addressed virtual organisations, but it has been hard to find a common understanding on how virtual organisations and mobility coincide. Thus, our work has concentrated on how to support mobile work processes irrespective of which organisational form they are performed within.

1.3 Research Questions

The main goals in the MOWAHS project were firstly helping to understand and to continuously assess and improve work processes in virtual organizations. Secondly, providing a flexible, common work environment to execute and share real work processes and their artefacts, applicable on a variety of electronic devices (from big servers to small PDAs).

Other considerations that naturally come with mobility are how we can model and build systems that support work ”anywhere, anytime”. Such systems should distinguish and utilise contexts like location, time, and environmental properties (either static or dynamic) to support work that is collaborative, parallel, or individual. In addition, they should support work activities that come as ad hoc opportunities.

Users expect that many of the properties in the fixed computing environment also retain in a mobile computing environment. Access to necessary and demanded information and the ability to persistently record information in a work situation are both requirements

that must be taken into account when building applications for mobile usage. Using this as a starting-point, this thesis will address the following research questions:

RQ1 *What are the special properties of mobile work* and *how do these influence how to build supporting technology*?

- **RQ1.1** *How can concepts and frameworks be developed and used* to understand and specify *how mobile work processes can be supported*?
- **RQ1.2** *How can such concepts and frameworks be used to support the development of mobile work support systems*?

RQ2 *How can work process and context information be modelled and utilised* in mobile, heterogeneous, and dynamic work environments?

RQ3 *How and which technologies can be used* to create an infrastructure to provide support of mobile work processes?

Propositions to these research questions are described in Chapter 5.1.

1.4 Research Methods

Several research methods have been used to conduct the research presented in this thesis. The research scope in this thesis started quite broad investigating different properties of mobile professions as well as how to develop and use mobile work support systems.

We have mainly developed prototypes to investigate and evaluate different properties of mobile work support systems. Prototype systems have been created to support different work scenarios. The engineering method [Bas92] has thus been quite important in our research. Chapter 3 presents in more detail the research focus and methods used in our research.

Two real-life user studies have been conducted: A broad, quantitative survey of the home nursing care with 173 respondents, and a qualitative case study of electricians. In addition, IT-support and other mobile professions have been investigated qualitatively to identify software requirements to support those user groups, construct software based on those requirements, and evaluate the software.

The scope of the research has eventually led to the proposal of a new research direction, suggesting novel challenges in integrating different technology/research trends. The research challenges include defining new theoretical and practical methods to develop sound computer systems that we believe will play an important role in the future computing environments.

We have advised and cooperated with many master students to build prototypes, and to perform data collection related to the survey and the case study. This has made it possible to investigate several different research directions during the research period. Importantly, this coverage has made us able to look through the immediate challenges related to mobile computing, and instead look at how and where the technological development is likely to lead with respect to support of mobile work processes.

Suggesting a novel research direction integrating different technological trends is daring, but we believe that the applicability and utility of work process support software will be much larger by integrating the perceived dynamics into the planning and enactment of work processes. It is still too early to suggest and study realistic work scenarios to prove the applicability of the suggested research field. Thus, it is difficult to identify proper research methods except from building prototypes that investigate architectures and properties that context-aware process-support software should possess.

Chapter 3 will present in more detail which methods that have been applicable for our research, and which methods that have been used to answer each research question.

1.5 Apparatus

We have as part of the research involved several professions that we have defined as mobile workers, and thus may be users of future work process support systems. The user studies have included the home nursing care, mobile physicians, emergency situations involving several different professions, IT-support, and partly software development and mobile academicians. Unfortunately, it has not been possible to in practice try out smart work process support systems because of economical, technological, and organisational constraints, and the time scope of the project.

1.6 List of Papers

This section shows a short summary of the papers part of the thesis. The papers are listed historically with the oldest first. Chapter 4.6 presents abstracts of the papers included in this thesis, while Appendix A presents abstracts of the papers not included.

P1 *The MOWAHS Characterisation Framework for Mobile Work*.
Carl-Fredrik Sørensen, Alf Inge Wang, Hien Nam Le, Heri Ramampiaro, Mads Nygård, and Reidar Conradi. In Proc. IASTED International Conference on Applied Informatics 2002 (AI'2002), Innsbruck, Austria, February 18-21, 2002, pages 258–264. (See Chapter 9, [SWL+02]).

P2 *Experience Paper: Migration of a Web-based System to a Mobile Work Environment*.
Carl-Fredrik Sørensen, Alf Inge Wang, and Øystein Hoftun. In Proc. IASTED International Conference on Applied Informatics 2003 (AI'2003), Innsbruck, Austria, February 10-13, 2003, pages 1033–1038. (See Chapter 13, [SWØH03]).

P3 *Requirement Indicators Derived from a Mobile Characterisation Framework* .
Heri Ramampiaro, Alf Inge Wang, Carl-Fredrik Sørensen, Hien Nam Le, and Mads Nygård. In Proc. IASTED International Conference on Applied Informatics 2003 (AI'2003), Innsbruck, Austria, February 10-13, 2003, 8 pages (Digital Proceedings). (See Chapter 10, [RWS+03]).

P4 *A Survey of Mobile Support Needs in the Home Nursing Care*.
Carl-Fredrik Sørensen, Bjørn Næss, Øyvind Sølberg Strand, Alf Inge Wang, and Reidar Conradi. In Proc. of the APAMI (Asia-Pacific Association of Medical Informatics) & CJKMI-KOSMI Conference 2003 – Ubiquitous Computing in Health Care: From wired to wireless, Daegu, Korea, October 20-22, 2003. Published in Journal of Korean Society of Medical Informatics, volume 9, supplement 2, ISSN 1225-8903, pages 390–395. (See Chapter 8, [SNØS+03]).

P5 *Novel Component Middleware for Building Dependable Sentient Computing Applications*.
Maomao Wu, Adrian Friday, Gordon Blair, Thirunavukkarasu Sivaharan, Paul Okanda, Hector Duran Limon, Carl-Fredrik Sørensen, Gregory Biegel, and Renè Meier. In Proc. ECOOP'04 Workshop: Component-Oriented Approaches to Context-Aware Systems, Oslo, Norway, June 14th 2004. Affiliated with 18th European Conference on Object-Oriented Programming (ECOOP 2004). (See Chapter 14, [WFB+04]).

P6 *A Context-Aware Middleware for Applications in Mobile Ad Hoc Environments*.
Carl-Fredrik Sørensen, Maomao Wu, Thirunavukkarasu Sivaharan, Gordon S. Blair, Paul Okanda, Adrian Friday, and Hector Duran-Limon. In Proc. ACM/IFIP/USENIX Middleware'04 workshop: 2nd Workshop on Middleware for Pervasive and Ad-Hoc Computing, Toronto, Canada, October 18-22, 2004, pages 107–110. Affiliated with ACM/IFIP/USENIX 5th International Middleware Conference, October 18-22 2004, Toronto, Canada. (See Chapter 15, [SWS+04]).

P7 *Mobile Peer-to-Peer Technology used to Promote Spontaneous Collaboration*.
Alf Inge Wang, Carl-Fredrik Sørensen, and Thomas Fossum. In Proc. 2005 International Symposium on Collaborative Technologies and Systems (CTS 2005) in conjunction with ICSE'05, St Louis, Missouri, USA, May 15-19, 2005, pages 48–55. (See Chapter 16, [WSF05]).

P8 *Using the MOWAHS Characterisation Framework for Development of Mobile Work Applications*.

Alf Inge Wang, Carl-Fredrik Sørensen, Heri Ramampiaro, Hien Nam Le, Reidar Conradi, and Mads Nygård. In Proc. Product Focused Software Process Improvement: 6th International Conference, PROFES 2005, Oulu, Finland, June 13-18, 2005. Editors: Frank Bomarius, Seija Komi-Sirviö, pages 128–142. Lecture Notes in Computer Science 3547 / 2005, ISSN 0302-9743, ISBN 3-540-26200-8, Springer-Verlag GmbH. (See Chapter 11, [WSR+05]).

P9 *Support of Smart Work Processes in Context Rich Environments*.
Carl-Fredrik Sørensen, Alf Inge Wang, and Reidar Conradi. In Proc. IFIP TC8 Working Conference on Mobile Information Systems (MOBIS) - 2005, Leeds, UK, December 6-7, 2005. Editors: John Krogstie, Karlheinz Kautz, and David Allen, pages 15-30. IFIP 191/2005, ISSN 1571-5736, ISBN 0-387-29551-8, Springer, New York, USA. (See Chapter 12, [SWC05]).

P10 *The Nidaros Framework for Development of Location-aware Applications*.
Alf Inge Wang, Carl-Fredrik Sørensen, Steinar Brede, Hege Servold, and Sigurd Gimre. In Proc. IFIP TC8 Working Conference on Mobile Information Systems (MOBIS) - 2005, Leeds, UK, December 6-7, 2005. Editors: John Krogstie, Karlheinz Kautz, and David Allen, pages 171-186. IFIP 191/2005, ISSN 1571-5736, ISBN 0-387-29551-8, Springer, New York, USA. (See Chapter 17, [WSB+05]).

The published papers can be downloaded from the software engineering group's publication list at the Department of Computer and Information Science, NTNU [Sof05] (http://www.idi.ntnu.no/grupper/su/).

1.7 Contributions

The contributions of this thesis can be divided into three main themes:

T1 Exploration and investigation of mobile workers.

T2 Investigation and development of concepts and frameworks to capture and describe properties of mobile work processes.

T3 Development of enabling technology to support "smart" work processes in pervasive and mobile, ad hoc computing environments.

The main contributions are:

C1 A characterisation framework for mobile work to understand and find requirements for mobile work processes.

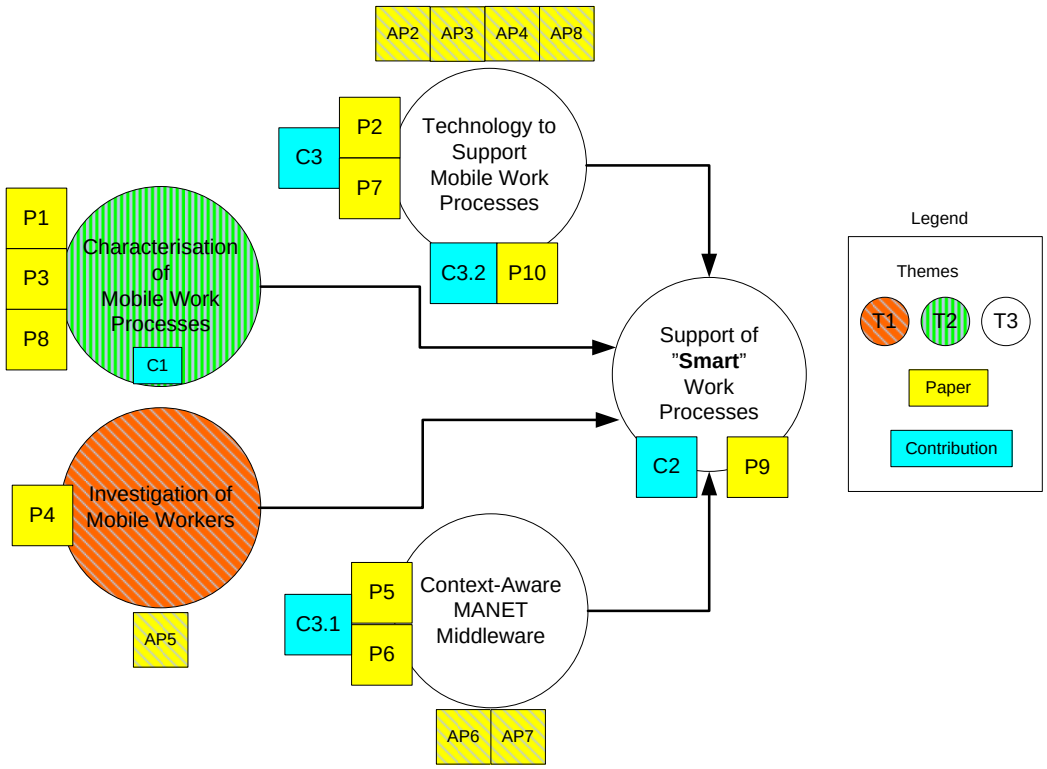

Figure 1.1: Papers and Contributions related to Research Themes

C2 An architecture to support context-aware, mobile (smart) work processes.

C3 Enabling technologies to support mobile work.

C3.1 A context-aware middleware for mobile ad hoc environments to support context-aware, mobile (smart) work processes.

C3.2 The NIDAROS framework for location-aware support on mobile devices.

Figure 1.1 shows the different research themes, and how the papers and contributions relate to each theme and each other. The figure partly shows the chronology of the papers.

Table 1.1 shows how the contributions and papers relate to the research questions.

Table 1.2 shows how the different papers are mapped into the three themes. The **X** indicates to which theme the paper belongs, while the **x** indicates that the paper partly belongs to the indicated theme.

Research Questions	Contributions	Papers
RQ1	C1, C2	P1, P2, P4, P9
RQ1.1	C1, C2	P1, P3, P8, P9
RQ1.2	C1, C2, C3	P2, P8, P9, P10
RQ2	C2, C3.2	P9, P10
RQ3	C3	P2, P5, P6, P7, P9, P10

Table 1.1: Papers and Contributions vs. Research Questions

Paper	T1	T2	T3	Comments
P1	x	X		A characterisation framework for mobile work.
P2	x		X	Experiences from development of a system to support mobile IT support.
P3		X		Requirement indicators based on the characterisation framework for mobile work.
P4	X			A quantitative user study of the home nursing care.
P5			X	A context-aware middleware for use in mobile, ad hoc computing environments.
P6			X	A context-aware middleware for use in mobile, ad hoc computing environments.
P7			X	Mobile peer-to-peer technology used to promote spontaneous collaboration.
P8		X		Experiences from using the characterisation framework presented in P1 and P3.
P9	x	X	x	The ”smart” work process, a context-aware, adaptive work process.
P10			X	A framework to support location-aware applications on heterogeneous mobile devices.

Table 1.2: Papers vs. Research Themes

1.8 Thesis Structure

This thesis is a paper collection where the main contributions are described through 10 papers. Further, the thesis contains a part setting the work in context and summarises what has been done. The thesis is organised into six parts as following:

Part I describes the motivation and context of the thesis, and outlines the research questions (this chapter). Chapter 2 presents basic concepts and theory related to our approach, and describes state-of-the-art relevant to the thesis, whereas Chapter 3 contains a summary of various research methods that can be applied in Software Engineering, and then describes the research focus and methods used in this thesis. In Chapter 4, the thesis contributions are summarised, and Chapter 5 evaluates our contribution. Future work is presented in Chapter 6, and Chapter 7 gives some concluding remarks.

Part II contains a *survey paper* which emphasis experiences and support needs in a mobile work profession (the home nursing care).

Part III contains *conceptual and framework papers* with models, methods, frameworks, and taxonomies developed to support the process of developing software and architectures for mobile work support applications.

Part IV contains *enabling technology papers*, and forms the experiences and results of technology development to partly evaluate the different research questions.

Please note that the format of the papers in Parts II-IV has been changed to the same format as Part I, and that all references are placed in Part VII.

Part V contains an appendix.

Part VI contains a glossary of terms and abbreviations.

Part VII lists all references used in the thesis as a bibliography.

CHAPTER 2

Theoretical Background and State-of-the-Art

This chapter gives an introduction to the research fields and theory relevant for this thesis. The research fields covered are *mobile, nomadic, pervasive, ubiquitous computing*, and *context-aware mobile computing*. Sections 2.1 to 2.5 present central concepts and general challenges related to these concepts.

This chapter also gives an introduction to work related to our approach. This includes different approaches for *systems to support mobile work*, and *context-aware systems*. Sections 2.6 to 2.8 present these approaches and some relevant research challenges. Finally, Section 2.9 presents a summary of the chapter.

2.1 Mobile Computing

Mobility is a term that denotes the *physical movement* of an object from one place to another in the desire to obtain resources or to move away from the scarcity of them [MDW99]. Mobility can involve both *physical* and *logical* objects [RPM00]. Physical objects can be people or mobile computing devices migrating in the physical space. Logical objects can be a running software application (process) or a mobile agent. These may migrate in a local area network, or to anywhere in a global network like the Internet.

Mobile computing involves the use of portable computing devices capable of wireless communication/networking such as laptops, palmtop computers, and modern mobile phones or smart phones [FZ94]. La Porta [La 02] introduces **mobile computing** as:

> **Mobile computing** *is a confluence of communication technologies (particularly the Internet), computing devices and their components, and access technologies such as wireless. A mobile computing environment will include not only real-time mobility of devices, but also mobility of people across devices. Therefore, the environment includes a wide range of devices, applications, and networks.*

A *mobile computing system* is referred to as the resulting distributed network from the integration of mobile communications and computing [CL97]. Mobile computing systems are in more that one way fundamentally different from conventional wired computer networks: Wireless connectivity enhances the functionality of computing equipment by freeing communication from the location constraints of the wireline infrastructure. By changing this basic characteristic, mobile computing systems operate on a set of assumptions made by traditional computing systems, requiring researchers and users to redefine their models of networked computing.

Mobile computing covers, as described above, a rather broad area of technologies and disciplines that also includes telecommunications and hardware manufacturing. Research within these specific disciplines is not covered in this thesis. The emphasis of this section is issues related to the support of physical mobility.

Forman and Zahorjan [FZ94] state that the challenges to design software for mobile computing systems stem from three essential characteristics of mobile computing: communication, mobility, and portability. Three aspects of physical mobility is the focus of [MDW99]:

1. *Weak connectivity:* This aspect covers how a computer can operate when disconnected from the network, intermittently connected, or connected over very slow communication links.
2. *Wireless connectivity:* This aspect deals the movement of computers between ”cells” in a wireless network.
3. *Ubiquitous computing:* (see Section 2.3).

A key requirement is to make access to critical information available regardless of the location of the clients. Satyanarayanan [Sat93] states that the need to access shared data implies interdependence between the elements of a mobile computing system. At the same time the need for robustness when encountering network and remote site failures requires clients to be as autonomous as possible. By its very nature, mobility exacerbates the tension between autonomy and interdependence so characteristic of distributed computing.

The research and development within mobile computing have moved from wireless connectivity to support for ubiquitous computing within a wireless world. The Wireless

World Research Forum (WWRF) is a forum for worldwide cooperation between academia and industry to identify and scope research issues relevant to future mobile and wireless communications, including pre-regulatory impact assessments [Wir05b]. WWRF has published "Book of Visions" in 2001 and 2004 that describe a common vision of the forum, including research topics to enable the vision.

The Wireless World Initiative (WWI) [Wir05a] is a European organisation established to build a path to the vision described by WWRF. The path is based on evolving current systems, enabling continuous evolution and innovation in applications and services, as well as investigating new concepts and technologies, which prepare the way for a range of new international standards and products. WWI focuses on developing personal environments to enable support for single users and their needs. This is accomplished through development of architectures and networks for an ambient wireless world.

2.1.1 Technical Challenges to Mobile Computing

Designing systems to support the nomadic, mobile, and wireless users, involves some key system parameters that influence both the users and the systems. Many of these are usual concerns for any computer-communication environment [Kle96]:

Communication:

Bandwidth, latency, reliability, error rate, delay, interference; i.e. Quality of Service (QoS) attributes. [CL97] state several challenges to both the mobile telecommunications and computing fields for the successful use of mobile computing:

- The communication link between the mobile host and the base station is unpredictable and varies greatly due to the constantly changing location of the mobile nodes and interference of non-network entities such as buildings.
- The topology of the network changes rapidly due to the movement and resulting connections and disconnections of the mobile nodes.
- The available bandwidth is limited and variable, depending on location, time, and number of users.
- The power available to mobile nodes is limited and, as a result, the power required for transmitting and receiving must be minimized. These constraints are unique to computer networks designed for mobile nodes, and need to be addressed to operate mobile computer networks with a large number of subscribers, each moving at various speeds across cells while requiring different types of service from the network.

Mobile device:

Storage, processing power, user interface, and limited battery life. [Sat93] describes some of the challenges of mobile computing that are intrinsic and not just shortcomings of the current technology:

- Mobile elements are resource-poor relative to stationary elements. Regardless of future technological advances, a mobile unit's weight, power, size, and ergonomics will always render it less computationally capable than its stationary counterpart. While mobile elements will undoubtedly improve in absolute ability, they will always be at a relative disadvantage.
- Mobile elements are more prone to loss, destruction, and theft than static elements. Even if security is not a problem, portable computers are more vulnerable to loss or damage.
- Mobile elements must operate under a much broader range of networking conditions. A desktop workstation can typically rely on local or wide area connectivity (LAN/WAN).

Others:

Interoperability between mobile devices, and between the supporting infrastructure and the mobile devices. The cost of using mobile networks and services. The cost of purchasing mobile equipment.

The parameters described above are of special interest because the values of these parameters may change dramatically as the user changes location as well as mobile device and wireless networks.

2.2 Nomadic Computing

Nomadic computing [Kle95] is described as **anywhere, anytime computing** [LSG96]:

> "*Following the explosive growth of cellular telecommunication and paging services, there is an increased interest in* **anywhere, anytime computing**, *often called* **nomadic computing**. *The goal is to provide users with access to popular desktop applications, applications specially suited for mobile users, and basic communication services in a mobile, sometimes wireless, environment. Nomadic computing is enabled by the advancement of portable computing devices, such as laptop computers and Personal Digital Assistants (PDAs).*"

Nomadic computing is related to mobile computing. The main difference is that mobile devices often may be stationary long enough to be physically attached to a wired network or to a local WLAN for faster and cheaper communication. In our work, nomadic workers that need access through wireless communication systems are of most interest, even though our emphasis is not on access to popular desktop applications. We focus more on actual support of mobile work processes than use of "normal" office applications.

The essence of a nomadic environment is to automatically adjust all aspects of the user's computing, communications, and storage functionality in a transparent and integrated fashion [Kle96]. **Transparency** is with respect to the current location, the used communication device (modem, Ethernet card, etc.), available communication bandwidth, the used computing platform, and whether or not in motion. The notion of transparency refers to the perception of a computing environment that automatically adjusts to the processing, communications and access available at the moment. E.g., changes in the bandwidth for moving data between a user and a remote server; or the computing platform available to the user that may vary from a low-powered PDA to a powerful supercomputer. Moreover, [Kle96] states that the ability to accept partial or incomplete results is an option that must be made available due to the uncertainties of the informatics infrastructure.

Nomadic systems can be regarded to be something between fixed and mobile distributed systems. The devices are mobile, but most of the time they are using a core infrastructure that is stationary and wired. Services to the mobile devices are mostly provided by the wired network.

2.3 Ubiquitous/Pervasive Computing

The term **ubiquitous computing** is based on a scenario depicted by Mark Weiser and others at Xerox PARC to reflect what could happen when computers are so small and cheap that they fade into the background. Weiser starts his article [Wei02]:

> "*The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.*"

Pervasive computing and *ubiquitous computing* have similar meanings and is therefore often treated as synonyms [Sat02a]. [LY02] distinguishes the different terms *mobile, pervasive, and ubiquitous computing* as shown in Figure 2.1. Here the terms are mapped into two dimensions where the dimensions are the level of mobility, and the level of embeddedness. The authors argue that the terms are conceptually different and employ different ideas of organising and managing computing services.

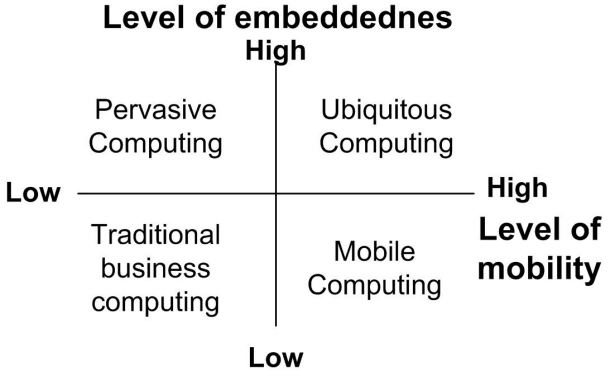

Figure 2.1: Dimensions of ubiquitous computing [LY02]

Since the vision of *ubiquitous computing* was presented in 1991, many products for ubiquitous computing in both the hardware and network area have become available. This makes the vision more viable and realistic. We observe that many of the devices used in the area of mobile and/or nomadic computing already can be looked upon as ubiquitous. The mobile phones and Personal Digital Assistants (PDA) are ubiquitous because of the widespread use in many segments of the modern society [AIM05]. Such devices often have computing power comparable with early PCs. This demonstrates how the computer disappears into mobile equipment primarily used for other purposes like, e.g., the mobile phones and personal organisers. We hardly notice that our mobile phones actually are equipped with computing capabilities [BGH02].

2.4 Mobile Distributed Systems

The last decade has changed the way software is constructed by the introduction of reusable and distributed components to compose software solutions. The idea of component use is analogous to how hardware is constructed. Many definitions of what a component is have been proposed. The core of many of these definitions is that a **component** is *a self-contained entity that exports functionality to its environment and may also import functionality from its environment using well-defined and open interfaces*. An **interface** is in this context *the syntax and semantics of the functionality it comprises* [Emm02], i.e., the interface defines and provides the access points of a component.

Modern program systems like Internet-based and enterprise applications offer multi-tiered, component-based architectures that incorporate middleware for distributing components across heterogeneous platforms. The platforms range from mobile devices like PDAs,

laptops, and mobile phones; to ubiquitous devices like televisions, refrigerators, and cars; to different types of stationary computers like mainframes and PCs.

The definition of a distributed system (see Part VI) applies both for fixed and mobile distributed systems. The differences between these can however be attributed to concepts like type of device, network connection, and execution context [MCE02].

A device is either fixed or mobile; mobile devices are normally resource-poor compared to their fixed counterparts.

The network connection is typically permanent and high-bandwidth for fixed devices, while intermittent and with varied and often low bandwidth for mobile devices. Disconnections in fixed systems are either for administrative purposes or caused by unpredictable network failures (treated as an exception from normal behaviour). For mobile systems, disconnections cannot be treated as exceptions but be regarded as a normal behaviour (disconnection because of missing network connection, or voluntarily to save, e.g., battery power).

The *execution context* can be regarded as either static or dynamic. Context is here treated as everything that can influence the behaviour of an application. This includes both external resources like location, proximity of services, or network parameters (QoS), and internal resources like the device hardware (see also Section 2.5). For fixed systems, the execution context is more or less static, while very dynamic for mobile systems. [MCE02] distinguish distributed systems based on these concepts into *traditional*, *nomadic*, and *ad-hoc* distributed systems.

Mobility should optimally be made completely transparent to most user applications, while for context-aware systems (see Section 2.5) mobility is an important source for contextual information related to both the electronic and physical environment.

Transparency eliminates the need to be constantly aware of the computing environment. Adaptation to a changing computing environment should be initiated by the system rather than by the user. Although perfect transparency is unattainable, that should not deter from striving to come as close as possible to that ideal [Sat93].

The concept of *adaptation* within (mobile) computing can be explained by how the computing elements (clients/servers/applications) dynamically react to changes in their computing environment. Adjustment of the computational functionality as a result of a change in the environment means that the computing element is adaptive. The goal of total transparency which makes sense in a static execution environment is challenged by mobility. In many cases it is not reasonable to hide away context information like location from the mobile user or to applications that can utilise this kind of information. By hiding such information, middleware must make decisions on behalf of applications. An application can normally make better and more efficient decisions based on the application-specific information [CEM01].

In a mobile distributed system, a loose or tight collection of trusted information servers are connected via a fixed network to provide information services to a much larger collection of distrusted mobile clients over wireless and mobile networks [JHE99]. The mobile devices are connected to the network through communication links with very variable quality (type, QoS).

No fixed infrastructure is present for the mobile devices as is the case for fixed and nomadic distributed systems. The devices can be isolated, or evolve into ad hoc clusters of hosts communicating with each other or with the fixed distributed system. The connection between hosts are not transitive [MCE02], meaning that devices do not have the same set of connected hosts. The networks can be short-range within small ad-hoc groups, or larger as in, e.g., emergency networks in disaster areas. The connectivity strategies and technologies for the network may involve heterogeneous networks where devices may support one or more network types. The use of different connectivity strategies requires much more coordination than is the case in fixed networks. Supporting technologies are needed to provide services that can be used across heterogeneous devices, networks, and applications.

2.5 Context-Aware Mobile Computing

Applications with the ability to continuously adapt their behaviour in response to changes in the physical and electronical environment have gained popularity since the vision of ubiquitous computing was presented by Weiser in 1991 [Wei02], and the paradigm of context-aware computing was proposed by Schilit et al. in 1994 [SAW94].

Mobility implies changing location and may therefore also imply rapid changes in the physical and electronical environment of the unit of mobility. These changes may lead to new situations and opportunities which users, devices, and applications can utilise or adapt to. Context-awareness is thus most relevant when mobile because of the rapid changes in the physical environment.

2.5.1 Context

Context refers to information that attempts to describe something about the conditions of a physical and electronic environment in which an application executes. No clear boundary divides what is and what is not context, but the most interesting kinds of context are those that humans do not explicitly provide [AEH+02].

Context has been defined by using examples like, e.g., location, identities of people and objects, and changes to these objects [ST94, Sch95]. Context has also been explained by using synonyms like environment or situation. Others are more specific [SAW94] who

claim that the important aspects of context are: where you are, who you are with, and what resources are nearby. Dey and Abowd [Dey01] criticise these definitions to be difficult to apply when building context-aware applications. They therefore provide this definition of **context**:

> *Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application themselves.*

Context is thus typical the location, identity, and state of people, groups, and computational and physical objects. This means that the context of an entity may be quite comprehensive. Attributes such as physical location, physiological state (e.g., body temperature and heart rate), emotional state, personal history, daily behavioural patterns, etc., may be of interest or relevant for a user and thus be included in supporting applications [Sat02b].

Dey and Abowd identify three entities to be the most relevant: *Places* – regions of geographical space, *People* – individuals or groups, co-located or distributed, and *Things* – physical objects, software components, or artefacts.

[DA01] describes four essential categories or characteristics of context – *Identity*, *Location*, *Status (Activity)*, and *Time*.

Identity refers to the ability to assign a unique identifier to an entity. *Location* refers to geographical position, orientation and elevation, and information that can be used to deduce spatial relations between entities (e.g., co-location, proximity, containment). *Status* identifies intrinsic characteristics of the entity that can be sensed, e.g., current temperature or light conditions at a place). For a person, status can refer to physiological factors, or the activity the person is involved in. For a group, it can be a description of activity like looking at a football match. *Time* helps to characterise a situation by indicating an instant or period which some other contextual information is known or relevant. The relative ordering of events or causality is, however, in some cases sufficient. The chosen categories can be used to infer additional context information to be used for a more extensive assessment of a situation. Inference or derivation can be used to find related context information from known information.

[Nar96b] defines context as everything related to an activity, including all participating actors, artefacts, and the activity itself. We can by the two understandings of context, provide a context framework for mobile work processes that includes the computational and the physical environment, the people and artefacts within the environment, and their activities within the environment.

2.5.2 Context-Awareness

Context-awareness refers to the properties of a system that make it aware of the state and surroundings of its user, and help it adapt its behaviour accordingly [Sat02b]. [Dey01] defines a system to be *context-aware* if *it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task*.

With advances in sensing and automated means of perceiving the physical environment, much more implicit content can be automatically collected [AEH+02]. In pervasive computing, this topic is central because a system that strives to be minimally intrusive must be context-aware. A key challenge of a context-aware system is obtaining the information needed to function in a context-aware manner. In some cases, the desired information may already be part of a user's personal computing space, like schedules, personal calendars, address books, contact lists, and to-do lists. Dynamic information must be sensed by systems in real time from the user's environment – such as position, orientation, people's identities, locally observable objects and actions, and emotional and physiological states [Sat02b].

A critical element in context-aware systems is to automate the collection of context information that can help direct the behaviour of a computational service [AEH+02]. Issues regarding the collection of context information are first to define what is of interest for a user, a situation, and to an application scenario, and then find means of collecting the relevant information in an automated manner. Since many sources of context information probably will be available in the future, the supporting systems must be able to filter the relevant information according to many different situations and scenarios. Context information may be competing, contradicting, or totally irrelevant for the user. Since a person plays many different roles and is involved in many processes during a day, the user context itself is changing, and with it the preferences, wishes, and needs for computational support.

[Dey01] presents three categories of features a context-aware application can support:

- *Presentation* of information and service to a user;
- Automatic *execution* of a service for a user; and
- *Tagging* of context to information to support later retrieval.

The first category refers to applications that either present context information to the user or use context to propose appropriate selections of actions to the user. Context-aware applications like tourist guides [LAAA96, AAH+97, CDMF00], and museum guides [FFK+02] belong to this category.

The second category refers to applications that trigger a command or reconfigure a system

on behalf of the user according to context changes. Adaptive middleware [BCD+97, Agh02] and applications belongs to this category.

The third category refers to applications that tag captured data with relevant context information, like time and location of the recording of a video, audio or other kind of digital information.

Other context-aware applications presented in the literature are, e.g., conference assistants [DSAF99], and context-aware chat programs [RCRM02, IWR02b, IWR02a]. Most of the context-aware applications, however, mostly deal with location-awareness, and have in lesser degree included other kind of dynamic context information.

2.5.3 Human Considerations of Context-Aware Systems

The notion of context-aware systems seems intriguing, but the design and use of such systems also introduces several issues related to how these impact humans, their processes, their privacy, and the accountability and dependability of the systems.

[BE01] discusses human considerations in context-aware systems and argues that complex machine inference based on human context is a difficult proposition, especially when the system attempts to take action based on a presumption of human intent. It is difficult enough to make reliable inferences about the nature and status of environmental conditions and systems, but to do the same for human and social context is extremely complicated. Appropriate system actions might depend on a myriad of factors that often cannot be reliably sensed or inferred. These factors might be very difficult or impossible to represent in a structured and deterministic way to make computers able to take autonomous actions on behalf of humans. They therefore propose that two key features must be supported in any context-aware infrastructure to allow users to make informed decisions based on context:

- Intelligibility: If systems seek to act upon what they infer about the context, they must be able to present to users what they know, how they know it, and what they are doing about it.

- Accountability: When systems seek to mediate action that impact others based on their inferences about the social context, they must enforce user accountability.

[BE01] further argues that context-aware systems should be built on a set of design principles to enable humans to reason for themselves the nature of their systems and environments, empowering them to decide how best to proceed. The reason is that designers are unlikely to be successful at representing human and social aspects of context in a deterministic fashion.

If we allow systems to take initiative to activities when human participants are involved, we might introduce genuine risks to the safety and privacy of the humans. The many variations of what can be done under different conditions make it very hard to in advance model appropriate outcomes.

2.6 Technologies to Support Work Processes

This section will look into different approaches to support work in general, and more specific, mobile work. A precise definition of *mobile work* is hard to find. In general, all work that is performed in a mobile situation can be characterised as a kind of mobile work. This includes:

- Work that is *inherently mobile*, i.e., the job itself requires mobility. This category includes drivers, pilots, etc. Adapting and adjusting to change of environmental context is considered extremely important. We regard this as *true mobile work*.
- Work requiring *change of location* to be performed. This category includes all kind of physical work, including carpenters, electricians, the home nursing care, emergency personnel, etc. The actual work is most often performed at a single site, but the site may change several times a day. Adapting and adjusting to change of environmental context is considered important. Even micro-mobility, e.g., turning away from the computer, can be characterised as mobile work since the work context changes because of the change of location. We regard this as *service work*.
- Work performed when not at the normal work place like, e.g., the office. In this category, *nomadic work* can be included. Adapting and adjusting to change of environmental context is not considered important.

In paper P9 (Chapter 12), we distinguish between work that does require change of context and work that does not require it. We characterise mobile work to be *dependent* of change of location and thus change of context. Other work performed when mobile is not characterised as true mobile work when the work can be performed independent of location or context. Nomadic work is thus not true mobile work by our definition. All work performed when in a mobile situation may, however, require similar technological and computational support. That means access to mobile computers, wireless networks, and ways to access or distribute information. We assume that all occupations in the future will have some form of computational work support.

From our characteristic of mobile work, we claim that *true* mobile work in addition to access to mobile technologies, requires management and support for context information derived from the physical work environment. This can be illustrated as shown in Figure

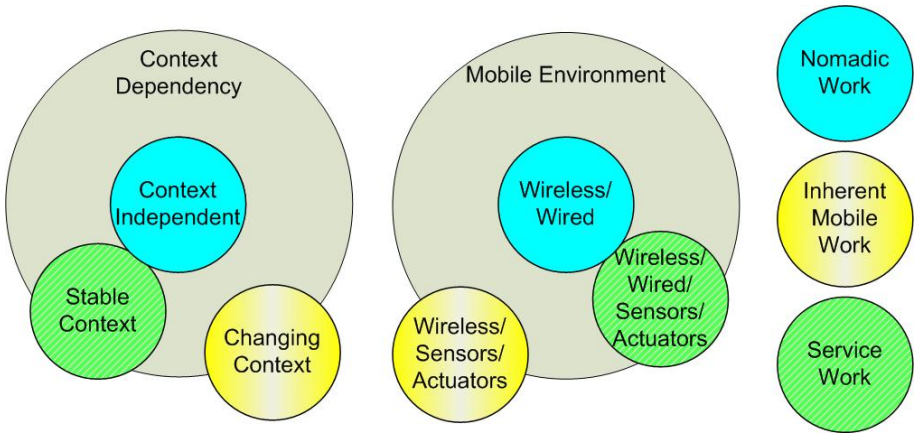

Figure 2.2: Mobile Work vs. Working when Mobile

2.2. The complexity of the computing environment increases with context dependency, required instrumentation of the environment, and necessary access to different types of mobile technology.

The type of computational support required for mobile workers is therefore different, but have all in common the need to access information. In ”anytime, anywhere” computing, a lot of research and attention have been directed to how to make documents and other information accessible and updatable to the mobile users. For mobile workers, the same functionality is required, but in addition there is a need to access or subscribe information in an ad hoc manner. In work process support systems, context information derived from the environment is to a small degree incorporated and integrated to perform workflow enactment and support the work processes. CSCW (Computer Supported Collaborative Work) systems do, however, include (limited) awareness as an important property for how to set up cooperative environments. In paper P9, we introduce the concept of a **smart work process** to capture adaptive and context-aware work process support.

[WL01b] has investigated how mobility and the concept of the global village[1] are related by conducting an empirical study of mobile telecommunication engineers. The tasks of the engineers were investigated to find to what extent these are dependent of time and place. [WL01b] concluded that the practical limitations of ”anytime, anywhere” made it impossible for mobile service engineers to conduct work ”anytime, anywhere”. **Mobile work** is unavoidable for many because physical presence is necessary to conduct their tasks. Work also often has to be performed within certain time limits that are not negotiable.

[1]The global village is described by Preece [Pre94] as where the convergence of computing and telecommunications has brought about an interconnected worldwide society.

Based on this characterisation of work, [WL01b] states that ”anytime, anywhere” do not necessarily mean ”everytime, everywhere”. The ideal mobile situation is not to work continually without any stops. Further, true mobility goes beyond mobile support for ”here and now”. There is also a need to support the place to go, and the place left behind as well as to make plans for the future or backtracking earlier events. Mobile work is in many cases a kind of stationary work because the worker has to stop to perform any real work when work is physical oriented.

2.6.1 Process-Centred Software Engineering Environments

A Process-Centred Software Engineering Environment (PSEE) provides automated support for development and management processes for software production activities. The development process will change due to changes in the work environment, while the management process standardises ad hoc routines in the work environment.

PSEE integrates production, process, and management technology into an engineering environment used by people in the work environment [Wan01]. There might potentially be a large group of developers working on a common project. These may be spread at several locations and time zones. Key aspects of PSEEs are how they are able to support *collaboration* between software developers, and *coordination* of development activities.

The need to represent software processes has motivated researchers to investigate different ways to model such processes and to support their execution. The complexity of the software processes has led to the creation of a number of formalisms and languages to model them. These are often called *Process Modelling Languages or PMLs*. Some of the elements represented by a PML are [Fug00]:

- Activities performed to accomplish the *process objectives/goals*.
- Process models and rules.
- Roles of the people in the process.
- Structure and nature of artefacts.
- Tools to be used.

The PMLs can be used for different purposes [Fug00] like process understanding, elicitation, formalisation, analysis, simulation, optimisation, design, and configuration. Other aspects PMLs may be used in are, e.g., training and education, process enactment, and process evolution. PSEE or the more general PCE (Process-Centred Environment) includes an engine to interpret PMLs, and assists users in performing the process instances. The assistance might come in different forms like automation, enforcement of consistency

constraints, guidance, analysis, simulation, monitoring, and measurement of the process [BSK95].

The research area on how to support distributed, decentralised, heterogeneous, flexible, and partly mobile processes has recently been popular within the software process community [BCN+96, BSK98, CDE+00, Tia98].

2.6.2 Workflow

Workflow is a technology as well as a discipline, practice, and concept [All01]. **Workflow** is defined by the Workflow Management Coalition (WfMC) as [Coa99]:

> "*The* **automation** *of a business process, in whole or part, during which documents, information, or tasks are passed from one participant to another for action, according to a set of procedural rules.*"

Allen [All01] segments workflow into *production*, *administrative*, *collaborative*, and *ad hoc* workflow, based on the specific aspects in each of these segments.

The term *process definition* is used within a workflow to denote the model of a process. A **Process Definition** is defined as [Coa99]:

> "*The representation of a business process in a form which supports automated manipulation, such as modelling, or enactment by a workflow management system. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc.*"

A process definition is thus similar to a PML as described above.

A **workflow management system** (WFMS) is defined by [All01] as:

> *"A system that defines, creates and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants, and, where required, invoke the use of IT tools and applications."*

A WFMS is proactively managing a flow of work activities performed by different actors, by following defined procedures to achieve a business goal or objective. Users are given different roles, and these are used to control the participation of the users in a workflow together with appropriate data-management resources [DMPD99]. The resources can

be accessed through the WFMS or through external applications. The goal or objective of a workflow can be achieved by contributions from deadlines, activity synchronisation conditions, and passing activity data from one participant to another in a correct sequence.

Conventional WFMS usually work in different phases by first defining and specifying the workflow. This phase is followed by an execution and activation phase, performing the specified workflow. [DMPD99] describes how this basic approach is evolving into two different extensions. The first extension is to use certain forms of organisational knowledge to adjust particular activities and to cater for unexpected input or results in each task. The second extension aims to make the definition phase adaptive to allow dynamic tailoring of the activity definition when the activity is executing, to satisfy unexpected goals, conflicts, and divergences of the activity performers.

The workflow community has focused on building systems that support and manage distributed work processes. Workflow architectures have historically been either message-based or database-based [Bol00]. In later years, the Web has been used as the underlying execution and deployment platform.

The WfMC workflow reference model [Hol95] describes a workflow management system architecture to consist of interacting modules, which are the process definition, the workflow engine(s), the administration and monitoring tools, the invoked applications, and the workflow client application. Figure 2.3 shows the generic workflow architecture proposed by WfMC, with the main components and interfaces.

The reference model separates the workflow enactment service, where the workflow process instances are processed, from the other functions which must be performed by a workflow management system through the use of interfaces. *Interface 1* provides the process definition for the workflow enactment service. *Interface 2* provides the workflow participant with a front end to the system, to make, e.g., work lists available. *Interface 3* allows the data belonging to activities to be processed in supported applications. *Interface 4* provides the interface to other workflow systems. *Interface 5* provides administration tools to control the workflow processing.

2.6.3 Computer Supported Cooperative Work (CSCW)

Computer-Supported Cooperative Work (CSCW) is a multi-disciplinary research area focusing on effective methods of sharing information and coordinating activities [EGR91]. CSCW involves three aspects:

- Communication between involved parties: often low-level control flow.
- Coordination where participant activities are planned, scheduled, and monitored, and where negotiation and arbitration also is necessary, i.e., mostly high-level control flow.

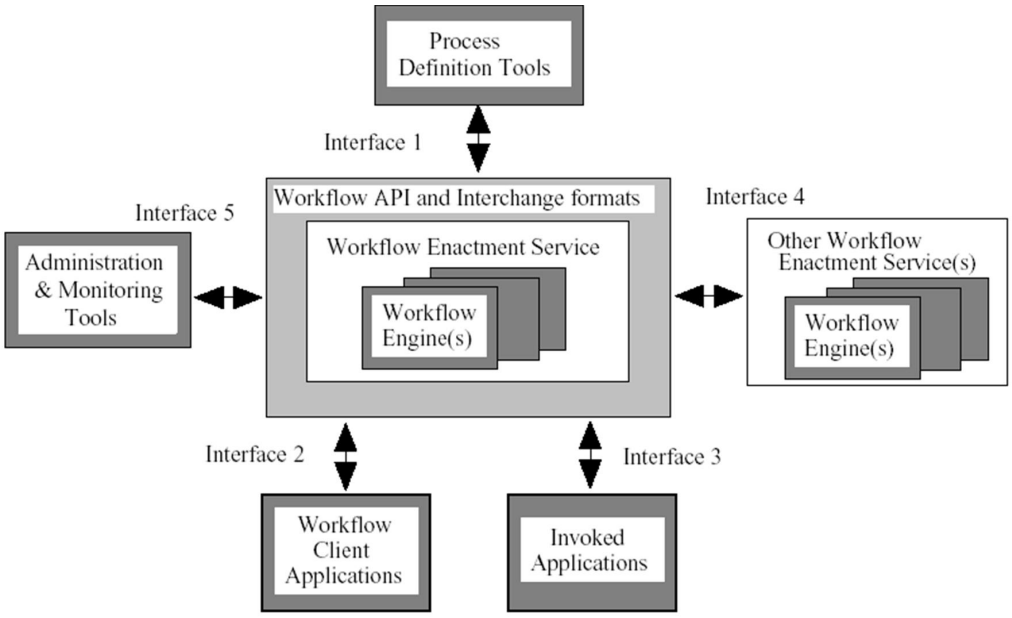

Figure 2.3: WfMC Reference Model - Components and Interfaces [Hol95]

- Collaboration, sharing, and exchanging of data and plans: mainly data flow at all levels.

CSCW focuses on human-to-human communication and interaction. Social aspects of collaboration and tools to enhance collaboration and coordination are very important. Much CSCW work have been characterized as being synchronous and distributed. [LC98] added an extra dimension to the CSCW typologies, where different kinds of cooperative work are characterized by the increasing complexity of the process support they need:

1. *Ad-hoc cooperative work* such as brainstorming, cooperative learning, informal meetings, and design work, etc. The related process support is often implemented by interactive blackboards and awareness triggers [DT93, SD97]. Further, there is little formal product modelling to explain why certain people are ”related”, and therefore need to cooperate.

2. *Predefined/strict workflow*, in traditional *Office Automation* style, is often represented by simple document/process flow [Orl92].

3. *Coordinated workflow*, such as traditional, centralized software maintenance work consisting of check-out, data processing, check-in, and merge steps. There exist several systems supporting coordinated workflow (mostly as prototypes), e.g.,

EPOS [ea94], MARVEL and partly Oz [BSK95], and ADELE and its APEL language [DEA98].

4. *Cooperative workflow*, such as decentralised software development and maintenance work conducted in a distributed organization or across organisations. Here (partially) shared workspaces and cooperation planning – in addition to evolution and federation – are the main extra factors from the process point of view. A system supporting distributed organizations and processes is Oz [BSK95].

[BPS02] describes a collaborative platform for fixed and mobile networks called C/Webtop supporting a large range of mobile devices like laptops, PDAs, and smart phones with sufficient display capabilities. The system provides interactive services to support common activities of virtual teams. The system is based on Internet-technology (e.g. Java and TCP/IP), but enhances the classical Web collaboration mechanism to support interactive and real-time collaboration.

[Bus95] claims that asynchronous CSCW applications are especially suitable to run on mobile equipment, in particular applications that are aimed to support activity coordination. These kinds of applications already deal with part-time disconnection. [Bus95] states that it can not be assumed that all data belonging to an activity can be downloaded to a mobile device due to the memory/disc space limitations, and the restricted bandwidth in wireless communications. It is therefore a challenge to provide support for activity coordination where activity artefacts are not present on the device, or are only partly present on the device. [Bus95] presents an activity coordination tool called *Task Manager*, which aims to support collaborative work in distributed work environments. This tool supports specification and management of activities which are distributed both geographically and in time. This tool has no a priori defined structure of the sequence of actions or distribution of information. Every person who processes a common task may modify the work plan and change or add information at any time of task execution.

2.6.4 Research Challenges in Supporting Technologies for Mobile Work

[Fug00] challenges the value of the current PMLs and PSEEs because of the late adoption in the industry. One explanation to this phenomenon is the complexity of the current PMLs and subsequently the PSEE. This raises issues on how to create more effective, flexible, non-intrusive, and adoptable technologies in the (software) process area.

Other issues of interest are how to manage evolving models with, e.g., changing user configurations, workspaces, and cooperation/process models; and how to provide different views for different user categories: agendas, data structures, data/control flow patterns, etc.

Mobility increases the space of variation where the user context in terms of location, device and network availability and capability, and other environmental/contextual factors provide constraints and/or opportunities for how to access, change, and report with respect to the process models.

Management of distributed and mobile processes is very challenging as aspects like definition, monitoring, and coordination of the processes are not easy in dynamic environments with different infrastructures, ad hoc tasks, formation of *virtual organisations*, partly missing network connectivity, etc. Other challenges in this area are to create PMLs that are tolerant and allow for incomplete, informal, and partial specifications. Inconsistencies and deviations must be tolerated and managed by the supporting systems.

Most of the present process and workflow systems have the assumption that clients should be connected to a server when using the system, and that only one type of client computer is used. However, people using these systems are getting more mobile, and are using desktops, laptops, palmtops, and mobile phones to communicate. Existing process and workflow systems do not cope with the heterogeneity of equipment and mobility of users. It is therefore a challenge to give effective and flexible workflow support for users through a common, light-weight workflow framework. Workflow systems should support users working on different places and moving around, users working on-line and off-line, as well as users accessing the workflow system using different equipment.

Some relevant research challenges within the **process and workflow** community are:

- **Handling workflow changes when users are not connected**. Both the users and peers need to be able to access the workflow state of other peers when participating in a cooperative work. It is therefore necessary to provide methods and systems that are able to handle disconnected workflow [CN00].

- **Creating a workflow client accessible and scalable for heterogeneous devices** like PDAs, desktop computers, and mobile phones, and creating workflow views suitable for PDAs [CN00]. The need for different user interface metaphors on the different platforms makes it necessary to transform content between different view metaphors.

- **The main problems with low-resource devices are to be able to process, store, as well as communicate large amounts of information.** The availability of many different types of mobile digital devices that can be used to produce workflow artefacts makes it challenging to integrate information from the different sources/devices into supporting tools. To save battery power, or because of missing support for wireless networks, some of the information produced cannot be transferred in a timely or synchronous fashion to the target systems. Different kinds of information can as well be produced in parallel, making synchronisation even harder. It is however possible to treat this kind of information specifically by postponing the

information itself, but report about the availability. Cyber foraging [BFS+02] is a technique to offload the mobile devices by using the supporting infrastructure to perform resource-demanding tasks.

- **The evolution in context-aware and ubiquitous systems might have impact on future workflow systems** (see Section 2.5). Context might have direct impact on both the workflow process itself, its artefacts, as well as where, when, with, and by whom the process/tasks should be performed. Context adds a new dimension since ad hoc opportunities or tasks might create new workflows.

- **Synchronization between different devices after working off-line** is challenging when synchronisation means exchanging information that may be half-complete, or that has been postponed due to missing power or connectivity when producing the information. Another issue is to integrate information from different sources where the sources are not connected to each other (like a digital camera).

- **Support for impact and consistency analysis upon changes in the workflow system** based on events and changes in dynamic and mobile environments is challenging. Mobility and non-connectivity of devices create unpredictable changes related to both the time and content of the workflow.

The MOWAHS project proposal presented some research trends in cooperative work related to mobility [CN00]. E.g., how to provide infrastructures for collaboration, how to support mobility in collaboration, how to provide collaboration using mobile devices, and what the balance is between forced behaviour (automation) and non-obtrusive support (e.g., notifiers), and for what kind of processes.

[WWSL02] states that despite accomplishments in many areas, a shared collaborative system or workspace for real-time distributed environments is still missing in the literature. Real-time mobile environments are even less represented. Asynchronous systems like SMS still foster collaboration, but is unstructured and thus of little use for computer applications.

The availability and evolution of ubiquitous instant messenger/chatting systems like Yahoo Instant Messenger, Microsoft MSN Messenger, and similar systems have in the last years exploded, the same applies to the number of active users. These kinds of systems foster collaboration and are supporting multi-user meetings, video, audio, games, uploading and downloading of files, etc. These systems have lately also integrated mobility into their solutions to make it possible to chat by using mobile phones.

We have in this thesis worked on some of these challenges, especially the challenges related to content adaptation to heterogeneous devices, context-aware workflow systems, and partly cooperation using mobile devices. The next section will look in more detail into context-aware systems.

2.7 Context-Aware Systems: Support for Mobile, Context-Aware Work

Research on adaptable workflow systems have been performed in more than a decade. Systems of special interest are those that are able to handle exceptions and adapt the process definition in run-time. Research on context-aware work support is just starting by unifying research on CSCW, ubiquitous computing, partly mobile computing, workflow, and process-centred environments.

Adaptable workflow systems that are able to change their process descriptions have been studied in more than a decade [BDK99, KBT+00, Kam00, van01]. E.g., [AS94] describes the InConcert Workflow system with a searchable library of process definitions, and thus provides adaptable process definitions. [BK95] describes the *obligations* model as prototyped in the two collaborative environments, *ConversationBuilder* [KTBB92], and the *wOrlds* environment [TKF95]. Vague plans are supported through either local modifications or through general changes. [FRF02] describes ULTRAflow that is able to update process specifications in run-time.

Situated actions [Suc87] and situated planning [Bar97] are both terms that are related to our approach. It is very hard to make predictions of which actions to perform in very dynamic environments, i.e., the environment itself is influencing the actions based on the state of the environment. This motivates for a broader use of contextual information to support both the execution of actions, and also the planning of them. Situated planning can be used to refine coarse-grained plans based on the contextual properties of the environment and the worker itself. Situated actions can be used to create plans and used in situ.

Few applications that aim to directly support mobile work processes have been developed and commercialised, but some examples exist that partly are using location information to support the mobile work process. [SVLZ01] describes a case study of a system to support a mobile sales force by deriving customer information from the current location of the sales person. The location is indirectly made available through an infrared connection between the mobile phone and an infrared transmitter sending a short message with the customer identification. The customer ID is then used in the back-end system to fill in details about the customer and to reduce the length of the product list presented on the mobile phone. The solution is based on an online GSM connection.

Similar solutions are made available today by using *Radio Frequency Identification (RFID)* technology [Bor05] that can be used to tag all kinds of physical objects, including people. Location can be indirectly made available through location look-up in databases, and thus provide topological information for applications. This is useful for objects that have a limited mobility. Location information is thus static for the object tagged, but can be used by mobile actors to decide their location. Dynamic physical locations can be made

available through the use of location technologies like GPS (outdoors) or other systems (like GSM, WLAN) that use triangulation to derive an exact location of an object in a given coordinate system.

If we look at our definition of mobile work, we can split mobile professions into three categories and evaluate the impact of context within each type.

Nomadic work

The work in this area is quite old. However, most of the work has been concentrated on how to make the move of infrastructure context as transparent as possible. Solutions like Virtual Private Network (VPN) from Cisco Systems, Inc., provide a secure channel for making the nomadic environment as similar to the normal work environment as possible. It is mostly office applications that are of interest for nomadic workers. The change of context influence the availability of data and typical server applications, the context state itself is not used as a driving force for application adaptation.

Service work

The most supported group of mobile workers is people that perform service work. The MONGO system (see Chapter 13) belongs to this group. Many of the industrial applications supporting services workers are, however, office applications and not context-aware with respect to the environment.

Work in Lancaster, e.g., the MOST system [DBCF94, Fri96, CBDF99, FDBC99], supported field engineers in the UK power distribution industry by adapting the user clients based on the perceived QoS, and thus provide awareness of the connectivity of field engineers. This awareness could then be used to ensure safety of field engineers by avoiding dangerous operations while uncertain of the status of people in the field (e.g., turning the power on).

Other groups supported are, e.g., within health-care [Bar05, JB04], retail industry [NPS03], airplane inspection [LSF04], utility industry [HCL00], and telecommunication engineers [WL01b].

[JB04] presents requirements to the PHCS project in Århus, Denmark. The work of this project partly covers context-aware work processes in a hospital. The context is however not part of the environment, but is fetched using RFID sensor technology identifying objects (like a patient). Relevant data and the GUI are adapted based on the identification of the treating person (e.g., a nurse) and the RFID-sensed object. The adaptable GUI is similar to our paper presented in 13. In addition to an adaptable GUI, we make content adaptation based on device types.

True mobile work

Few applications have been developed for true mobile work. One of the reasons is probably the need for real-time response. However, some industrial applications in the car industry have been developed (see related work of the CORTEX middleware in Chapter 14 and 15) that sense the environment and make adjustments to the car operation. The driving process is not directly influenced by these solutions, but is informed/warned about changes in the environment (e.g., outside temperature approaching the freezing point $0\,^{\circ}\mathrm{C}$).

Applications used by, e.g., the US Army [ZJWF02], fire-fighters [JHTL04], and the police [KML04] are using context information (especially location) to direct and coordinate activities.

Entertainment/Leisure/Private

In this group, all applications that are made for entertainment or personal aid belong. Included in this group are different types of tourist guides, games, m-Commerce applications, and support for disabled persons. The main context information used in this group is co-location of people and physical artefacts, and location. The example application based on the NIDAROS framework (see Chapter 17) belongs to this group. A personal application is, e.g., support for visually impaired persons [HMR01].

2.7.1 Design of Context-Aware Systems

[DRD+00] presents a taxonomy of context where four forms of context have been associated with relationships to an interaction device and surrounding elements. Each form of context introduces different issues with respect to design of systems to support mobile users. The four forms are *infrastructure*, *system*, *domain*, and *physical* context. [DRD+00] has especially looked at how to exploit space and location as a design framework for interactive mobile systems. Further, [DRD+00] presents a framework to get an understanding of the location in space of devices and other bodies, mobility through space of devices and other bodies, the kinds of bodies populating space which devices may interact with, and the awareness of other devices. The framework consists of a taxonomy of location, mobility, and population. The spaces consist both of physical and virtual worlds, where computers and mobile systems exist and are present in many spaces by simultaneously inhabiting a real world and some form of one or more virtual worlds (like, e.g., the Web, the desktop, file systems, etc.). Physical manipulation (like changing the position) of devices generates in some systems computational or virtual effects, and thereby interleave the physical and virtual worlds. The interplay between the real and the virtual is suggested by [DRD+00] to be the core of the design of cooperative mobile applications, as devices and users have a location and presence that is both virtual and physical, and

since each is available to the computer application.

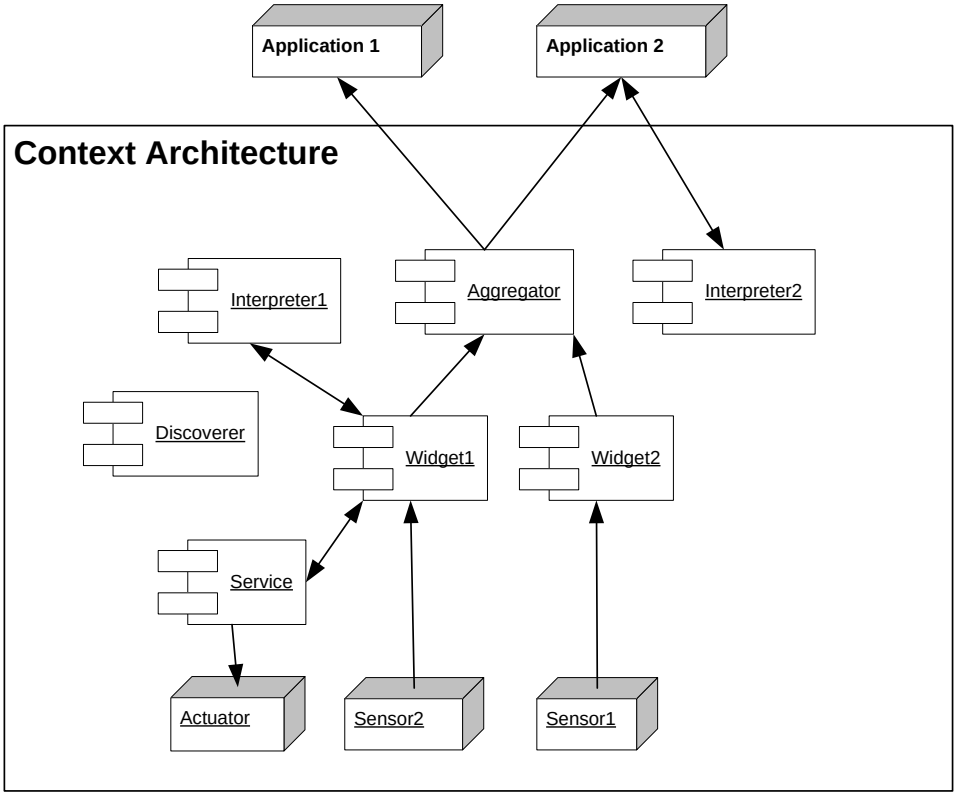

Figure 2.4: An Example Configuration of the Context Framework

[Dey00, DA01] present requirements and a conceptual framework for handling context information. The requirements to be fulfilled by the framework are *separation of concerns*, *context interpretation*, *transparent, distributed communications*, *constant availability of context acquisition*, *context storage and history*, and *resource discovery*. The conceptual framework for handling context presented by [DA01] suggests the use of **context widgets** to provide access for applications to context information from their operating environment. The context widget is regarded as a mediator between applications and the operating environment, insulating applications from context acquisition concerns. Context-specific operations are addressed in the framework by four categories of components: interpreters, aggregators, services and discoverers. Figure 2.4 shows an example architecture of the context framework.

2.7.2 Research Challenges in Context-Aware Systems

Implementing a context-aware system requires addressing many issues [Sat02b]:

- **How does the system represent context internally?** How do we combine this information with the system and application state? Where should the system store context – locally, on the network, or both? What are the relevant data structures and algorithms?
- **How frequently does the system need to consult contextual information?** What is the overhead of considering context? What techniques can we use to keep this overhead low?
- **What are the minimal services that an environment must provide to make context awareness feasible?** What are reasonable fallback positions if an environment does not provide such services?
- **What are the relative merits of different location-sensing technologies?** Under what circumstances should we use one and not another? Should we treat location information just like any other contextual information, or should we handle it differently? Is historical context useful?

Dey and Abowd [DA01] describe the difficulties of developing context-aware applications. First, designers lack conceptual tools and methods to account for context awareness. The available context acquisition mechanisms drive the choice of context information to be used in applications. The selected sensors might not be the most appropriate, and the details and shortcomings of the sensors may be carried up to the application level. Second, distribution, modifiability, and reusability are problems to be faced. Mobile devices are heterogeneous with different computing, communication, and user interface capabilities, thus context-aware applications require lightweight, portable, and interoperable mechanisms across a wide range of platforms.

To support a mobile worker, it is important to identify both what context is vital for the worker, how to collect this context, as well as how to use it when creating supporting systems for the mobile worker. The context in a mobile and dynamic environment might have a much higher influence on work processes than in a distributed, but static environment. As can be seen above, context-awareness has in a small degree been included in workflow, process, and cooperative work (CSCW) systems. CSCW has, however, to a much larger degree included context since CSCW deals with interactive cooperative work.

[DA00] states that the goal of context-aware computing is to make interacting with computers easier. The management of context should be automated to let the supporting applications deal with it, instead of the users. This perspective means that the application developer should both decide what information is of interest, as well as how to deal with

it. A context-aware, mobile work support application has to deal with context related to all the categories of context described above.

When building architectures to support mobile work, a number of issues arise. Many of the existing components are often network specific and fail to provide adequate performance over a range of infrastructures [DRD+00]. The interfaces to such components are also often application specific, making use of different kinds of, e.g., sensor components difficult. The information passed from sensors might often be too fine-grained for upper-level components. It is therefore a need to specify generic application architectures for part of the mobile domain to utilise different kinds of sensor components. Infrastructures modified from the fixed to the mobile setting are in danger of missing important utilities because of missing support for the dynamic situation when in mobile environment. Context-aware applications have to take into account contextual issues across the whole system design, from the infrastructure and system layer, to the domain and physical layer. For mobile devices to handle and reflect on many possible context sources, a lot of resources have to be spent. It is therefore important to find a balance between the utilisation of context and the consumption of resources.

We suggest that context information will have different importance based on which kind of context both the user and devices themselves are situated within. It is therefore a matter of utility, reliability and safety for how context information should be included into mobile work support applications. Important context information should pass to the user where non-influential context information is either filtered by the context source based on user/device preferences and capability, or through coordination services especially tailored for the user/device.

2.8 Relations to Software Engineering

This section describes the research field of software engineering, with emphasis on developing software for mobility as this is the focus of this thesis.

The production of software systems has the last decades changed from development of purely monolithic systems residing on large mainframe computers, to development of systems that are highly distributed over a large range of heterogeneous server and client types. The communication networks (both wired and wireless) become faster and more reliable as the time goes by. Mobile devices are delivered with processing power comparable to stationary computing devices, battery life for several hours of continuous work, and with a screen size and resolution that make them usable for a large range of tasks and activities. These developments have caused a demand to make the functionality and information from the current wired software systems available for the mobile and nomadic users. This evolution creates new challenges to the software engineering field to be able to produce applications that can be used on a broad and heterogeneous spectre of de-

vices and network connection types. Software engineering for mobility introduces several challenges to software developers. The most important reasons are scarcity of computing resources like processor power, memory, battery power, and unstable or missing connectivity for mobile devices. The user interface metaphors used in desktop applications are considered less usable when creating mobile user interfaces, mainly because of other and different means of data input and output. Mobile workers may, e.g., not be able to use both hands for operating computing devices. In hostile environments neither hands nor voice may be acceptable for I/O operations. This introduces challenges for how to provide usable computing support ensuring safe and on-time communication.

2.8.1 Software Architecture and Mobility

Software architecture has in the last decade become an important subfield within the software engineering field. This has resulted in a technological and methodological base to treat software architecture as an engineering discipline. The change of focus from the development of monolithic and closed systems, to highly distributed, network-centric and open systems has imposed many new architectural challenges. [Gar00] defines the software architecture as *the gross organisation or structure of a collection of interacting components of a system*. The structure shows the high-level design decisions, like how the system is composed of interactive parts, the main interaction pathways, and the key properties of the parts. The architecture and design play a role as a bridge between requirements and implementation.

The architecture is important to ensure that non-functional requirements/properties like performance, scalability, reliability, portability, security, safety, and interoperability are satisfied (also called quality attributes [Bos00]). This has led to the recognition that the software architecture is a critical success factor for implementing software systems. Explicit architectural choices and reuse of existing architectural designs in creation of new software products have become very valuable. The principles, methods, and practices to codify the architecture, have led to repeatable processes of architectural design, criteria for making principled tradeoffs among architectures, and standards for documenting, reviewing, and implementing architectures [Gar00].

In this thesis, the software architecture plays a vital role to provide software that supports a distributed and mobile workforce in context-aware environments. Wireless and mobile computing imposes a set of challenges to fulfil non-functional requirements, i.e., the quality attributes. Many of the quality attributes are directly affected by changes in the quality of service (QoS) of wireless networks, and device capabilities. One of the main challenges is to provide an architecture that can handle clients that from time to time will be disconnected or have very weak network connections. The unpredictability of the on/off connectivity of mobile devices is a particularly hard problem, because handling the reconfiguration dynamically while guaranteeing uninterrupted processing raises big challenges

[Gar00].

Another main architectural challenge in mobile computing systems, compared to ordinary distributed systems, is to handle a large number of clients. The ubiquity of portable devices and wireless sensor networks will dramatically increase the number of nodes that may be connected to a network.

Another issue is the rich variety of heterogeneous devices that will populate the ubiquitous computing environment. The mobile computing field will probably not have the same variety, but also in this environment, the devices will have different capabilities in computing power, network connectivity, physical resources like memory and battery power, and computing environment (e.g., operating system). These issues impose special considerations in the software architecture(s). An example is to have an architecture that allows us to modify the fidelity of computation based on the power reserves at its disposal [Gar00].

Physical mobility also raises challenges to the software architecture. The current lack of automatic reconfiguration of the computing environment when moving from one environment to another, leads to manually configuration and explicit management by the user. This kind of problem is similar to handover in the cellular networks between base stations.

In mobile systems with context-aware applications, the software architecture is likely to be very dynamic. This means that the current software architecture may be unknown because new services and components can become a part of the software architecture, and leave again when outside reach of the mobile device. This raises challenges for how to provide semantics in mobile applications to identify and integrate the components that are likely to provide important functionality for the actual applications. The applications must be reconfigurable to handle the dynamic change of available components when moving in space.

2.8.2 Research Challenges to Mobility in Software Engineering

This section presents some of the research trends and challenges in software engineering that are applicable to this thesis.

[NE00] describes the need for better modelling and analysis of problem domains, as opposed to the behaviour of software. This is particularly important for many mobile users as these often will use mobile systems to perform only parts of the work. The environment which the users operate within may directly influence the capabilities to perform work on mobile devices. The problem domain will often be dynamic, requiring adaptation of the supporting systems. It is also a need to develop richer models for capturing and analysing non-functional requirements. Non-functional aspects influence mobile systems in a larger

degree than the wired counterpart. The communication, mobility, and portability properties restrict the capabilities of the devices with respect to e.g. the connectivity, size, and power consumption.

[Gar00] describes research challenges of creating software architectures that support dynamic coalitions of software services. The technical constraints in mobile devices like limited storage, memory, and processing capabilities make this need even greater than in the fixed counterparts. Mobile devices are not suitable for storing a large number and variety of software packages. The concept of service adaptation based on the user context and preferences, implies that the computing environment should both adapt to the user, as well as provide tailored, personalised services to the users.

Software architectures should also be able to adapt themselves to their physical settings. As discussed above, services on mobile devices, as well as the supporting services in the network, should adapt to the user context. In addition to the physical context, other types of contexts might influence the software architecture. Section 2.5 presents context and context-awareness in more depth. [Gar00] also mentions the need for design principles for making architectural tradeoffs between correctness, resource consumption, and reliability. The inherent physical constraints on mobile devices and wireless networks increase the need to provide software architectures that are able to handle changing computing environments. The design principles can be used to create self-monitoring systems that are able to adapt to the changing computing environments taking the described tradeoffs into account.

2.9 Summary

In this thesis, we extend the reach of traditionally work support systems to also include mobile, ubiquitous, and context-aware computing. Context-aware and adaptive systems are of special interest since these areas covers important properties of a support system for smart work processes. In addition, we believe that middleware especially developed for mobile ad hoc environments will play an important role to collect and distribute information between all the different actors that may participate in a ubiquitous, sensor-rich computing environment. Mobility leads to changing circumstances, and thus to a partly unknown dynamic environment of actors, sensors, work processes, and other activities that can influence the mobile worker.

This chapter has presented state-of-the-art in many different areas, where the support for mobility has been the common denominator. The properties of the mobile devices, the supporting networks, and the usage patterns of the mobile user are the main factors that differentiate mobile systems from distributed systems. The extension of distributed services to also support mobility makes new and often persistent challenges to software developers. As the desktop metaphor is not usable for small mobile devices, this chal-

lenges the way both how to interact with the devices as well as how to build applications and services that automate the process of using them. Context-awareness and the vision of ubiquitous computing are driving forces for the future computational and communicational services for the mobile user, both professionally and privately.

Different approaches of systems that support work in different ways have also been addressed. Common for all of these approaches are that they handle tasks or activities, and includes actors like systems and/or people that play different roles by having certain skills and/or responsibilities. Artefacts may in all be created, transformed, or shared. The artefacts represent objects of work within an environment, and correspond to, e.g., documents, pictures, plans, video, and other multimedia objects. The different actors perform work in an environment where activities may come from plans, presence of resources, or need for coordination and/or collaboration. Different aspects of how to support work are however more visible in some of the technologies. CSCW focuses on collaboration and the social aspects related to it. CSCW is therefore more directed towards user interfaces and synchronous communication, while PSEE/PCE is focused on process models and environments to support/perform processes. Workflow systems might be looked upon as a combination of both.

The mobile dimension changes the working environment by introducing new sources of context that might influence how work is performed and thereby how work should be supported by computers. The need to support work performed on a diversity of devices, asynchronously as well as synchronously, connected as well as disconnected, planned as well as ad-hoc, context-aware as well as context-unaware (see Section 2.5), and individual as well as cooperative, adds many challenges that need to be addressed to some extent. The assumption of a stable infrastructure with unrestricted access to networks, powerful computers, and a fixed environment, is changed by mobility to a partly unpredictable and very dynamic environment where the way software is architected gives new and different challenges in supporting the mobile work processes.

CHAPTER 3

Research Focus and Methods

This chapter describes the research focus and method of this thesis.

3.1 Research Focus

The research focus and contributions of this thesis are divided into three themes:

T1: Exploration and investigation of mobile work

T1 has been realised by different types of user studies to find, explore, and understand requirements for computational support for different user groups. We have further asked for which experience users have from the use of mobile work support applications. The user studies are presented in paper P4 and AP5 (in Appendix A). The other papers present specific user scenarios that contribute to our understanding of mobile work.

T2: Investigation and development of concepts and frameworks to capture and describe properties of mobile work processes

T2 has resulted in development of a process characterisation framework for mobile work. The framework is intended to support the development of process support architectures and infrastructures through investigation of different properties in work activities to es-

tablish a set of requirements, both functional and non-functional. Paper P1, P3, and P8 present the MOWAHS characterisation framework including applications of it. In addition, a tool has been developed to support the characterisation. Paper P9 proposes a new research field for supporting mobile work through the use of context information. We have partly explored different scenarios for where and how context can play an important role for workflow enactment. A coarse-grained model of an architecture has been proposed to support smart work processes.

T3 Development of enabling technologies to support "smart" work processes in pervasive and mobile, ad hoc computing environments

We have developed applications and prototypes to explore and evaluate different technologies that can be utilised in support of mobile work processes. Paper P2 presents experiences and an evaluation of an implemented mobile work support system. This system addresses challenges related to content and activity adaptation to different hardware platforms as well as usability issues. Paper P5 and P6 present a middleware especially developed for the target environment envisioned for smart work processes; mobile ad hoc wireless environments. Paper P10 presents a framework developed to support development of location-aware applications, and

3.2 Research Methods

This section presents relevant research methods applied in software engineering research.

The goal of research within the software engineering discipline is to enable software to be produced more effectively, with better quality, according to plan, by spending less resources. This can be done by making better models, methods, and tools. Software engineering is a multi-disciplinary research field that includes both technical and non-technical research. Technical research includes development of various tools, models, and languages. Non-technical organisational research includes issues related to how to manage people, and how to change development processes and organisations.

Research approaches in software engineering are typically *descriptive*, *evaluative*, or *formulative*. The descriptive approach includes describing systems and review of the literature. The evaluative approach may, e.g., be deductive, interpretive, or critical. The formulative approach may include frameworks, guidelines/standards, models, processes/methods/algorithms, classifications/taxonomies, or concepts. With such a toolbox of research methods, we can employ different approaches based on the purpose and apparatus of the study.

[ZW98] defines four general categories of research approaches: the *engineering*, *empirical*, *scientific*, and *analytical* method. The engineering and empirical method can be

seen as variations of the scientific method [Bas92]. The analytical method can be seen as equivalent with the mathematical method. Typically, several research methods are often used together at the same time. [ZW98] describes a taxonomy for software engineering experimentation based on various examples of technology validation. 12 different experimental approaches are grouped into three broad categories: Observational, historical, and controlled methods.

Observational methods:
Collects relevant data as a project develops, and is characterised with little control over the development process other than using the new technology being studied. The methods are divided into four types: *1. Project Monitoring*, *2. Case Study*, *3. Assertion*, and *4. Field Study*.

Historical methods:
Collects existing data in completed projects. The methods are divided into four types: *5. Literature search*, *6. Legacy data*, *7. Lessons learned*, and *8. Static analysis*.

Controlled methods:
If sufficient instances of an observation are available, these research methods provide statistical validity of the results. Controlled research methods are used in experimental design in other scientific disciplines. The methods are divided into four types: *9. Replicated experiment*, *10. Synthetic environment experiments*, *11. Dynamic analysis*, and *12. Simulation*.

[GVR02] presents research methods typically used within software engineering. Based on a study of papers in the most important software engineering journals, the most used methods are:

Conceptual analysis – 45.5%, *Concept implementation (proof of concept)– 17.1%*,
Conceptual analysis/mathematical – 10.5%,
Laboratory experiments (human subjects) – 3%,
Case study – 2.2%,
Data analysis – 2.2%,
Descriptive/exploratory surveys – 1.6%,
Literature review – 1.1%, and
Simulation – 1.1%.

Empirical studies in software engineering have in the last decade gained recognition within the research community. Especially has the human role of software engineering been addressed in studies based on new types of questions and methods. Non-technical issues and the intersection between technical and non-technical software engineering have been addressed in recent studies [Sea99, CW03].

The validation of research results is a fundamental problem in all research. Correctly col-

lected data does not automatically mean that the results are generally valid. The *internal validity* of an experiment indicates that research results are valid within the organisation experiments were designed for. *External validity* means that the result is applicable also outside the scope of the experiment. Software engineering experiments seldom get results that are externally valid. Many factors may affect the result such as organisation size, organisation type, product type, or the organisation of the development process etc. In our work, we have mostly worked with singular applications and partly validated the results as proof-of-concept.

3.3 Research Methods used in this Thesis

This section describes the research methods used both with respect to the different research questions, and to the included papers.

The research methods used in this thesis have been different from paper to paper, but the main emphasis has been on conceptual analysis and concept implementation, i.e., the engineering method, but we have also conducted some limited empirical studies.

3.3.1 Research Questions and Methods

Below, we indicate which research methods that have been used to address the different research questions. The research question will be stated before the relevant methods are presented.

RQ1: What are the special properties of mobile work and how do these influence how to build supporting technology?

The first part of this research question is exploratory, thus case studies, descriptive/explorative surveys, and literature review have been the dominant methods used to answer this question. Based on the findings in the first part, we have partly addressed the second part of the research question by investigating existing technologies, developing prototypes to address different properties of such technology, and developing new concepts to describe how mobile work can be supported.

RQ1 is divided into two sub-questions:
RQ1.1: *How can concepts and frameworks be developed and used* to understand and specify *how mobile work processes can be supported*?
RQ1.2: *How can such concepts and frameworks be used to support the development of mobile work support systems*?

RQ1.1 addresses tools and concepts for developing mobile work applications, while RQ1.2

specifically addresses the development process of such mobile work applications.

RQ1.1 addresses work that can use conceptual analysis and concept implementation. Firstly, we established an initial framework based on experiences from developing mobile work support applications and from knowledge of how such technologies have been used on more general mobile activities. Secondly, we specified some relevant work processes for different user groups to test out the framework in practice by letting different people characterise the same process. Based on experiences from this activity, we adjusted both the framework itself and the characterisation process to be able to make them more precise and unanimous. Further, we found the need to introduce new concepts to more clearly distinguish nomadic work from true mobile work, i.e., to extract which properties are common and which are different. The conceptual analysis and implementation of a framework thus lead to definitions of new concepts opening for new analysis and implementation of the framework.

For RQ1.2, we have employed the *engineering method* to create prototypes to explore different technologies to support mobile work processes, and thus get experience on how such applications can be built. We have then used the frameworks and concepts based on RQ1.1 to identify and capture requirements related to both functional and non-functional properties of supporting applications. Such requirements directly influence the architectural and design process. The *assertion method* has been used to compare our framework with similar frameworks.

RQ2: How can work process and context information be modelled and utilised in mobile, heterogeneous, and dynamic work environments?

This research question uses *exploratory research* to understand and specify properties relevant for the mixture of context-awareness and more traditional work process support systems like workflow. We stated a hypothesis that workflow systems can be extended to also support context-awareness without requiring fundamental changes in the basic workflow server architecture. This hypothesis has been tested by using the *engineering method* to create prototypes for client-based context-reasoning and workflow enactment and using the already defined interfaces in the workflow reference model to communicate with an ”ideal” workflow server.

RQ3: How and which technologies can be used to create an infrastructure to provide support of mobile work processes?

This research question is partly answered using the engineering method to build prototypes using different technologies to support mobile work processes. The prototypes are partly vertical and have been addressing different parts of an architecture to support such processes. *Literature review*, *conceptual analysis, and concept implementation* are thus the most used research methods. In addition, we have employed the *assertive method* to

compare our architecture to similar work.

3.3.2 Papers and Research Methods

Table 1.1 shows how the different papers are mapped to the research questions. We will, however, summarise the main research methods used in each paper to detail how our research has been conducted. Table 3.1 shows how the different papers map to research method(s).

Paper	**Main method**	**Approach**
P1	Engineering	Conceptual analysis, case study, assertive/literature review
P2	Engineering	Concept implementation, case study
P3	Engineering	Conceptual analysis, case study
P4	Descriptive/exploratory survey	Case study
P5	Engineering	Concept implementation, assertive/literature review
P6	Engineering	Concept implementation, assertive/literature review
P7	Engineering	Concept implementation, assertive/literature review
P8	Engineering	Concept implementation, conceptual analysis
P9	Formulative	Conceptual analysis, assertive/literature review
P10	Engineering	Concept implementation, conceptual analysis, assertive/literature review

Table 3.1: Papers vs. Research Methods

CHAPTER 4

Main Contribution: Support of Smart Work Processes

This chapter will present the contributions of this thesis and in more detail the proposed architecture for a smart work process application. Section 4.1 presents the background of the thesis, while Section 4.2 gives a motivating scenario for our approach. Section 4.3 outlines a system to support mobile work. An architecture of such a system is presented in Section 4.4. The enabling technology papers are here put into the context of the architecture. Further, Section 4.5 briefly discusses the prototypes developed; Section 4.6 presents the paper abstracts; and finally, Section 4.7 summaries this chapter.

4.1 Background of this Thesis

The problem areas covered in this thesis are mainly related to exploration and development of systems intended to support mobile workers. It is therefore important to understand which properties make mobile work special to distinguish mobile work from other types of work situations. Mobile work shares many properties with stationary and distributed work, but is driven by a motivation and need for changing the working context, especially the location.

Mobile work differs from stationary work by (possibly rapid) changes of the work environment, and thus implicitly the contextual state of the work environment. These contextual changes might influence the work environment itself, the work processes performed within it and the need and wish to perform work. This makes context information an im-

portant catalyst and source of information for the mobile work processes. Many aspects of being mobile must be taken into account when creating systems to support mobile work.

Few of the traditional work support systems have the necessary properties to support the dynamic environments envisioned by ubiquitous computing. There is therefore a need to investigate how such systems should be developed to exploit the new possibilities imposed by ubiquitous computing environments. Especially when mobile, a smart work support system can have a great merit towards making work more efficient and safe. Systems to support mobility have to take into account several considerations and restrictions as described in Chapter 2.

4.2 Motivating Scenario

Many work places have a complex structure of participants, activities, and artefacts that may make even simple work processes very hard to plan and execute. Such environments are dynamic and unpredictable. Resources can be scarce and different participants may therefore have to compete to make necessary adaptations in the environment to fit their own goals. Often in such environments, *safety* is a very important issue for the employers and other actors in the society. Avoiding personnel injuries and damage of expensive equipment are important to ensure continuous operation. Places that possess such properties are, e.g., oil platforms, ship yards, roads, hospitals, and construction sites.

Safety for single actors can be ensured by establishing safe working conditions to perform activities within, or to re-establish safe conditions by performing situated or planned activities.

In environments with multiple actors, activities, and a constantly changing working environment, the process enactment can be very complex. The actors can cooperate to perform tasks or work in parallel on different tasks to reach a common process goal. The actors may also work independently with different process goals and may therefore not be aware of the other actors' processes and goals. The actors influence a working environment in different ways that can create hazards for other actors as well as divert resources from others. By providing context information to the individual and to supervisory process enactment services, it is possible to initiate coordination activities to establish safer and more economic work conditions.

Context-aware and adaptive work processes can be used as a means to coordinate multiple actors. Reactiveness to environmental changes is important to ensure an optimal safety and production rate. Such changes can be captured as context information and provide input to situated planning and actions in the environment.

In environments that are self-aware, i.e. equipped with augmented, "intelligent" artefacts [SGKK04], activities can be initiated by the environment to ensure certain environmental

goals. In addition to provide activities, such artefacts can help in coordination of actors. The coordination can help to maintain or enhance productivity, safety, or other goals defined by the environment, by the actors, or by their work processes.

4.2.1 Smart Work Process for an Electrician: A Scenario

Electricians are a good example of a profession performing mobile work. To elaborate our approach, we present the following futuristic scenario of an electrician with the process goal of installing an electric wall heater in a house. The process can be divided into the following activities where the environment also is performing activities to support the process goal.

1. The electrician company gets a work order to install a heater at a given address.
2. A time slot is calculated for the work based on available personnel. The work order is put into the work plan of an electrician based on the following properties: Skill and minimum driving distance.
3. Necessary equipment is made available to pick up by the electrician.
4. A driving route is calculated for the electrician based on the today's activities and presented to the electrician in the car together with information about the next activity.
5. The system finds historically information about the house to be visited and the relevant information is presented by audio when the electrician is driving to the actual address.
6. The owner of the house indicates the position of the installation to the electrician.
7. The embedded in-house computing environment (system) presents a map of the available circuits, switch boxes and other information related to the electricity infrastructure in the house. All cables and other equipment are instrumented with location sensors and an internal knowledge database with technical information about themselves and their connections. The system can deduce the map, power usage and other information based on the locations and the technical information in the equipment.
8. The map is interactive and the position and technical information of the heater is given to an in-house system.
9. The in-house system presents an overview of already installed equipment together with the already used effect on the nearby, relevant circuits, and proposes to either use an existing circuit, or to install a new one.

10. The electrician takes a decision to either use an existing circuit, or to install a new one, and informs the in-house system about the decision.

    - If installing a new circuit with fuse, a conduit route is proposed by the in-house system based on given preferences from the fuse box to the heater position.
    - If using an existing circuit, the nearest switch box is shown and a proposed route of the conduit is presented.

11. The system turns off the part of the electricity system affected by the decision in activity 10.

12. The electrician installs the conduits, cables, and other necessary equipment and the heater itself.

13. The electrician informs the system that the work is finished.

14. The in-house system updates the circuit map and includes technical information about the installed equipment.

15. The in-house system informs whether the new circuit is working or not.

16. The in-house system ensures that it safe to turn on the power, and turns it on.

17. The newly installed equipment gives the technical information back to the electrician's computer and an invoice is created based on the used material, time spent and the driving distance and time. The inventory is updated, and the invoice is either transmitted electronically to the in-house invoice receiver service, or printed out.

18. The invoice is paid by the customer and the electrician leaves the house eventually continuing from activity 4.

This scenario illustrates many cases where context is meaningful and necessary to drive the scenario further. We have not included any exceptions in this scenario, but, e.g., overriding the power turn-off could create a dangerous situation for the electrician. Likewise, the decision in activity 10 to use an existing circuit can result in hazards if the circuit is overloaded by installing the heater without moving/removing other equipment to other circuits.

The location in the scenario has several roles. Firstly, it is used geographically to find addresses and routes to those addresses; secondly, it is used to topologically connect electrical equipment by identifying which objects are connected to which other objects.

4.3 A System to Support Mobile Work

Work with other actors in a mobile environment may introduce a quite complex and fluctuating organisational structure. Workers that are aware of other actors' activities and goals can partly cooperate with these. If unaware, they work independently. Coordination of activities can therefore be divided into at least three types: *explicit*, *implicit*, and *no coordination*.

There are situations and environments where coordination can be implicit, based on explicit or implicit rules and practices. In road traffic, coordination and cooperation are implicit and not related to the primary goals of the different actors (e.g., driving from A to B). Implicit coordination is based on secondary or inherent goals like, e.g., to ensure safety of people, vehicle, and other entities that participate in such an environment; adhere to regulations and rules to avoid fines, etc. Likewise, secondary goals are likely to be present in all work environments, requiring implicit or explicit coordination. A typical goal in most work places is to ensure the safety of people, and avoid damages on artefacts or on the environment itself.

4.3.1 Mobile Work Context

The mobile work context may consist of many elements that influence what activities to perform, and how and when to perform them. First of all, most work is specified by plans, or is covered by job instructions, stated responsibilities, or through a professional role. These plans may be tentative, or act as templates for the worker. In addition, there are appointment schedules, to-do lists, and workflow tools partly used by mobile workers to manage and coordinate the work activities.

Activity theory [Vyg78, Nar96a, EMP99] has been used to describe how process goals can be achieved through activities, realised by actions, and concrete and atomic operations. The actions and operations do not necessarily see the goal of themselves. Context is here described as everything related to an activity, including all participating actors, artefacts and the activity itself.

For many mobile workers the location is important, but may play different roles related to the work situation. A location may contain important resources (artefacts, people, etc), or the problem to be handled/solved. We will call the motivation for mobility *an object of interest*. The location of an object of interest can be:

1. **Fixed**, i.e., we have to move to reach the object.
2. **Coinciding with our own**, i.e., we happen to be at the same location as the object.
3. **Moving**, i.e., we move together with the object, or the object moves together with us.
4. **Distributed**, i.e., we have to work towards several locations at once.

The mobile work groups studied in this thesis – home nursing care in paper P4, electricians in paper AP4, and IT Support in paper P2 and AP4 (in Appendix A) mostly belong to category 1 with respect to location. They all have to move to the object of interest to perform their work activities.

Another important work context is *time*. The time property contains elements like deadlines, and availability of people and artefacts including our own availability and artefacts.

The physical environment, in which activities are performed within, may have a great impact on how to manage the work activities. Typical physical properties like weather, temperature, pressure, air quality, altitude, location, action, and state of objects and artefacts, etc., may introduce or pose conditions that affect how, when, with whom, and if to perform activities. Other context information that might influence the work activities are the personal context; e.g., physical and psychological situation, skills, and expertise; and the social context (who are present and how do we communicate with them).

Personal or activity preferences may be used to change the environmental state to fulfil wishes or needs. Activity preferences are especially important since such preferences may be regarded as necessary preconditions to be able to perform the activity. Electronical or physical actuators can be employed to change the state of the environment, or separate, situated actions are necessary by one or several actors to meet the preconditions.

In all dynamic environments, many un-planned situations may occur. The situational information of dynamic environments requires actors to be able to both plan and perform activities in an ad hoc manner. To support automated work processes in such environments, we thus need to be able to sense and reason about the state and subsequently propose and support actions to meet stated process or, e.g., safety goals.

In addition to the context sources described above, mobility and the possible presence of electronic devices introduce context sources that are related to the execution environment of software, hardware, and (wireless) networks. Availability of tools, networks, applications, and services including the state of these, are important to include in the reasoning about the work environment since the conditions and outcome of a process might heavily be influenced.

4.3.2 Context-Aware Work

In paper P9, we discuss the implications context information may have on work processes. It is especially interesting to look at mobile work processes since these most often require situated support. Mobile work processes are, compared to nomadic or stationary work processes, much more dependent on up-to-date context information. For the present mobile workers, most of the important context information is captured through human sensing, and this information is reasoned upon by using their knowledge and experience.

Future mobile workers may experience an infrastructure capable of communicating relevant context information tailored to their plans, activities, and goals. Cooperation can be achieved in the environments through separate services implemented to enhance productivity and safety. In the long run, such experiences and technology can give opportunities to automate manual work using robots capable of performing complex work operations using knowledge, process enactment, context capturing, and reasoning.

4.3.3 Requirements

We have used the framework and experiences reported in the papers P1, P3, and P8 to explore requirements to mobile work support systems. The characterisations have been used to develop a generic architecture that is presented in Section 4.4. Further, paper P9 presents requirements and issues related to support of context-aware, mobile work processes. Paper P9 also presents the generic architecture shown in Figure 4.1 and specifies some of the functionality each component should offer.

4.4 Smart Work Process Architecture

This section presents the architecture included in paper P9, and discusses how the technology presented in the *enabling technology* papers can be integrated into a solution for support of smart work processes. The architecture is generic in the sense that it should support multiple user scenarios.

The high-level architecture in Figure 4.1 shows the main parts of the architecture and how the different papers relate to each part. Note that the technological papers demonstrate possible enabling technologies to implement parts of the architecture. None of technology presented in the papers were intended to give direct support for the architecture. The architecture came as a result from the experiences gained in work related to the papers.

Most of the enabling technology papers are not attached to a single component in the architecture. The middleware can for instance be used as part of the supporting infrastructure when implementing the architecture.

The architecture consists of seven main parts which will be described in more detail below.

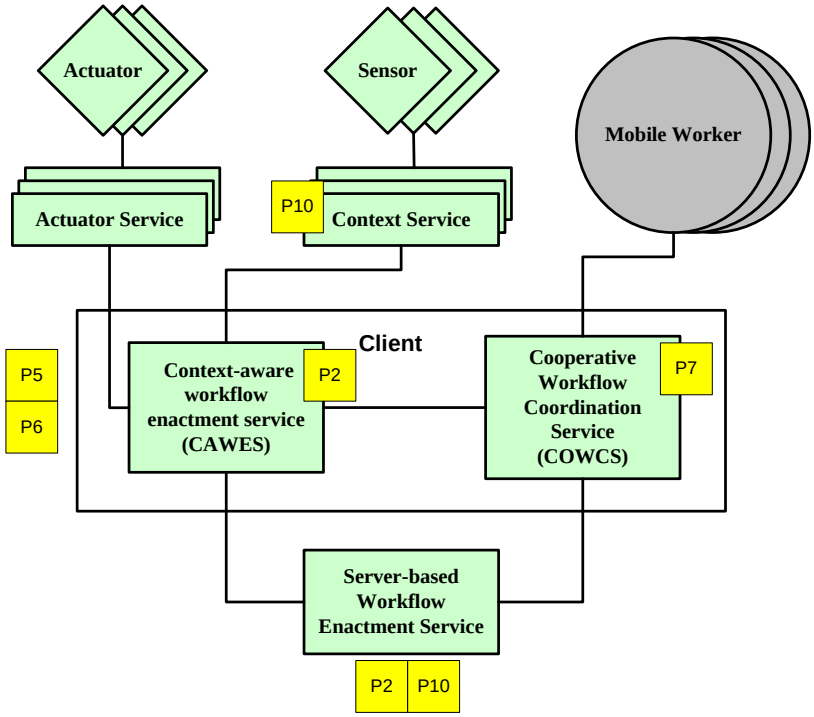

Figure 4.1: The Smart Work Process Support Architecture

Actuator:

An actuator is an entity that is able to change the environment based on digital input from some digital equipment. In our architecture, an actuator is responsible for acting as specified by an actuator service.

Sensor:

A sensor is responsible for measuring and transmitting sensor readings specified by a context service. Sensors might vary from very simple, to autonomous, augmented entities.

Actuator Service:

An actuator service is responsible for initiating actuations in the environment. The service should be able to receive instructions from the workflow enactment service and translate these to the actual actuator. The actuators can in this service be represented as abstract entities with published properties. Actuator services hide the actual communication with

the different actuators and provide a well-defined interface to be used by clients of the service. The actuator service should be able to communicate with actuators through various physical communication implementations like IR, WLAN, LAN, Bluetooth, etc.

Context Service:

A context service is responsible for identifying sensors, initiating sensor readings, checking sensors, setting up/terminating sensor subscriptions, etc., i.e., to provide communication to all actual sensors. The context service can accept subscriptions from different clients and is then responsible for setting up the sensor subscription properties. The clients can specify which properties they prefer to receive.

Based on properties received from the various clients, the context service should send context messages to the client subscribers related to subscriber preferences. A context service may be extended to also represent, e.g., augmented artefacts. It should then be able to communicate process fragments based on templates initiated by certain contextual states/conditions either contained within the context service, or received from smart, autonomous sensors. Possible context services that can be provided are, e.g., weather conditions, engine and application states (including the state of the client itself), location, or the mobility of the user and other objects/entities within the environment.

Client-Based, Context-Aware Workflow Enactment Service (CAWES):

CAWES is responsible for monitoring and executing delegated activities through surveillance of the current process/activity state, and contextual events received either from the context service, the COWCS (see below), or the Server-based Workflow Enactment Service. CAWES can also receive process/activity fragments from the context service, the COWCS, and the Server-based Workflow Enactment Service.

CAWES can send its own process/activity state as contextual events to the COWCS, and the Server-based Workflow Enactment Service. If there is a need for actuations in the environment, CAWES initiates actuations using the Actuator service.

Further, it can display/inform about environmental state with respect to the process itself, safety state etc. It can also display/inform about information related to the activity performed. Such information can be based on the activity itself (manuals, checklists, or multimedia information), location-based information, safety information, etc.

In addition, CAWES should provide a traditional workflow user interface for filling in information, create/update artefacts related to the activity, e.g., interactive checklists (can also be filled automatically using the context service), manage/integrate audio/video/textual information collected through the device or an actuator service) etc.

An extension of CAWES is to provide virtual/augmented, location-based information

(manuals, multimedia) related to the working environment, process type, available tools, and sensors.

Client-Based, Cooperative Workflow Coordination Service (COWCS):

COWCS is responsible for managing and coordinating activities in a multi-actor environment. This includes a coordination service that, based on policies, can send messages to all involved parties about coordination needs, competitive resources, etc.

COWCS receives process goals and currently performed activities from other actors. Similarly, it can send its current state of executing activities and possibly prepare other users about new activities to be started within a time limit. It is also responsible to send coordination messages to other users including needs for artefacts, services, actuators, and a preferred environmental state.

Internally within the client, COWCS sends contextual events to CAWES based on coordination reasoning. Such events may include information about the need for artefacts, services, actuators and an environmental state with priorities, time, demands, etc. It can also send new process fragments derived from the coordination service or received from other clients based on demand/need from other users. The fragments are sent both to the CAWES, and to the Server-based Workflow Enactment Service.

Server-based Workflow Enactment Service:

This service is responsible for traditional workflow management including sending activities to the users based on delegation/plans, and managing the complete, high-level process for the involved participants.

4.4.1 Middleware for Smart Work Process Architecture

We have included two papers (P5 and P6) that cover context-aware middleware for mobile ad hoc networks (MANET). The different client-based components are supposed to work in MANET environments with no fixed infrastructure. We have partly implemented a smart work process prototype that is using the CORTEX middleware in communication between artificial sensors and the executing client, and between different clients communicating process state and plans (not reported in detail in this thesis). The proposed middleware can thus be used for all communication between the different components and actors within the architecture by representing the different components as, e.g., sentient objects.

4.4.2 Supporting Heterogeneous Devices: The MONGO System

A lot of practical applications have been developed to present adaptive information based on client capabilities. Especially Web browsing has had attention among researchers (e.g., ARtour Web-Express [CTC+97], Power browser [BGMPW00], and WEST [BHR+99]).

The focus in these papers has been to present methods for light-weight browsing that does not require heavy computations and communication. We have included the paper P2 (and partly P8) that presents practical experiences of a mobile work support system. The MONGO system adapts process contents to different types of devices. Our work shows some considerations that must be taken into account to implement usable mobile work systems on mobile devices. The prototype is developed specifically for IT-support, but can be modified to support similar professions. The *server-based workflow enactment service* and *CAWES* in Figure 4.1 can be made aware of client capabilities, and can thus tailor the relevant information and services to these capabilities.

The MONGO system was developed for IT support work processes. This system has later been extended with location-awareness to make it able to download tasks and information based on the current location of the worker.

4.4.3 Location-Aware Support: The Nidaros Framework

Paper P10 (and partly P8) present location-aware applications using the location to adapt the content to be received by the client. P10 describes the Nidaros framework developed to support location-specific multimedia presentations. The position is used to look up relevant information for that location and the system transmits this information to heterogeneous devices like PDAs and mobile phones. Figure 4.1 shows how paper P10 can be mapped into the architecture by, e.g., using the Nidaros framework in specific *context services* to deploy location-specific information to the client. Similarly, the *server-based workflow enactment service* can be extended to support location-specific workflow services and information to the workflow client. Similar applications are, e.g., CyberGuide [LAAA96, AAH+97], the Lancaster GUIDE [CDMF00], Interactive museum guides [FFK+02], and MUSE [GCP03].

4.4.4 Collaboration: Peer-To-Peer

Paper P7 presents technology that can be used to enhance collaboration in mobile environments. In our architecture, the main focus is on how to best support mobile work, thus, sharing of resources and cooperation in actual work processes are the most interesting issues to be addressed in a P2P environment.

[DMPD99] describes a workflow architecture to manage mobile collaborative work. The motivation of this architecture is to support group-oriented collaborative activities that scale beyond a conventional stationary organisation. Their architecture considers requirements like support for flexibility, and promoting collaboration-aware resource sharing preserving adequate levels of autonomy for individual mobile workers. The architecture should give support for resource sharing facilities, participation, and awareness control about the tasks performed by workers in a group-oriented perspective. Finally, such an architecture should include coordination support by means of an adequate workflow management model specially tailored for flexible task assignment and scheduling for mobile and intermittent connection environments.

COWCS in Figure 4.1 must take into consideration P2P issues as well as coordination and cooperation. COWCS gives context information to the workflow enactment service (CAWES) based on input and negotiation between peers in a mobile work environment.

4.4.5 Process Models

We have in this thesis not covered in detail how to create process models suitable for mobile work. We have partly based our work on the standards proposed by WfMC [The05]. Further, the MOWAHS project has partly been based on the CAGIS project [pro00] that worked with process models. The process models used in paper P2 were mainly individual and thus did not require any specific coordination mechanism.

4.5 Implementation

In addition to the systems described in the attached papers, we have developed several prototypes investigating different properties of context-aware work support systems. These prototypes have been based on the WfMC reference architecture as shown in Figure 2.3 and described in Section 2.6.2. We will not go into further details of these prototypes since they are not included in the paper parts of this thesis.

4.6 Paper Abstracts

This section presents abstracts of all the papers contributing to this thesis. The papers have been published at international conferences and workshops, and have been fully reviewed by peers. The papers have been divided into the three sections *survey* papers, *conceptual and framework* papers, and *enabling technology* papers. The papers have been grouped

into these three themes based on the research methods used and how they contribute to the research questions presented.

4.6.1 Survey Papers

This section presents a paper abstract of a survey of the home nursing care.

P4: A Survey of Mobile Support Needs in the Home Nursing Care

This paper describes the results of a survey performed in the home nursing care at different municipalities and health regions in Norway. The purpose of the survey was to map which kind of technical or informational mobile support the personnel reckoned as important when visiting patients at their homes or elsewhere when mobile. The sample was drawn based on an index that groups the regions based on the size of the population. The people in the home nursing care are familiar with the use of mobile phones and computers. They also demands access to certain type of information when mobile to support their daily work.

Relevance to thesis: This work gives insight into how the mobile work force looks at using mobile technology in their daily work. Further, it gives information about which kind of information and services that are valued as most important when mobile.
My contribution: I was advisor to the survey and leading author of the paper.
Where in thesis: Chapter 8, page 83.

4.6.2 Conceptual and Framework Papers

This section presents paper abstracts of papers that outline concepts and tools related to support for mobile work.

P1: The MOWAHS Characterisation Framework for Mobile Work

This paper describes a framework used to characterise mobile work in order to elicit functional and non-functional requirements for a mobile process support system. The framework is a tool for specifying and analysing mobile scenarios in detail, resulting in a characterisation of scenarios. This characterisation will indicate requirements to the software architecture and services the system should provide. In addition, the framework will indicate non-functional requirements like network capacity, network connectivity, security. To show the practical usage of the framework, we have applied the framework to a scenario describing a mobile researcher. As far as we know, there are no similar frameworks.

Relevance to thesis: The framework provides a tool for capturing requirements for mobile systems. Especially, discovery of non-functional requirements in mobile systems is vital to create architectures that are able to give the mobile user reasonable support for the most important tasks.
My contribution: This paper is a result of group work in the MOWAHS project. I was the leading author of this paper.
Where in thesis: Chapter 9, page 101.

P3: Requirement Indicators Derived from a Mobile Characterisation Framework

This paper describes eight requirement indicators derived from a mobile work characterisation framework (MWCF). The MWCF is a framework for analysing mobile scenarios (mobile tasks) in order to implement a system to support such scenarios. The requirement indicators are computed from the score of the MWCF. These are used to reveal the complexity of the different parts of a final mobile support system (software and hardware). Further, these indicators can be a help to prioritise the non-functional and functional requirements of the end-system. We have identified the following indicators: General Task Indicator, Information Complexity Indicator, Location Complexity Indicator, Time Complexity Indicator, Network Connectivity Indicator, Network Speed Indicator, Energy Consumption Indicator, Transactional Support Indicator and Mobility Indicator. In this paper, we apply these indicators on four different scenarios and discuss how they can be used in practice.

Relevance to thesis: This work is a continuation of the work on the MOWAHS characterisation framework. The contribution is identification of certain indicators that influence how to identify difficult or challenging architectural problems for mobile systems.
My contribution: This paper is a result of group work in the MOWAHS project. I participated in fine-tuning of the basic framework and in creating an applicable process for using the framework. Further, I participated in using the framework in different scenarios. Finally, I wrote the main parts of the related work sections.
Where in thesis: Chapter 10, page 115.

P8: Using the MOWAHS Characterisation Framework for Development of Mobile Work Applications

This paper describes an evaluation of a characterisation framework used to analyse mobile work scenarios to make software systems to support these scenarios. The framework identifies complexity issues of the systems to be taken into account when implementing it. The framework can also be used to elicit requirements from the scenario. Three research questions are investigated in this evaluation: 1) Can the framework be used to identify relevant challenges in the final system, 2) Can the framework be used to identify func-

tional requirements for the final system, and 3) Can the framework be used to identify non-functional requirements for the final system.

The evaluation of the framework was performed using the framework to analyse and implement an IT-support scenario.

The paper also describes a web-tool for this framework that makes the characterisation process simpler. The tool introduces consistency rules to ensure more consistent characterisation of the scenarios.

Relevance to thesis: This paper presents an evaluation of the characterisation framework presented in paper P1 and P3, and a tool to support the characterisation process.
My contribution: This paper is a result of group work in the MOWAHS project. This paper is a continuation of P1 and P3. I participated in the development of the Web-tool, and in the evaluation of the research questions.
Where in thesis: Chapter 11, page 129.

P9: Support of Smart Work Processes in Context-Rich Environments

The evolution of mobile and ubiquitous technologies gives promises for computational services and resources to support and influence work processes planned or performed in physical work environments. Such support should be provided in an unobtrusive manner, and optimally provide the workers with a safer work environment for both the environment itself and the workers. The extended support can give more economic and optimal work processes through proactive and situated planning and execution. Support for situated planning and actions can be achieved through in situ defining activities to be performed, deciding when to start and stop different activities, and possibly controlling the duration of activities. This paper presents issues related to support for mobile work distinguishing the mobile use of technology from mobile work.

We introduce the concept of a *smart work process* to capture adaptive and context-aware process support. This combination of ubiquitous computing and workflow defines a new research direction to be investigated. This paper elaborates on research challenges related to how smart work processes can be supported. We present and discuss general cases where context information or change in context information influences mobile work activities. Finally, we propose a framework for modelling smart work processes, and present a high-level architecture to support smart work processes.

Relevance to thesis: This paper presents the main contributions of this thesis with concepts related to support of context-aware mobile work processes.
My contribution: This paper is the main contribution of this thesis with me as the leading author.
Where in thesis: Chapter 12, page 147.

4.6.3 Enabling Technology Papers

This section presents paper abstracts of papers that describe enabling technology to support smart work processes.

P2: Experience Paper: Migration of a Web-based System to a Mobile Work Environment

It is a large collection of software supporting the planning, execution and reporting of work activities, but these systems are mainly created to be run in a distributed wired network, or as standalone systems. The systems do not directly support the mobile part of work. The cost and effort to make such systems available to mobile users are not just an issue of making adjustments to the network protocols, but also to be able to adapt the system to a heterogeneous set of hardware spanning from small mobile phones to portable PCs. This paper presents experiences from migration of a Web-based task administration system to a mobile environment. Different solutions of system adaptation are discussed, and a prototype of a system to support mobile work is presented and evaluated.

Relevance to thesis: This work gives valuable feedback for how to create and adapt mobile work support applications based on existing technology, and discovers issues related to deployment of software in heterogeneous environments.
My contribution: I was advisor for the project, and leading author of the paper.
Where in thesis: Chapter 13, page 165.

P5: Novel Component Middleware for Building Dependable Sentient Computing Applications

With the advances in sensor-based computing and mobile communication, people have started to explore ubiquitous or pervasive computing systems that aim to have computing devices literally available everywhere, making them disappear into the physical environment. Novel ubiquitous computing applications such as intelligent vehicles, smart buildings, and traffic management have special properties that traditional computing applications do not possess, such as context-awareness, massive decentralisation, autonomous behaviour, adaptivity, proactivity, and innate collaboration. In this paper we argue that such applications require a new computational model and middleware that can reflect the autonomy and spontaneity of cooperative entities. The EU funded CORTEX[1] project proposes the sentient object model to support the construction of such large-scale applications. We report on a flexible, run-time reconfigurable component based middleware that we have used to engineer the sentient object programming paradigm. We demonstrate the

[1]The CORTEX (CO-operating Real-time senTient objects: architecture and EXperimental evaluation) project is supported by the Future and Emerging Technologies programme of the Commission of the European Union under research contract IST-FET-2000-26031, http://cortex.di.fc.ul.pt/.

appropriateness of the novel computational model and validity of the middleware by constructing a proof of concept demonstrator based on the notion of autonomous cooperating vehicles.

Relevance to thesis: This paper describes a middleware solution for context-aware, autonomous objects in ad hoc network environments.
My contribution: I participated in the CORTEX project from September 2003 to April 2004. The related work section has been partly written by me.
Where in thesis: Chapter 14, page 177.

P6: A Context-Aware Middleware for Applications in Mobile Ad Hoc Environments

Novel ubiquitous computing applications such as intelligent vehicles, smart buildings, and traffic management require special properties that traditional computing applications do not support, such as context-awareness, massive decentralisation, autonomous behaviour, adaptivity, proactivity, and innate collaboration. This paper presents a new computational model and middleware that reflect support for the required the properties. The sentient object model is proposed by the CORTEX project to support the construction of ubiquitous applications. A flexible, run-time reconfigurable component-based middleware has been built to provide run-time support to engineer the sentient object programming paradigm. An application infrastructure using sentient objects to enable cooperation between autonomous and proactive vehicles has been implemented to demonstrate the appropriateness of the computational model and the validity of the middleware for pervasive mobile ad hoc computing.

Relevance to thesis: This paper describes a middleware solution for context-aware, autonomous objects in ad hoc network environments. Such a solution can be used to support context-aware work processes in pervasive environments.
My contribution: I participated in the CORTEX project from September 2003 to April 2004. I was the leading author of this paper.
Where in thesis: Chapter 15, page 193.

P7: Mobile Peer-to-Peer Technology used to Promote Spontaneous Collaboration

Mobile devices have in the recent years been able to form peer-to-peer networks making it possible for people to share information and interact. This technology makes it possible to create new tools that initiate and improve human collaboration. The paper presents experiences from creating a prototype for spontaneous collaboration, supported through the appliance of peer-to-peer applications on wireless mobile devices with the ability to form ad-hoc wireless networks. The paper gives an example of an application that promotes human collaboration by using peer-to-peer technology, and describes the technological challenges faced when implementing such applications. The paper reflects on some user

tests that were carried out using this prototype.

Relevance to thesis: This paper presents experiences from spontaneous peer-to-peer collaboration on mobile devices. Such technology shows a way to create cooperative services to be used when mobile that can help support cooperation in smart work processes.
My contribution: I was co-advisor for the project and participated in the fine-tuning of this paper.
Where in thesis: Chapter 16, page 203.

P10: The Nidaros Framework for Development of Location-aware Applications

This paper presents the Nidaros framework for developing location-aware applications that provide location dependent functionality based on the current location of the user. The framework can be used to develop location-dependent advertisement, city guides, guides for tourist attractions, etc. The framework consists of three main components: *A runtime system* that manages user locations and the interaction with the user clients; *a creator tool* that is used to map information and multimedia content to locations; and *a logging tool* that is used to log the movement of users to monitor the interest for certain locations. The paper also describes an implementation of a location-aware tour guide for the Nidaros Cathedral in Trondheim that can run on different mobile devices. Further, the paper describes experiences from installing, configuring, and running a location-aware tour guide in a real environment. A demonstration of the tour guide was tested on PDAs and mobile phones. The event received a lot of publicity.

Relevance to thesis: This paper presents a concrete context-aware system with emphasis on adaptation of content to the current location. The presented framework can be extended to also handle other context types than location.
My contribution: I was co-advisor for the project, and participated in the fine-tuning of this paper.
Where in thesis: Chapter 17, page 217.

4.7 Summary

This chapter have presented our main contribution and how the different papers can be related to the proposed architecture for support of smart work processes. We have, however, not integrated all the reported contributions into a complete supporting system ready for a thorough validation with respect to stated or unstated requirements.

CHAPTER 5

Thesis Evaluation

This thesis has presented surveys, concepts, frameworks, and enabling technology with the purpose of investigating how to support work especially in mobile, context-rich environments. We have proposed the concept of a *smart work process* to characterise work processes in environments with dynamic and changing properties, both electronical and physical. *Mobile work* has been clearly distinguished from *nomadic work*. This has made us able to propose models and architectures tailored to support mobile work. We have developed and investigated enabling technology and applications that can be used to realise the smart work process architecture.

This list summarises the main contributions of this thesis:

C1 A characterisation framework for mobile work to understand and find requirements for mobile work processes.

C2 An architecture to support context-aware, mobile (smart) work processes.

C3 Enabling technologies to support mobile work.

C3.1 A context-aware middleware for mobile ad hoc environments to support context-aware, mobile (smart) work processes.

C3.2 The NIDAROS framework for location-aware support on mobile devices.

This chapter is organised as the following: Section 5.1 describes how the research ques-

tions that were identified in the introduction of this thesis have been addressed, and Section 5.2 evaluates the contributions.

5.1 Propositions to our Research Questions

In Section 1.3 the research topics were outlined, describing the focus of the research in this thesis. Below is a summary of how these research questions have been addressed:

RQ1: What are the special properties of mobile work and how do these influence how to build supporting technology?

We have based on the surveys in paper P4 and AP4, and the work on the MOWAHS characterisation framework, differentiated between different types of mobile work. The most important property of service and *true* mobile work is the demand for location or context change to be able to perform work. Nomadic work cannot be regarded as mobile work since context changes are not necessary to perform work.

RQ1.1: How can concepts and frameworks be developed and used to understand and specify how mobile work processes can be supported?

We have developed the MOWAHS characterisation framework to capture and describe mobile work processes. In the framework, properties related to, e.g., mobility, time, and network support have received special attention. The framework is applicable for all kind of work that requires computational support when mobile. The MOWAHS characterisation framework is described in papers P1, P3, and P8. The *MOWAHS characterisation framework* is partly answering RQ1.1 and is stated as main contribution **C1**.

We propose, based on the differentiation between nomadic and mobile work, the concept of the *smart work process* to capture the adaptive behaviour needed to support mobile work processes computationally. This is reported in paper P9. The *smart work process* concept and architecture is partly answering RQ1.1 and is stated as main contribution **C2**.

RQ1.2: How can such concepts and frameworks be used to support the development of mobile work support systems?

The MOWAHS characterisation framework is especially developed to describe mobile work processes in detail. Each part of the work process is then characterised in detail to identify both functional and non-functional requirements. Based on the characterisation, we can extract different types of indicators that can help to identify the most challenging parts of a work support system.

Paper P2 reports experiences from developing the MONGO system, paper P8 reports

experiences from using the MOWAHS characterisation framework, paper P10 describes the smart work process architecture, and paper P14 describes the Nidaros framework.

The *MOWAHS characterisation framework* is partly answering RQ1.2 and is stated as main contribution **C1**.

The *smart work process* concept and architecture is partly answering RQ1.2 and is stated as main contribution **C2**.

The enabling technology reported in paper P2, P9, and P10 is partly answering RQ1.2 and is stated as main contribution **C3**.

RQ2: How can work process and context information be modelled and utilised in mobile, heterogeneous, and dynamic work environments?

We propose an architecture for supporting smart work processes based on the results found in RQ1.1. The architecture is presented in paper P9, and forms together with *smart work process glue model*, a basis to develop mobile work support systems that are context-aware. The *smart work process* concept and architecture is partly answering RQ1.2 and is stated as main contribution **C2**.

Further, we have developed the Nidaros framework. This framework is context-aware, delivering tailored information based on both location and device type. This work is presented in paper P10. The Nidaros framework is partly answering RQ2 and is stated as contribution **C3.2**.

RQ3: How and which technologies can be used to create an infrastructure to provide support of mobile work processes?

This research question is partly answered through all the papers that present technological solutions to different aspects of mobile support needs. We have not been able to integrate all the different technologies into one application, but we believe that the experiences can be used to develop mobile work support applications.

Paper P2 presents experiences from developing a mobile work support application, MONGO.
Paper P5 and P6 present a context-aware middleware for use in mobile, ad hoc computing environments.
Paper P7 presents a P2P technology to promote spontaneous collaboration.
Paper P9 presents the glue architecture for supporting mobile work processes.
Paper P10 presents the Nidaros framework, a server-based framework for context-aware delivery of content to heterogeneous devices.

The CORTEX middleware for MANET is partly answering RQ3 and is stated as contribution **C3.1**. The Nidaros framework is partly answering RQ3 and is stated as contribution

C3.2.

The rest of the papers are presenting partly solutions for different issues in support for mobile work processes. These papers are partly answering RQ3 and are stated as main contribution **C3**.

5.2 Evaluation of the Contributions

We have in this thesis outlined the concept of the *smart work process* to capture adaptive, context-aware work processes performed in mobile, heterogeneous environments. Although such a concept is somewhat alluring and at the same time vague, it outlines a lot of challenges related to representation, collection, storage, computation, and use of information. In addition, smart work processes requires a ubiquitous computing environment to enable communication between the user, the work process, and the physical and collaborative environment.

We will below evaluate our contributions based on contribution towards the MOWAHS research project, the presented state-of-the-art, and reflect and discuss weaknesses and strengths of the contributions.

The project goals of the MOWAHS project were threefold:

G1 Helping to understand and to continuously assess and improve work processes in virtual organizations.

G2 Providing a flexible, common work environment to execute and share real work processes and their artefacts, applicable on a variety of electronic devices (from big servers to small PDAs).

G3 Disseminating the results to colleagues, students, companies, and the community at large.

Goal G3 has been met through 18 papers whereof 10 are included in this thesis. All the papers have been published and presented in international peer-reviewed conferences. In addition, the research in the project has led to industrial contact with different companies like Statoil, Telenor FOU, Klipp og Lim Media, RadioNor, and Marintek.

C1 – A characterisation framework for mobile work to understand and find requirements for mobile work processes.

Evaluation of contribution against the MOWAHS project goals:

C1 contributes partly to goal G1 of the MOWAHS project goals. We identified early in the MOWAHS project the mismatch between the notion of mobility and virtual organisations. The contribution does, however, partly address some of the issues related to virtual organisations. We believe that the framework helps to understand and improve work processes in mobile work environments and thus can be used to assess and improve the work processes based on the mobile computation needs. The characterisation framework is, however, not developed to especially look at organisational issues like roles, responsibilities, cooperation, or coordination.

Evaluation of contribution against state-of-the-art:

The characterisation framework can be characterised as requirements engineering, and is especially directed towards mobile process support. We are using scenario analysis to perform characterisation and then discover requirements. Scenarios has been used for a least a decade to provide, e.g., input for generating use cases [JC92], experience-based narratives for requirements elicitation and validation [PTA94], or to give advice in requirement analysis and validation [Sut98].

The use of indicators as expressed in the characterisation framework is however not included in any work as we have discovered so far.

We mean the characterisation framework may help in better modelling and analysis of problem domains as stated as a research challenge in Section 2.8. Further, the framework helps in discovering non-functional requirements based on actual user scenarios/-work processes. The results can be used to develop software architectures suitable for the domain.

Reflection:

The developed framework helps us capture requirements and represent work processes, but the framework is not an absolute tool to ensure a correct picture of the application requirements. Such frameworks will always have subjective interpretations, and thus, give different answers based on, e.g., the knowledge of the application area and of technology in general. We therefore have entitled our measurements as indicators (in the MOWAHS characterisation framework) and avoided any absolute interpretation. This is also reflected by characterisations using an ordinal scale. The main disadvantage with some of the indicators is that they always will tend to be in the middle of the scale. Thus, it can be hard to use the indication to give any clear conclusion with respect to the issue characterised.

The MOWAHS characterisation framework has been through several reviews and been adjusted accordingly. We have tried to avoid anchoring the indicators to the present technological challenges. Instead, we have tried to have indicators that can have inherent properties regardless of supporting technology. It is, however, important to adjust some of the characterisations to manage change in the available technology.

C2 – An architecture to support context-aware, mobile (smart) work processes.

Evaluation of contribution against the MOWAHS project goals:

C2 contributes partly to goal G2 of the MOWAHS project goals.

The notion of smart work processes (paper P9) is applicable to all kind of work organisations, including virtual organisations. The architecture has, however, not been fully implemented and is therefore hard to validate against goal G2. We have studied emergent virtual organisations as part of MOWAHS and the coordination service (COWCS) in Section 4.4 is a result of this work. In addition to coordinate workers through mobile clients, we also rely on a central workflow enactment service to provide support to clients in coordination of artefacts and work processes.

Evaluation of contribution against state-of-the-art:

Many mobile professions have been studied and/or provided mobile technology in the last decade. The work of the PHCS project [JB04] partly covers context-aware work processes in a hospital. The context is fetched using RFID sensor technology identifying objects (like a patient). The work processes is adaptable for different roles (nurse, doctor) and situations based on, e.g., the current treatment state of the patient (e.g. need for medication). The work processes are mostly pre-defined, but adaptable based on available data. Activities are not fetched from the environment and are not directed by any sensed physical context except for presence of objects in the environment. Collaboration is not directly supported in the PHCS project (deduced from the description in [JB04]). Mobility is not a separate issue in this system, since the important information is the co-location of roles and things, i.e. the identification of the user and nearby objects.

The idea of using process templates (paper P9) is not novel and is used in both commercial and academical workflow systems as presented in Section 2.6.2. The main difference from these approaches is that our approach is using context information to automatically select fitting definitions. In our approach, information about processes can as well be decentralised and distributed in the physical environment and be triggered by changes of the state in the environment. The process definition can be distributed to possible actors in the environment and thus adapt the work process. Context information is in addition an important factor of influence in the actual work process enactment. Another issue of these approaches is the missing support for mobility.

Reflection:

We have not evaluated the architecture in detail because of a missing implementation. The architecture has not been fully implemented because of the time and resource constraints of the MOWAHS project. Many research challenges remain to be solved in this area, especially on how to model context-aware processes and how to perform inferences based on context information and process definitions. This is reflected in other work on context-aware applications. These have a limited range of context types included.

One of the weaknesses of our approach is the uncertainty whether it is possible to model and utilise context information in work process models in such an extent that the user is not overloaded with context management. One of the main reasons is that the current physical operations in work are based on human skills and experience. The context inference is performed unconsciously by the workers, i.e., it is hard to understand how the environmental context is applied by the workforce. In addition, it is hard to predict how the future ubiquitous environments will change the way we work.

An observation is that the mobile phone in many ways has changed how we coordinate activities in all kind of work. The data presented for the home nursing care in Chapter 8, shows that more than 92% of the nurses were using mobile phones in their daily work. This means that almost all have access to a computer through the mobile phone. Modern mobile phones have integrated several ways to communicate like GSM, GPRS, UMTS, WLAN, Bluetooth, and IR. In that respect, the vendors of sensors and other ubiquitous equipment have many different possibilities for communication that can provide data for work process enactment.

C3 – Enabling technologies to support mobile work.

Evaluation of contribution against the MOWAHS project goals:

C3 contributes partly to goal G2 of the MOWAHS project goals.

We have developed a flexible work environment, MONGO (paper P2 and partly P8), to support mobile work processes on heterogeneous devices. The MONGO system has been extended to support location-aware work processes, and thus the system is adaptive with respect to device type and to the user location. The system is, however, not supporting the notion of virtual organisations.

The CORTEX middleware is applicable for mobile ad hoc networks (paper P5 and P6). We have experimented using the middleware to provide a component platform for communication between work support clients (PDAs). A problem we experienced was a stable ad hoc WLAN environment on for our PDAs. Further, we have worked with the sentient object paradigm towards work processes. The CORTEX middleware is configurable and reconfigurable and thus provide a flexible environment for application and service development for mobile work processes.

The P2P application for spontaneous collaboration (paper P7) is a first step towards a collaborative component to support collaboration within the smart work process architecture (COWCS). We believe that technology can be further extended with context-awareness towards collaboration in work environments. The technology can thus support work environments for different types of organisations, included highly virtual organisations that can appear in mobile ad hoc environments.

The Nidaros framework (paper P10) is based on state-of-the-art technology and can al-

ready be deployed in industrial applications. The framework is usable to relate multimedia information to objects and locations and can thus connect the virtual world with the physical.

The smart work process architecture (paper P9) provides the glue of the enabling technologies presented in paper P2, P5, P6, P7, and P10. The technologies have, however, not been integrated.

Evaluation of contribution against state-of-the-art:

Each paper has a separate related work section and has thus compared the contribution towards state-of-the-art. The different papers have addressed different challenges related to the sub-areas reported. We can see the applicability of the approaches in realising the smart work process architecture.

CSCW researchers are working with collaborative systems using theory from the social sciences, e.g., activity theory. Our approach is extending the reach of such systems to also give support to unconscious collaboration when working in dynamic and ad hoc environments.

We have not implemented a ubiquitous computing environment to support mobile work processes. Instead, we have suggested the applicability of research and applications of such environment to be an important catalyst for extended research context-aware work support. Augmented artefacts [SGKK04] and environments [Myn00], and ”smart dust and brilliant rocks” [Sat03] all provide information and data that is directly or indirectly usable for smart work processes.

Reflection:

We have developed a flexible work environment for supporting mobile work processes, but this environment is mainly supporting single user work processes. We address device heterogeneity with content and GUI adaptation to these devices based on Web technology.

The technologies needed to support smart work processes range from hardware including ubiquitous sensors and robot technology, wired/wireless communication, and software. In addition, there is a need for extensive (social science) studies of the current mobile workforce to suggest a supporting infrastructure and software to support collaboration and work processes in smart environments.

The CORTEX middleware is applicable for supporting both fixed and ad hoc infrastructures, adapting to QoS requirements on the mobile device and the network protocols. We believe that the sentient object paradigm is applicable as a metaphor for smart work processes as well as for the components suggested in the smart work process architecture.

CHAPTER 6

Future Work

The vision of context-aware work processes is not easily realised because of the lack of realistic physical environments. We need environments that include enough context sensing and monitoring facilities to give sufficient information to deduce actions. The current work processes are mostly performed without any computational context support.

Location-awareness is the foremost property included in the current initiatives for context-aware work support systems. The location is most often derived from other information like IDs or tags (e.g. RFID). The working environment and work processes have to be further evolved to come to a state where the coupling of and relationship between context information types and activities can be specified and implemented. This must be done in such a fashion that the user involvement in the specification process can be minimised. We envisage an environment where artefacts themselves can help to deduce the relationship between context and activity related to the artefact itself, or to other artefacts in the environment.

The notion of smart work processes and the supporting infrastructure has to be further explored to empirically find the usability and applicability of such processes. *Accountability* of the supporting infrastructure and the process support software has to be both improved and proved. *Dependability* is another issue that have to be explored and suitable methods need either to be incorporated or created to ensure a computing environment that is reliable and veracious.

A way to test the applicability of smart work processes is to extend the scope of context-aware work processes to also include non-work activities related to e.g. home and car

management. Importantly, such activities are similar to work processes because context information is used to propose and initiate activities. A *smart home* can embed a range of sensors and actuators that can be used to explore both situated actions and situated planning.

Important activities related both to the normal work and home environments can be integrated into activity support applications. For normal workers, such systems can help to make both short- and long-term plans related to daily activities. Calendar/task tools can, e.g., be extended with an interface to the smart home to send/receive and update activities. Shopping lists can, e.g., be automatically updated as items are used, activities related to home maintenance can be automatically be generated measuring the degradation of painting, plumbing, electricity items, etc. During a working day, the worker can be reminded at the end of the day, or in pauses about home activities that can be performed at the current location.

The support for context-aware or smart work processes is to a small degree covered in the literature and is thus a new area of research within pervasive computing and workflow/process support. Since the domain has little coverage in the research community, many challenges arise in the cross-section of mobile/wireless computing, ubiquitous/pervasive computing, and workflow. Examples of relevant challenges to be explored in the future include:

- Specification of *contextual pre/post-conditions* related to some process goal.
- Specification of *environmental behaviour* related to an activity (adaptation).
- Specification of a *uniform representation of sensors and actuators* from a process enactment perspective to make it possible to reason about and change the context state of the environment.
- *Planning, specification, and execution* of activities in concert with the current environmental context. These challenges relate to how the dynamics are handled by the workflow enactment services, both on client and on server.
- *Managing process changes to ensure a consistent state* of the process.
- *Managing the dynamics of ad hoc activities and process changes* locally and in a central enactment service.

CHAPTER 7

Concluding Remarks

We have in this thesis explored and reported research related to support of mobile work in different settings. In the future, we believe that vision of the ubiquitous computing environment will be realised in many different forms. It is, however, probably a long way to go before computing is so integrated into everyday life that we forget about the presence of computers. Importantly, we can already now start to explore and exploit the possibilities the technological evolution continuously is presenting in form of enabling technologies and concepts.

Services and applications are very slow to emerge except for high-profit areas like gaming, adult entertaining, and partly in e/mCommerce. In these areas, the number of customers may very fast reach a critical point for return of investment. To improve the return of investment in other mobile application areas, an infrastructure of communication, information, services, and devices must have reached a critical point before professionals can defend their investments. Work processes must naturally change to adapt to the changing infrastructure, and the environment must be able to provide the same kind of service everywhere.

We believe that smart work processes are likely to first be deployed in hazardous environments like shipyards, offshore installations, and in environments where work processes already are automated and/or performed by machines and robots. Work place, public, and environmental safety and protection of expensive equipment will be the foremost motivations for such investments.

Part II

Survey Paper

CHAPTER 8

P4 – A Survey of Mobile Support Needs in the Home Nursing Care

Carl-Fredrik Sørensen, Bjørn Næss, Øyvind Sølberg Strand, Alf Inge Wang and Reidar Conradi [1]

Abstract
This paper describes the results of a survey performed in the home nursing care at different municipalities and health regions in Norway. The purpose of the survey was to map which kind of technical or informational mobile support the personnel reckoned as important when visiting patients at their homes or elsewhere when mobile. The sample was drawn based on an index that groups the regions based on the size of the population. The people in the home nursing care are familiar with the use of mobile phones and computers. They also demands access to certain type of information when mobile to support their daily work.

Keywords:
Mobile work, Mobile Health Informatics

[1]Dept. of Computer and Information Science, Norwegian University of Science and Technology (NTNU), N-7491 Trondheim, Norway. Phone: +47 73 594485, Fax: +47 73 594466, http://www.mowahs.com

8.1 Introduction

The home nursing care is an important part of the public primary health care in Norway. More than 30% of the expenses in the municipalities are used in services related to health and elderly care. More than 43% of these expenses are used in the home nursing care. About 90000 people are employed within the nursing care in Norway. These serve around 180000 patients [Hyg03].

The population in Norway is spread over relatively large distances, requiring the nurses to travel correspondingly. The home nurses have thus nomadic working conditions, and are often alone when decisions regarding a patient have to be taken. They serve a heterogeneous group of patients including physically and mentally disabled people, mental patients, convalescents, and elderly people with different demands.

In this survey we have focused on revealing the communication and information needs in the home nursing care. The home nursing care covers the demands of home care and medical treatment. The home nursing care services also include coordination and communication of other services like hospitals, nursing homes and similar.

Patient information is spread across both digital and manual archives. Both kinds of archives are used within the home nursing care, and information is often duplicated. This situation may lead to incomplete and obsolete information in the patient medical records. Reporting is handled through meetings in addition to adding information to the different archives. Informal communication between nurses is frequent and is usually not documented. In a few years it is expected that the health care personnel will be able to access all patient records electronically. This will greatly improve the information flow between the different instances that a patient directly or indirectly is cared by. A further improvement would be to access the information also when mobile.

8.2 Description of the Survey

The survey is based on the current information flow and demands in the home nursing care, and measures relatively few variables in a large sample to make it representative for the whole population. The questions were given as closed alternatives in a Likert-type scale. The respondents were selected by probability sampling. The population was divided into subgroups based on KOSTRA[2] [Sta02]. KOSTRA groups the 434 municipalities in Norway into 16 groups based on population size and their economical conditions. From these groups 39 random samples were drawn. The numbers of municipalities drawn from each group were proportional to the size of the groups. Ten questionnaires were sent by mail to the drawn municipalities. Some of the municipalities could however not an-

[2]KOSTRA is an abbreviation for ”Municipality-State-Reporting”

swer more than 5 questionnaires because of the small size. This resulted in a quite high dropout rate. Of a total of 390 questionnaires, 175 were returned.

The questionnaire was divided into 4 parts, where parts 3 and 4 have sub parts. Part 1 contains questions related to demographical data. Part 2 contains general questions related to the normal working day. Part 3 contains questions related to the nursing care, and for demands for administrative and professional information and services. Finally, part 4 describes a technological scenario and asks questions related to experiences and expectations to new technologies, and questions of which kind of functionality a mobile device should offer.

8.3 Results from the Survey

8.3.1 Part 1: Demographical Data

The age distribution of the respondents was: 16.0% in the group 20-29 years, 24.0% - 30-39, 39.4% - 40-49, 20.0% - 50-59, and 0.6% above 60. This gives an average age around 40 years. The respondents have in average worked 15.9 years in the health care, and 85.5% have permanent employment. The educational level of the respondents is distributed evenly between college and university. Six of the 175 respondents did not have higher education. 86% of the respondents were women. This confirms that the nursing care traditionally is dominated by women. 81.7% had beside their occupation as a home nurse, additional responsibility related to administration, specific patient care, and other kind of leader roles.

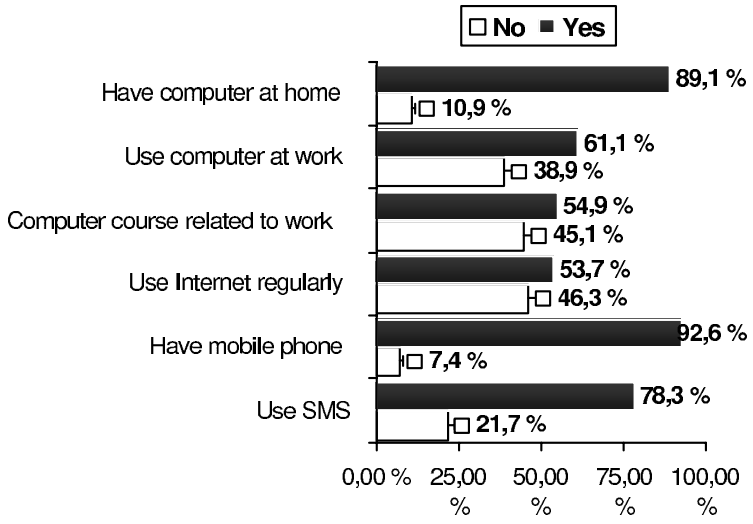

Figure 8.1: Use of Information Technology

Figure 8.1 shows the use of information technology – both professionally and privately. A surprising large amount of the home nursing care personnel are users of different kind of computing and communication technology.

8.3.2 Part 2: The Perceived Normal Working day

The questions in this part are related to the perceived normal working day in the home nursing care. The questions in this part appealed to the feelings of the respondents. The questions were given with closed answer alternatives (1-Strongly Disagree, 2-Disagree, 3-Uncertain, 4-Agree, and 5-Strongly Agree). The results indicate that meetings are necessary to get the required information for the working day (94% agree or strongly agree with Q2-3: $\mu = 4.37/\sigma = 0.688$), and they are social important (79% agree or strongly agree with Q2-2: $\mu = 3.83/\sigma = 0.898$). 78% of the respondents mean that the allocated tasks take longer than estimated (Q2-12: $\mu = 3.79/\sigma = 0.747$), and 70% find it important/nice to know were the colleagues are (Q2-11: $\mu = 3.66/\sigma = 0.815$).

8.3.3 Part 3: Demands for information and services

Part 3 in the survey asked for information and services that a health care worker demands during a working day. The respondents were asked to consider a situation when mobile, either on the move or visiting the patient. Part 3 is divided into three parts where 3a considers the treatment work, 3b considers issues with respect to administration of the working day and other administration tasks, and 3c looks at demands for different types of professional health-related information. The questions in part 3 and 4 were given with closed alternatives (1-Not at all, 2-Little, 3-Some, 4-Much, and 5-Very much). The respondents answer generally quite high in this part. There are differences between the parts as will be presented below. Professional information, and information and services related to the patient treatment, are valued as more important than administrative information and services.

Part 3a: Patient treatment

Table 8.1 shows the results of some of questions related to information in the patient treatment. The respondents value access to this kind of information as quite important when mobile.

It is a trend in this part that people without any specific responsibilities in the whole answer lower to every question. The mean value of this group is 3.64 compared to 4.01 in the total mean. People with a specific patient responsibility have a mean value of 4.18 for all questions. Zone leaders have a higher score to Q3a-15 than the rest. It is not clear from

No.	Question	μ	σ
Q3a-4	General information about the patient (like closest relatives, family circumstances, living situation, professional career etc.)	4.15	.940
Q3a-5	Access to the patient epicrisis/medical records/case history (previous and current clinical picture and treatment, allergies etc.)	4.23	.917
Q3a-6	Information about the patient task, the care study, and what needs to be brought to the patient	4,49	.800
Q3a-8	Be able to make requisitions of services from others (e.g. food transport, occupational therapy etc.)	4.07	.955
Q3a-9	Information about medical tests and the results from these	3.94	1.174
Q3a-11	Information about which kind of patient support equipment a patient possesses	3.90	1.050
Q3a-12	Possibility to record needs for patient support equipment	4.08	.946
Q3a-13	Be able to report the need for additional help when visiting a patient	4.34	.940
Q3a-15	Be able to record start and end of patient visits	3.11	1.276

Table 8.1: Part 3a: Patient treatment (N=170)

the survey whether the zone leaders want to follow up the personnel for security reasons, or if they in a higher degree want to monitor them.

Part 3b: Administration of the working day

Table 8.2 shows the average score and standard deviation of questions related to administration of the working day and other kind of administrative tasks.

The respondent should imagine to be working on a mobile task. A little bit surprising for us was that this kind of information was less demanded. Examples of tasks were e.g. to read hour lists or edit recent mileage use. The respondents did however value information like name of patient responsible, address information of the personnel, and being able to record needs for a place at institutions.

The standard deviation is somewhat higher in this part, but we have not been able to find any particular reason for this. It is however a tendency that the zone leaders and personnel with specific patient responsibility is more positive than the rest of the respondents.

Part 3c: Professional health information

Table 8.3 shows the results of questions related to what kind of professional health information the home nurse needs during a working day (working mobile). Generally, the

No.	Question	μ	σ
Q3b-1	Be able to apply for, or record need for a place at an institution	3,71	1.219
Q3b-3	Get information about what are in stock (e.g. the medicine stock or the supporting equipment stock)	3.24	1.104
Q3b-4	Access to personal hour lists and information about the salary	3,12	1,543
Q3b-5	Possibility to look at your own or others‘ rotation scheme	3,37	1.535
Q3b-6	Be able to find who is having duty after yourself to be able to inform the patient about it	3,39	1,266
Q3b-7	Be able to find mileage use e.g. last week, month, or year	2,42	1.429
Q3b-8	Access to address and phone number to the personnel (both fixed and temporary staff)	3.75	1.317
Q3b-9	Be able to report the need for additional help when visiting a patient	4.13	1.033
Q3b-10	Overview of responsible professional health persons for the different patients	3.79	1.111

Table 8.2: Part 3b: Administration of the working day (N=167)

respondents mean that they have ”much” need for access to professional information. The most important information is related to medicine effect and by-effects.

No.	Question	μ	σ
Q3c-1	Information about medicine effects and by-effects	4.23	1.048
Q3c-2	Access to information about concrete diagnoses (pathology)	4.09	1.049
Q3c-3	Information about diets for different diagnoses/diseases	3.81	1.122
Q3c-4	Information about practical procedures that are used in your job	4.05	1.046

Table 8.3: Part 3c: Professional information (N=166)

It is interesting to observe the difference in responses between the zone leaders and the others in which questions that are regarded as most important. Zone leaders value Q3c-4 highest, where the others value Q3c-1 highest. This difference is important to be aware of when developing support systems in the home nursing care. Zone leaders are probably the group that is most natural to use when specifying requirements for support systems. The results show that it is important to also include other stakeholders when prioritising requirements.

8.3.4 Part 4: Requirements to a Handheld Device

Part 4 in the survey focused on technology. The first part asked for experiences and expectations related to new technology. We also presented a futuristic scenario for the

respondents where they have access to a multi-function, multimedia handheld computer without any technical limitations.

Part 4a: Technological Attitude

Table 8.4 shows the results of questions related to experiences and expectations to new technology. Some of the questions were negative formulated, and the scores were therefore adjusted. Most of the respondents have positive experiences and expectations to use of new technology.

No.	Question	μ	σ
Q4a-1	I believe introduction of new computer systems will make my working day simpler	3.82	.964
Q4a-2	Use of new technologies is time-consuming and ineffective	3.49	.908
Q4a-3	I feel comfortable using computers	3.42	1.110
Q4a-4	Use of technology leads to increased employee surveillance at the workplace	3.74	0.952
Q4a-5	My impression is that computers create a larger distance between people	3.18	1.063
Q4a-6	I think the technological development is exciting	3.75	.978

Table 8.4: Part 4a: Technological attitude (N=170)

Part 4b: Functionality a mobile device should offer

Table 8.5 shows the results of questions related to what kind of functionality a mobile device should offer the user. The respondents are valuing different kind of information and services as quite important, but the answers have a larger variation than in part 3. In this part, being able to record and update patient information is judged as the most important. Access to professional documentation and medical test results is also judged as quite important. The respondents did not wish to record information and statistics to non-health organisations using mobile devices. They are also less positive to record time coming to or leaving the patient, or to update the hour lists.

8.4 Analysis of the Survey

The questions in part 4a were aggregated to get an impression of the technological attitudes of the respondents. The result of the aggregation was divided into 4 subgroups where each group represents a different attitude to technology. The respondents had gen-

No.	Question	μ	σ
Q4b-1	Be able to record and update patient information, if possible directly into the computer system	4,24	1.020
Q4b-2	Be able to show medical test results to the patient	3.99	1.121
Q4b-3	Give me access to professional documentation like procedures or corresponding	4.12	.989
Q4b-4	Let me consult with a physician or other kind of professional health personnel when in demand for assistance concerning the patient (using e.g. sound or video)	3.89	1.158
Q4b-5	Let me write and send a message to set up an appointment with other health personnel (physician, physiotherapist etc.)	3.93	1.084
Q4b-6	Let me write and send messages to order new medicines	3.58	1.291
Q4b-7	Have the possibility to record information when coming to or leaving a patient to make other aware of your tasks to ensure your own safety	3.17	1.375
Q4b-8	Give me an overview of patient relatives/contacts, and eventually let me call or send messages to these	3.90	1.123
Q4b-10	Be able to let me give messages (e.g. in text or voice messages) to my colleagues	3.93	1.188
Q4b-12	Be able to anytime read and update the hour list	2.60	1.380
Q4b-13	Be able to access your own or others rotation scheme, and eventually record wishes in the rotation scheme	2.76	1.448
Q4b-14	Let me record and get informal information concerning my work tasks	3.34	1.207
Q4b-15	Be able to record and update information about the task itself (when, where, who etc.)	3.79	1.159
Q4b-17	Be able to give general information about diets	3.41	1.162
Q4b-18	Be able to record refund of expenses (mileage coverage, patient shopping list etc.)	2.84	1.399
Q4b-19	Be able to record or give statistical information to non-health databases	2.22	1.332

Table 8.5: Part 4b: Questions of which kind of functionality a mobile device should offer (N=158)

erally a positive attitude to technology. Table 8.6 shows which factors that have the highest correlation to technological attitude[3].

Attitude to technology (categorised)	**Correlation**
Q1-13: Do you use the Internet regularly?	-0.461
Q4b-1: Be able to record and update patient information, if possible directly in the used computer system	0.369
Q4b-5: Let me write and send a message to set up an appointment with other health personnel (physician, physiotherapist etc.)	0.293
Q1-3: Formal education	0.278
Q1-15: Do you use SMS (Short Message Services)?	-0.245
Q4b-15: Be able to record and update information about the task itself (when, where, who etc.)	0.235
Q4b-8: Give me an overview of patient relatives/contacts, and eventually let me call or send messages to these	0.232
Q4b-2: Be able to show medical test results to the patient	0.228
Q1-11: Do you use computer at work?	-0.217
Q3b-4: Access to personal hour lists and information about salary	-0.211
Q4b-6: Let me write and send messages to order new medicines	0.211
Q4b-10: Be able to let me give messages (e.g. in text or voice messages) to my colleagues	0.208
Q4b-7: Have the possibility to record information when coming to or leaving a patient to make other aware of your tasks to ensure your own safety	0.204

Table 8.6: Factors that correlate with technological attitude ($p<0.01$, N=165)

Regular Internet usage and technological attitude has a strong correlation. Use of SMS also correlates significantly with technological attitude. This can either be interpreted as that use of Internet and SMS gives better attitudes to technology; or that people with a good attitude to technology in a larger degree will be a user of that kind of services. It is a quite strong correlation between Q4b-1 and the technical attitude. Table 8.7 shows the distribution of answers to this question and technological attitude in general.

The rows show the answer alternatives in Q4b-1, and the columns in the distribution show the subgroups in the aggregated variable; technological attitude. Only 1/3 of the respondents in the group with lowest technological attitude strongly agree that they should be able to record and update patient information. In the group with the strongest technological attitude, close to 70% wish this functionality (strongly agree). This functionality is recognised to be the most central in all kind of health applications, and is also most valued in part 4b ($\mu = 4.24$). Formal education correlates with technological attitude. People

[3]Negative correlation in Q1-15, means that those who answer NO on this question, have lower attitude to technology

		Attitude to technology (categorised)				Total
		1	2	3	4	
Q4b-1: Be able to record and	1	12.5%	5.9%		2.3%	4.9%
update patient information, if	2	2.5%	2.9%			1.2%
possible directly	3	20.0%	8.8%	4.5%	4.5%	9.3%
in the used	4	35.0%	50.0%	31.8%	22.7%	34.0%
computer system	5	30.0%	32.4%	63.6%	70.5%	50.6%
Total		100%	100%	100%	100%	100%

Table 8.7: Cross-tabulation of Q4b-1 and technological attitude categories (part 4a) (N=161)

with lower education have generally less expectations to and experiences from the use of technology.

We asked the respondents how much they valued the possibility to report needs for additional medicine (Q3a-2: $\mu = 4.35/\sigma = 0.969$). They are less positive to be able to order new medicines by sending electronic messages (Q4b-6). We found a significant positive correlation (0.205/p¡0.01) between the technological attitude and this question. People already using SMS valued, not surprisingly, this functionality higher than the rest (Using SMS $\mu = 3.70$, not using SMS have $\mu = 3.15$). We found significant correlation (p¡=0.01) between technological attitude and only one of the questions from part 3. This was Q3b-4, access to personal hour lists and information about the salary. The other questions from part 3 have not significant correlation. This observation is positive for the validity of the survey since part 3 should be answered independent of technology. We did not find any significant correlations (p¡=0.01) between technological attitude and any of the questions in part 3.

8.4.1 Informal Communication

Ellingsen [Ell00] and conversations with home nurses indicated use of informal communication channels like message books or post-it notes. KITH[4] [KIT01] means that informal information is important. The results of our survey, however, show that this kind of communication has a lower score than the other questions (Q3a-16: $\mu = 3.49/\sigma = 1.050$). This can be interpreted in different ways; the question may be vaguely formulated, or that this kind of functionality is lower prioritised or that they want to keep the informal communication as-is. It is however interesting that those personnel with less experience (students or trainees) have a mean of 4.75 to this question. The number of respondents in

[4]Expert Centre for Information Technology in the Health Care.

this group (4) is however to low to draw any conclusion, but we find informal communication to be especially interesting for qualitative studies.

8.4.2 Start the Working Day at Home

Q4b-20 asked about starting the working day at home. We wanted to investigate how the respondents felt about starting their work at home, and thereby identify needs to access required information using mobile/nomadic applications. However, the respondents were more negative than expected ($\mu = 2.39/\sigma = 1.471$).

We therefore wanted to investigate how this question correlated to type of municipality and thereby travel distance. A cross-tabulation between the municipalities and Q4b-20 shows significant differences, but it was difficult to interpret these results. We found a significant negative correlation (p¡=0.05) between this question and the number of people working in the unit. People in smaller units prefer to start the working day at home more than people in larger units. Table 8.8 shows the statistical significant correlations (p¡0.01) between the other questions in the survey and Q4b-20.

Questions	Correlation
Q4b-19	0.363
Q4b-18	0.357
Q4b-12	0.332
Q4b-13	0.310
Q4b-7	0.302
Q4b-6	0.288
Q4b-2	0.247
Q4b-4	0.247
Q3b-6	0.233
Q3b-3	0.226
Q4b-14	0.212

Table 8.8: Questions that significantly correlate with working at home ($p<0.01$, N=157)

An evaluation of the results lead us to the following observation: The wish to start the work from home is small among the home nurses. 42.4% did not wish to work from home at all, but there is also a large group, 27.9%, who wish to do this more than "much" (4-in the Likert scale).

In Table 8.8, we observe that people that want to work at home have greater need for other services than the mean. E.g. to record statistics (Q4b-19), record refund of expenses (Qb-18), and update the hour list (Q4b-12); all administrative tasks. An investigation of the type of position shows that the zone leaders want to work at home (20.0% strongly

agree), while nurses with specific patient responsibility did not wish to work at home (3.3% strongly agree).

8.4.3 Requirements to a mobile support system

Part 3 in the survey was designed to reveal requirements of information and services. I n part 4b we presented technological proposals to find whether these were interesting with respect to the needs in part 3. For all categories in part 3, we have compared the results with the corresponding technological solution in part 4b. The respondents generally answered lower to the presented solutions than to the information needs. The nonconformity between the general need and the proposed solution shows that the solutions probably do not cover the perceived need. This issue is natural for future investigation and interpretation.

Management of Medicaments:

The category discussed here deals with management of medicaments, i.e. what medicaments a patient use, and be able to report of needs for additional medicaments. This category is covered by Q3a-1, Q3a-2, and Q4b-6. The questions from part 3a is quite similar and deals with getting overview of the patient medicines (Q3a-1: $\mu = 4.442/\sigma = 0.941$), or be able to report needs for more medicines (Q3a-2: $\mu = 4.35/\sigma = 0.969$). The correlation between Q3a-1 and Q3a-2 is of course strong (0.659, p¡=0.01). But both questions have a lower correlation to Q4b-6 that asks whether the respondent wants the possibility to write and send messages to make requisitions of new medicines. Q4b-6 ($\mu = 3.58$) has a strong correlation with technological attitude.

The respondents reported high demands for information and service related to patient medication, but did not wish the proposed functionality by using messages to order new medicines. This picture changes as a function of technological attitude where the respondents with strong technological attitude valued this information as very important ($\mu = 3.93$). The results indicate that management of medicines is very important and probably problematic.

Task:

Many of the questions in the survey cover the category task. These questions correlates internally in the different parts (3a and 4b), but have lower correlations between the parts. We cross-tabulated the parts by aggregating the questions, and found a significant correlation (0.285/p¡=0.01) between the aggregated variables. We interpret from the results that the technological proposals partly can fulfil the needs from part 3.

Access to professional health information:

Professional health information is a category that covers areas like access to health information for all kinds of health professions, information about medicaments and diets, information of laboratory results and what these imply, or access to professional information related to specific diagnoses. Part 3c mainly covers this category, and has the highest average answers in the survey. This indicates that functionality related to accessing professional information is strongly demanded in the home nursing care. This category is different from the other categories in that answers are equally or more positive to technological solutions (part 4) than in the more general parts 2 and 3. The home nurses regard access to this kind of information as an area where a handheld unit is suitable.

8.5 Discussion

The survey was pre-tested before being sent to the respondents. We had several questions related to operational sub-components to have a better overlap in the survey to increase the internal validity.

External validity:

The systematic errors are mainly related to non-responses. Of the 39 municipalities contacted in the survey, 28 have responded. The average number of responses from the municipalities is 6.2. The distribution of responses was uniform in the responding municipalities.

Internal validity:

The variables measured in this survey are based on earlier investigations within the health care [BN01] that strengthen the internal validity. The missing empirical investigations of use of mobile technologies and solutions make it difficult to estimate the internal validity. A qualitative study could raise important issues related to the internal validity of our study.

Reliability:

A pre-test showed that the questionnaire was built up thoroughly, and the question formulations were clear and concise. However, one of the respondents meant that some of the questions were leading: "*I think that the whole questionnaire leads us to wish to have a handheld device. Existing systems are not covered and this makes it hard to answer many of the questions*". The results from this respondent were considerable lower with respect to part 4b ($\mu = 1.9$) than the mean ($\mu = 3.4$). We did specifically ask for opinions to spe-

cific mobile support services. The respondents should neglect the current technological constraints. They should instead be open to how technology could be used to simplify, make more effective, and improve the work. This should be possible without any prior detailed knowledge of technology. The technological attitude correlated significantly to the half of the questions in part 4b (p¡0.05). This can be interpreted that the respondents were influenced by the technological view in part 4b, while in part 3 the respondents answered independently of technology. We therefore mean that the questionnaire was not leading with respect to technological solutions based on the statistical results. We have done an investigation of potential threats to reliability and validity, but not been able to find any factors that disqualify the design of the questionnaire. We therefore consider the survey to have a satisfactory validity and reliability. A negative aspect of the survey is the treatment of non-respondents. We have not been able to trace and explain all the missing returns of the questionnaire.

8.6 Related Work

KITH has surveyed the information flow and the information technology needs in four municipalities in Norway [KIT01]. The survey shows that all consider IT to be absolutely necessary within the nursing care, and that it improves the efficiency considerably.

In the ”Alta-project”, a test of implementing IT-services in the home nursing care was carried out [Ell00]. The purpose of the test was to investigate if it was possible to do all journaling, and to access important patient information when in the field. The test was performed using a provisional Wireless Local Area Network (WLAN), but problems with the network caused few results. The participants were, however, positive to use this kind of technological aids. The information flow was improved, and the patients could better influence decisions by participating in the journaling.

Tveito and Hasvold [TH02] refer to an observation of the people in the hospital ward that the work situation of the professionals is such that they are constantly interrupted. They are never able to plan their working days in detail because new situations occur all the time. The situation might be dramatic, or barely a nuisance, but during a working day there is no time to concentrate on one thing. This observation can be extended to the home nursing care where the home nurses often have tasks that are not planned for, and also have many patients during a day.

8.7 Conclusion

The survey confirms that the personnel in the home nursing care have great need to receive and exchange the kind of information presented here during a working day. Administra-

tive functionality might be omitted when mobile, but some professions have more need for this kind of functionality than others. The home nurses have in average a positive technological attitude. Some of the groups have less positive experiences, and it is therefore important to be observant of these when introducing new technology.

The respondents recognise the need for functionality related to the treatment and access to professional health information as very important. The survey shows that applications to support their work routines and methods might make the working day easier. We observe that the home nurses recognise the need for ad-hoc information and services when mobile. This kind of functionality is probably hard to implement in practice because of the hardware limitations on many digital devices. Offline information access requires storage space as well as pre-emptive caching of the required information. The digitalisation of the health care information will further make requirements for services to update central patient information when visiting patient at their homes. We therefore believe that mobile workers in after a while will become regular users of mobile information systems and services. A qualitative survey can elaborate more on what functionality a mobile support system should offer the home nursing care. It is also of interest to observe the home nurses to further capture processes and requirements not covered in this survey.

Acknowledgement

This paper is a result of work in the MObile Work Across Heterogeneous Systems (MOWAHS) project [MOW04]. MOWAHS is sponsored by the Norwegian Research Council's IKT-2010 program. The survey was performed as part of the interdisciplinary course "Experts in Team" at the NTNU. We want especially to thank Amar Bhargava, Rune Hansen and Karwan Rush for their participation.

Part III

Concepts and Framework Papers

CHAPTER 9

P1 – The MOWAHS Characterisation Framework for Mobile Work

Carl-Fredrik Sørensen, **Alf Inge Wang**, **Hien Nam Le**, **Heri Ramampiaro**, **Mads Nygård**, **and Reidar Conradi** [1]

Abstract
This paper describes a framework used to characterise mobile work in order to elicit functional and non-functional requirements for a mobile process support system. The framework is a tool for specifying and analysing mobile scenarios in detail, resulting in a characterisation of scenarios. This characterisation will indicate requirements to the software architecture and services the system should provide. In addition, the framework will indicate non-functional requirements like network capacity, network connectivity, security. To show the practical usage of the framework, we have applied the framework to a scenario describing a mobile researcher. As far as we know, there are no similar frameworks.

Keywords:
Mobile computing, mobile work, process support, software architecture.

[1]Dept. of Computer and Information Science, Norwegian University of Science and Technology (NTNU), N-7491 Trondheim, Norway. Phone: +47 73 594485, Fax: +47 73 594466, http://www.mowahs.com

9.1 Introduction

Mobile computing devices are now a part of everyday life. These devices offer possibilities for supporting both planned and unplanned mobile work. Mobile work does in this context mean work where people have to be at a specific location to reach their goal. The range of mobility can be everything from within a building to travelling world wide.

In January 2001, a Norwegian research project called MObile Work Across Heterogeneous Systems (MOWAHS) [MOW04], sponsored by the Norwegian Research Council, was initiated. The focus of this project is to investigate how to provide process support for mobile work using different kinds of equipment (ranging from a small mobile phone, or PDA to laptop/desktop PCs). An initial goal of the project is to find characteristics that can define and describe mobile work. In [RPM00], mobility is divided into physical and logical mobility. This classification is at a very coarse level, and was not very useful for classifying mobile work. From a more practical point of view, mobile work can be classified into hardware mobility, software mobility and combined mobility [WL01a]. Although this classification is more detailed, it focuses too much on what functionality the end-system should have. We therefore decided to look for characteristics that could be used to describe mobile work scenarios, rather than the end-systems to support such scenarios. To find these characteristics, we suggested different attributes that could be used to describe mobile work processes. We then divided the useful attributes into groups that resulted in the framework that is presented in section 9.3.

The aim of the MOWAHS characterisation framework is to produce requirements for an end-system supporting mobile computing. Such an end-system can be a process support system for mobile work consisting of a server or a set of servers, some mobile clients (laptops, PDAs, mobile-phones etc.), and human resources required to carry out a work process. The characteristics of the framework will give requirements to the architecture (topology, connectivity, network, type of client(s), type of server(s), reliability, security, performance, etc.), to services needed, and requirements for hardware to be used.

The rest of the paper is organised as follows: Section 9.2 relates our framework to other similar frameworks, while section 9.3 describes our MOWAHS characterisation framework for mobile work. Section 9.4 gives an example of how the framework can be applied to a scenario and the corresponding results, while section 9.5 concludes this paper and gives some indications for further work.

9.2 Related Work

Many research papers on mobile work have proposed a support system for specific mobile work scenarios, resulting in the development of tailor-made systems. Other papers have

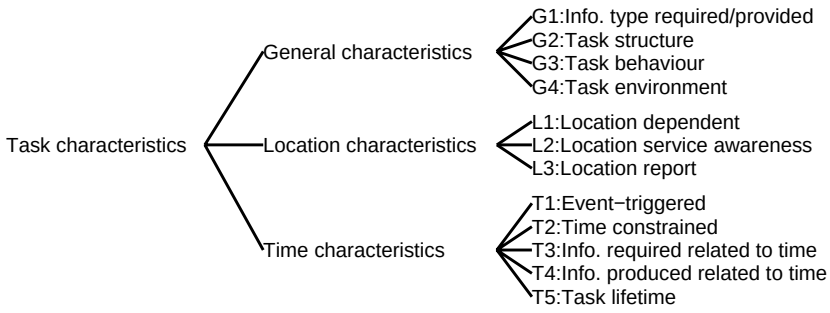

Figure 9.1: An overview of the MOWAHS characterisation framework for mobile work

attempted to give an overview of characteristics of mobile computing based on state-of-the-art technology.

Zimmerman [Zim99] suggests the ”MOBILE” framework to determine when mobile computing technology should be used to solve challenges related to mobility. This framework focuses on current technology and software development trends in mobile computing. Related common scenarios discussed include news reporting and hotel operations. The framework provides a useful overview of necessary support needed for specific mobile environments. However, the framework does not provide any guidelines for how to develop or design systems for mobile support.

Satyanarayanan [Sat96] identifies four constraints of mobile computing which mainly are concerned with limited resources, physical security (e.g hazards), communication and durability issues. Another approach is proposed by Forman and Zahorjan [FZ94] who examine three basic features of mobile computing including wireless communication, mobility and portability. These two approaches provide different ways of addressing mobility issues. The former focuses on connectivity issues, while the latter deals with Quality of Service (QoS), such as network bandwidth and device durability.

Rakotonirainy [Rak99] discusses current and future technology (e.g CORBA, mobile IP), adaptable to mobile computing environments. For this, he presents a scenario revealing the limitations of current technology. Although characteristics of mobile work can be derived from this approach, he does not provide a comprehensive framework for characterising mobile work environments.

The related work mentioned above mainly focuses on the technical parts of mobility including mobile hardware, limitations, and benefits. Our framework focuses on the mobile work itself and tries to derive the functionality, architecture and hardware required to support specific mobile scenarios. In a real mobile work situation there are more factors than mobile technology to consider. Examples are co-operative work and time and location constraints of the tasks themselves. In addition, our framework can be used to divide mobile scenarios into groups with similar characteristics.

The MOWAHS characterisation framework has not a formal basis for describing mobility. In [WR96], the notation UNITY is extended to formally describe mobility. The goal of the extension is to establish a formal specification and design technique that can accommodate the concepts of place, time, and action in a manner consistent with the design requirements for mobile systems.

9.3 The MOWAHS Characterisation Framework

Figure 9.1 shows an overview of our framework illustrating how it can describe a task of a mobile scenario through several attributes. The task characteristics are divided into three main categories: general, location and time. From the framework, we can derive both non-functional and functional requirements for mobile computing. We believe that the usage of such a framework will allow us to explore typical classes of mobile work with different process and transaction support. Further, the ability to identify similarities among different scenarios, allow us to reuse software systems. These systems should not only support one specific scenario, but also groups of related scenarios. However, we do not claim that the MOWAHS characterisation framework for mobile work covers all possible attributes used to describe mobile work.

To evaluate the different characteristics in the framework, we use an ordinal scale (1-5). Higher values indicate more complexity in terms of system requirements, while lower values indicate none or less complexity. Some of the characteristics do not use the full scale, only the values 1, 3 and 5 to get a uniform representation of extreme values.

The framework should be applied to a mobile scenario focusing on one role and one task at a time. The tasks will usually vary in importance for a specific user. In our framework, each task must be assigned a weight on a scale 1-5 (very low, low, medium, high, very high). After analysing several tasks, the mobile process characteristics can be determined based on the weights of importance for each task. The mobile process characteristics are established by calculating the mean value of each characteristic for every task and role. In addition, a *complexity indicator* is calculated for the whole scenario. The complexity indicator is the mean value of all process characteristics. Note that the process characteristics are not statistical values, but should be used as an indicator of complexity in a mobile support system. Below, we indicate the process steps for applying our framework.

```
1. Select the mobile scenario
2. Identify different roles in the scenario
3. For each role; identify tasks and responsibilities
4. For each task:
   a) Assign task weight
   b) Characterise the task using the characterisation framework
5. Apply the process characteristics:
   a) Calculate weighted mean values for each characteristic
   b) Calculate an overall scenario complexity indicator
6. Derive system requirements and priorities from the process
        characteristics
```

The rest of this section describes the framework in more detail.

9.3.1 General Characteristics

The general characteristics are used to describe task structures and attributes that are indirectly related to mobility. The words in parenthesis describe the possible answers and measurements of each characteristic element.

G1 **Information type required/provided** *(1 None, 2 Text, 3 Audio, 4 Picture, 5 Video)*
G1 is used to establish the type of information needed to perform a task. The information involved can be required/provided by the system, by people, or in combination. This property is used to decide the level of QoS for communication, and the client capability. In addition it specifies the media that the system must support.

G2 **Task structure**
This item describes how tasks are organised and how they are related:

G2.1 **Decomposable** *(1 No, 3 Uncertain, 5 Yes)*
G2.1 is used to decide whether the task is composed of sub-tasks or not. This can be used to decide if the end-system needs to support hierarchical compositions of tasks.

G2.2 **Part of sequence** *(1 No, 3 Partial, 5 Yes)*
G2.2 specifies if a task has order dependencies with respect to other tasks. This will decide if the end-system needs specification of execution order (typically a state-machine).

G2.3 **Collaboration type** *(NA[2], Client/server, Peer-to-peer, Broadcasting, Ad-hoc)*
G2.3 is used to specify the type of collaboration among task executors. This

[2] *NA = Not Applicable*

property can be used to determine system architecture/infrastructure, system coordination, and collaborative system services.

G2.4 **Cooperation with other task(s)** *(1 No, 3 Partial, 5 Yes)*
G2.4 defines whether a task must be coordinated with other concurrent tasks during execution or not. This can be used to specify the needs for cooperation and coordination among tasks, and needs for cooperative transactions.

G3 **Task behaviour**
This item describes how a task is created and terminated:

G3.1 **Pre-planned** *(1 No, 3 Partial, 5 Yes)*
G3.1 describes to what degree a task is planned in beforehand. This can be used to decide the needs for communication and back-end support, and support for on-the-fly planning and creation of tasks. If a task is pre-planned, information required for a specific task can be provided in beforehand on a mobile device.

G3.2 **Accomplishable** *(1 No, 2 Low, 3 Medium, 4 High, 5 Yes)*
G3.2 is used to define how much of a task that usually can be completed, and this will decide the needs for transaction support in the end-system and how the end-system must handle exceptional cases, and possible delays of termination.

G4 **Task environment**
This item describes demands on the task environment.

G4.1 **Security demands** *(1 None, 2 Low, 3 Medium, 4 High, 5 Very high)*
G4.1 specifies if the task requires a secure environment for system communication and execution (e.g. secure information handling on client, in the communication channel and on server). This can specify the level of protection against malicious access, virus attacks, theft, and denial of service.

G4.2 **Secrecy demands** *(1 None, 2 Restricted, 3 Confidential, 4 Secret, 5 Top secret)*
G4.2 specifies the level of protection from external insight to the definition and execution of a task and task environments (e.g. military operations, confidential information, privacy). This property can be used to decide the level of data encryption, access to the servers and clients, and protected communication.

G4.3 **Reliability demands** *(1 None, 2 Low, 3 Medium, 4 High, 5 Very high)*
G4.3 defines demands for stable execution of tasks (e.g. for critical tasks like establish or retain connection to emergency systems).

9.3.2 Location characteristics

The location characteristics are used to specify if a task or back-end system support depend on geographical locations. The following items are used to describe location characteristics:

L1 **Location dependent** *(1 No, 3 Partial, 5 Yes)*
L1 describes to what degree a task must be executed at a specific location. This can be used by the end-system to decide what services are needed for the task execution at a specific location. This may also be used to decide what physical equipment is required to perform the task.

L2 **Location service awareness** *(1 No, 3 Partial, 5 Yes)*
L2 specifies if a task can exploit services provided at the location. The task may dependent on services provided (e.g. printer, fax, etc.) to be executed or terminated. L2 can be used to determine the location dependent services that are needed in the end-system.

L3 **Location report** *(1 No, 3 Partial, 5 Yes)*
L3 identifies if a task must report its location to the system (e.g. by using GPS, GSM etc.). This can be used to decide the mobile equipment and services needed, and if a location reporting service should be part of the mobile client.

9.3.3 Time characteristics

The time characteristics are used to investigate the temporal properties of a task, and the system support for these properties. The creation and execution of tasks may require timely coordination between an end-system and the task executor. These properties (excluding task lifetime) can be used to determine the required connectivity (communication) between the end-system and mobile clients. All the characteristics may imply requirements for extended transaction support (e.g. synchronised information flow, postponed commitments etc.). The following items are used to describe time characteristics:

T1 **Event-triggered** *(1 No, 3 Partial, 5 Yes)*
T1 is used to decide whether a task is triggered by an event or not. This property can be used to imply the need for instant access to information resources and services. The property may also require a re-scheduling and/or re-definition of existing tasks.

T2 **Time constrained** *(1 No, 3 Partial, 5 Yes)*
T2 is used to describe if a task must be executed at a specific time or within a specific timespan. This property can be used to decide how the system should be

synchronised. A time constrained task may also demand support for re-scheduling of tasks and if a task is achievable or not, related to time constraints.

T3 **Information required related to time** *(1 NA, 2 Low, 3 Medium, 4 High, 5 Real-time)*
T3 describes how important the information required to execute the task is related to time. This property can be used to determine the connectivity and the bandwidth needed to transport data from a server to a mobile client.

T4 **Information produced related to time** *(1 NA, 2 Low, 3 Medium, 4 High, 5 Real-time)*
T4 describes how important the information produced by the task is related to time. This property can be used to determine the connectivity and bandwidth needed to transport data from a mobile client to a server, or other clients.

T5 **Task lifetime** *(1 Short, 3 Medium, 5 Long)*
T5 is used to describe the expected lifetime of a task. For example, short may mean less than 1 hour, medium between 1 hour and a working day (8 hours), and long more than one working day. This scale can be adjusted or changed in respect to the specific scenario. The task lifetime property can be used to determine the connectivity (on-line/off-line), the type of transaction support needed (e.g. long, nested transactions with a need for relaxed ACID[3] properties.), and the response time of the system.

9.4 The Framework Applied to a Scenario

This section presents the results from applying our framework to a mobile work scenario.

9.4.1 Mobile Researcher Scenario

The scenario we have used as a test case for our framework is a mobile researcher participating at a conference. We have not only considered the current practice, but also included possible technological enhancements to existing tasks. Each task has been assigned a weight of importance (1-5).

The mobile researcher typically performs these tasks:

S1 **Prepare a presentation** which involves using available information (e.g. the paper to be presented and some addition background information) to produce a presentation that will be given at a conference. *Importance: 4 High*.

[3] ACID stands for Atomicity, Consistency, Isolation and Durability

Char.	S1	S2	S3	S4	S5	S6
Weight	4 High	5 Very high	3 Medium	1 Very low	2 Low	3 Medium
G1	4 Picture	4 Picture	5 Video	4 Picture	4 Picture	1 None
G2.1	3 Uncertain	1 No	1 No	2 Uncertain	3 Yes	1 No
G2.2	5 Yes	5 Yes	1 No	1 No	1 No	1 No
G2.3	NA	NA	Peer-to-Peer	NA	Peer-to-Peer	NA
G2.4	1 No	1 No	5 Yes	1 No	5 Yes	1 No
G3.1	5 Yes	5 Yes	3 Partial	1 No	3 Partial	3 Partial
G3.2	5 Yes	5 Yes	4 High	4 High	4 High	5 Yes
G4.1	2 Low	2 Low	1 None	1 None	3 Medium	1 None
G4.2	1 None	1 None	1 None	1 None	2 Restricted	1 None
G4.3	2 Low	4 High	3 Medium	2 Low	3 Medium	1 None
L1	1 No	5 Yes	5 Yes	1 No	1 No	5 Yes
L2	1 No	1 No	1 No	1 No	1 No	1 No
L3	1 No	5 Yes	1 No	1 No	1 No	3 Partial
T1	1 No	1 No	1 No	1 No	1 No	1 No
T2	3 Partial	5 Yes	5 Yes	1 No	3 Partial	3 Partial
T3	3 Medium	4 High	1 NA	1 NA	2 Low	1 NA
T4	3 Medium	1 NA	5 Real-time	1 NA	2 Low	1 NA
T5	3 Medium	1 Short	3 Medium	5 Long	5 Long	1 Short

Table 9.1: The framework applied to the scenario

S2 **Perform a presentation** which involves talking to an audience in a specific room using available equipment to aid the presentation. *Importance: 5 Very high*.

S3 **Listen to a presentation** which involves being in a specific room at a specific time listening to and making notes about the presentation. The notes can be written on paper, a laptop, a PDA, or other suitable devices. This task should also record audio or video, that may be transmitted (using a network connection) to colleagues not being at the conference. *Importance: 3 Medium*.

S4 **Read a paper/book** which involves reading some text written on paper, presented on a laptop, presented on a PDA or on another device, and making notes of interesting parts of the text. This task can possibly involve downloading the paper/book from a server. *Importance: 1 Very low*.

S5 **Write a scientific paper** which involves writing about an idea, results and/or some specific experiences. This task requires having sufficient information available (e.g. papers describing related work, technical reports, etc.) that should be a part of the

paper. To write the paper, the researcher can use pen and paper, a laptop, a PDA or another device. The task might also include cooperation, and coordination with other authors of the paper. *Importance: 2 Low*.

S6 **Talk to conference participants** which involves meeting people at a specific place at a specific time. Talks can result in new ideas (that should be written down on napkins, PDAs, mobile phones etc.) or trading of business cards (either using electronic devices or paper cards). *Importance: 3 Medium*.

9.4.2 The Framework Applied to all Tasks

The results of applying the framework to each task of the scenario (S1-S6) is shown in table 9.1. The task characteristics in the table are only denoted by a letter and a number (see section 9.3 for a full description).

Table 9.1 shows that the three tasks S1, S4 and S5 are not necessarily mobile, since they are location independent (L1). S2, S3 and S6 require the task executor to move to a specific room (conference room or similar) to reach the goal of the task. Further, the table shows that the two tasks S3 and S5 involve interaction with other tasks and task executors (G2.4). If we consider the demands for these tasks (S3 and S5) to information required/produced related to time, we discover that that S3 has the highest resource demands (T4). In addition, S3 is also location dependent. This means that S3 is hardest to support in terms of both software and hardware (including network).

9.4.3 Summary of Process Characteristics

Table 9.2 shows a summary of the task characteristics where the tasks are seen as a part of a process. All results except G2.3 in the table show mean values of the six tasks (S1-S6).

Table 9.2 shows that the six tasks in the scenario are using different media types (G1), and are partially organised and pre-planned. This implies that mobile clients do not need to have a permanent network connection, but may work asynchronously. This again means that the mobile clients must carry data required to perform the tasks as well as data created or manipulated by the task. The calculated complexity indicator for this scenario is 2.50, indicating a medium to low complexity of the mobile process support system.

9.4.4 System Design Discussions

From table 9.2 we can deduce the system support needed to implement a mobile process support system for this scenario. The end-system should provide:

Characteristics	Result	Comment
G1 Info.type req/provided	3.66 Picture	Audio+Video needed for S3
G2.1 Decomposable	3.00 Uncertain	Only S5 are fully decomposable
G2.2 Part of sequence	2.00 No/Partial	
G2.3 Collaboration type	Peer-to-Peer	Required in S3 and S5 if several authors
G2.4 Coop. with other task	2.11 Partial	Required in S3 and S5 if several authors
G3.1 Pre-planned	3.89 Partial	Most of the tasks are (partially) pre-planned
G3.2 Accomplishable	4.67 Yes	S3, S4 and S5 might not complete
G4.1 Security demands	1.72 Low	S5 required most security
G4.2 Secrecy demands	1.16 None	S5 requires **restricted** secrecy
G4.3 Reliability demands	2.56 Medium	S2 requires high reliability
L1 Location dependent	3.44 Partial	S2, S3, and S6 are location dependent
L2 Location service awareness	1.00 No	
L3 Location reporting	2.44 Partial	In S2, must notify when presentation begin
T1 Event-triggered	1.00 No	
T2 Time constraints	3.78 Partial	S2 and S3 must follow conference agenda
T3 Info. required related to time	2.39 Low	S2 has high requirements
T4 Info. produced related to time	2.22 Low	S3 has real-time requirements
T5 Task lifetime	2.44 Medium	The task lifetime range from short to long

Table 9.2: Summary of the Process Characteristics

- Support for handling the media types text, pictures, and audio/video. This includes functionality to create, store, present, and transmit the corresponding media.

- Support for decomposing tasks, sequencing tasks, and for allowing cooperative tasks. This means that the system should provide functionality to plan, organise and execute individual and group tasks.

- Support for peer-to-peer collaboration between task executors. This means that the system should provide an infrastructure for transferring data directly between clients of the system. The network should have capacity to transfer audio or video streams between clients on-line. Other kind of information could be stored on-demand at the clients for later batch transmission.

- Specific support for security, secrecy and reliability is not considered as important for the end-system. We assume that the available network and operating system will provide sufficient support.

- Support for scheduling and execution of tasks at specific times.

From table 9.1 we know that the tasks S2, S3 and S6 are location dependent. These tasks must be carried out in specific places to meet people. However, the execution of these tasks does not require any software location services from the end-system.

Further we can deduce the client requirements (both hardware and software):

- Audio/video recording, storing and manipulation (e.g. buffering, filtering, scaling, compression, formatting).
- Presentation of multi-media.
- A fast and reliable network connection that is wireless or fixed (depends on the conference room facilities).

Because of the multi-media requirements mentioned above (task S3), the mobile client in our scenario must reside on a high capacity computer (e.g. a laptop with recording capacity). If the audio/video transmission of S3 is not needed, it is sufficient to use a low capacity computer as host for the mobile client (e.g. a PDA). For the receiving host of multi-media information, a PDA can be sufficient (current technology). The PDA must then support multi-media and network connectivity.

In the development of the mobile process support system, the weights of the tasks as presented in table 9.1 can be used as a starting-point for a project plan. Each task can form a Use Case, and the weights determine the importance of the Use Cases [JBR99].

9.5 Conclusions and Further Work

In this paper we have presented a framework to characterise mobile work. We have shown that this framework can be applied to derive software and hardware requirements for a mobile process support system.

To get more experiences with and to validate our framework, we will consider these options:

- Try the framework on more scenarios to improve/extend the framework to cover mobile work in as many areas as possible.
- Try the framework on already implemented systems, to compare the requirements (both functional and non-functional) that are derived from the MOWAHS framework against implemented solutions. This may validate the framework and improve the outcome from applying the framework.

- Establish a prototype system based on the proposals given from applying the framework. The prototype should support mobile work processes in as many scenarios as possible.
- Try out the prototype in real cases, preferably in scenarios where stakeholders may have benefit from using such a system. The prototype should be tried on a range of mobile devices, both in on-line and off-line modes.
- Establish interfaces to other back-end support and legacy systems for mobile work.
- Establish transaction models that can provide special support required in mobile process support systems.

Acknowledgement

This paper is part of the MObile Work Across Heterogeneous System (MOWAHS) project. The MOWAHS project is sponsored by the Norwegian Research Council's IKT 2010 programme.

CHAPTER 10

P3 – Requirement Indicators for Mobile Work: The MOWAHS Approach

Heri Ramampiaro, Alf Inge Wang, Carl-Fredrik Sørensen, Hien Nam Le and Mads Nygård

Abstract
This paper describes eight requirement indicators derived from a mobile work characterisation framework (MWCF). The MWCF is a framework for analysing mobile scenarios (mobile tasks) in order to implement a system to support such scenarios. The requirement indicators are computed from the score of the MWCF. These are used to reveal the complexity of the different parts of a final mobile support system (software and hardware). Further, these indicators can be a help to prioritise the non-functional and functional requirements of the end-system. We have identified the following indicators: General Task Indicator, Information Complexity Indicator, Location Complexity Indicator, Time Complexity Indicator, Network Connectivity Indicator, Network Speed Indicator, Energy Consumption Indicator, Transactional Support Indicator and Mobility Indicator. In this paper, we apply these indicators on four different scenarios and discuss how they can be used in practice.

Keywords:
Requirement engineering, mobile requirements, mobile process system.

10.1 Introduction

The proliferation of network and wireless technology has increasingly influenced our way of performing work. People carry out their work while they are on move - e.g., on their way to work, on a trip or even while they are on vacation. Also, several companies have started to work as virtual organisations, where people collaborate over several locations and time zones. This means that these people are distributed, collaborative, mobile and require asynchronous technology that enables sharing of documents and work plans. However, the infrastructure and tools for carrying out projects in virtual organisations are still immature. We must deal with a heterogeneity of tools, equipment (laptops, PDAs, mobile phones) and work models. In addition, the mobility of devices and the partial lack of connectivity require regular synchronisation of such devices against stationary PCs (including servers). Hence, there are several challenges that must be faced.

Our MOWAHS framework[SWL+02] is aimed at revealing these challenges. It presents relevant characteristic elements of mobile work. From such characteristics, there is a gap between characteristics and requirements that must be bridged in order to provide appropriate technology that supports a specific type of mobile work, and defining the actual requirements. To deal with this, we have identified several indicators that are useful in defining the requirements working as a guideline for designating the needed technology. This paper presents these indicators and discusses their usefulness.

The rest of this paper is organised as follows. To put work in perspective, Section 10.2 discusses the relation of our work to existing frameworks and approaches. Section 10.3 summarises our mobile work characterisation framework based on [SWL+02]. Section 10.4 describes the requirement indicators that we have identified based on the mobile work characteristics. The use of these indicators are illustrated in more detail in Section 10.5. Finally, in Section 10.6 we discuss and conclude our paper.

10.2 Related Work

Some research papers on mobile work (see below) have proposed systems to support specific mobile work scenarios, which has resulted in the development of tailor-made systems. In addition, there are papers that give an overview of characteristics of mobile computing focusing on state-of-the-art technology, and there are a few research papers that treat requirement engineering related to the development of applications for mobile environments. However, it has been difficult to find other work that can directly be compared to ours. Hence, to our knowledge, the approach presented in this paper is unique in that it is based on indicators derived from a specialised characterisation framework for mobile work.

Mead [Mea00] and Leite [dPL00] pointed out that there is a gap between research and practice in requirement engineering. In particular, they point out the lack of tools for new methods and technical support in requirement engineering. Further, Mead stresses the necessity of improving the management of visibility and control. However, although most of the ideas of the above papers are useful and can be applied to problems such as ours, none of them directly deals with mobile environments.

Maccari [Mac99] presents an example of requirement engineering challenges in the mobile telephones industry due to the complexity of mobile phone architecture and performance. In addition, requirement engineering in terms of mobile telephones, has to cope with many different factors such as protocols and technology standards. Also, the limitation of wireless devices, such as network connectivity and network speed, yields important challenges that developers have to deal with. The author argues that requirement engineering is a collaborative task.

Hammond et al. [HRH01] presents interesting scenarios such as air and railway traffic control, which reveal risks in requirement engineering. The authors argue that understanding the operating environment and application domain is important especially in system integration. Although, many of these arguments are valid, the authors only focus on the relationship and traceability among subsystem properties, regardless of the mobility properties of these subsystems. Note that, in mobile environments, mobility properties have a crucial impact on the system development.

Zimmerman [Zim99] suggests the ”MOBILE” framework to determine when mobile computing technology should be used to solve challenges related to mobility. This framework focuses on the current technology and software development trends in mobile computing. Further, common related scenarios are discussed, including news reporting and hotel operations. The framework provides a useful overview of necessary support needed for specific mobile environments. However, the framework does not provide any guidelines for how to develop or design systems for mobile support.

Satyanarayanan [Sat96] identifies four constraints of mobile computing which mainly are concerned with limited resources, physical security (e.g. hazards), and communication and durability issues. Another approach is proposed by Forman and Zahorjan [FZ94] who examine three basic features of mobile computing including wireless communication, mobility and portability. These two approaches provide different ways of addressing mobility issues. The former focuses on connectivity issues, while the latter deals with Quality of Service (QoS), such as network bandwidth and device durability.

Rakotonirainy [Rak99] discusses current and future technologies (e.g. CORBA and mobile IP), adaptable to mobile computing environments. For this, he presents a scenario revealing the limitations of the current technology. Although characteristics of mobile work can be derived from this approach, he does not provide a comprehensive framework for characterising mobile work environments.

The related work mentioned above mainly focuses on the technical parts of mobility including mobile hardware, limitations, and benefits. Our framework focuses on the mobile work itself and tries to derive the functionality, architecture and hardware required to support specific mobile scenarios. In a real mobile work situation there are more factors than mobile technology to consider. Examples are co-operative work and time and location constraints of the tasks themselves. In addition, our framework can be used to divide mobile scenarios into groups with similar characteristics.

The MOWAHS characterisation framework has not a formal basis to describe mobility. In [WR96], the notation UNITY is extended to formally describe mobility. The goal of the extension is to establish a formal specification and design technique that can accommodate the concepts of place, time, and action in a manner consistent with the design requirements for mobile systems.

10.3 The MOWAHS Characterisation Framework for Mobile Work

The MOWAHS characterisation framework is aimed at helping to specify requirements for an end-system supporting mobile computing. Such an end-system can be a process support system for mobile work consisting of a server or a set of servers, some mobile clients (laptops, PDAs, mobile-phones etc.), and human resources required to carry out a work process. We believe the characteristics elements of our framework are comprehensive enough to indicate the requirements that are relevant in terms of necessary and/or appropriate architecture (topology, connectivity, network, type of client(s), type of server(s), reliability, security, performance, etc.), services needed, and requirements for hardware to be used.

The task characteristics are divided into four main categories: *general* –install used to describe task structures and attributes that are indirectly related to mobility, *information* – used to specify the information requirements that must be fulfilled to support a task, *location* – used to specify if a task or back-end system support depends on geographical locations, and *time* – used to investigate the temporal properties of a task, and the system support for these properties. Table 10.1 summarises our characteristics for mobile work. From the framework, we can derive both non-functional and functional requirements for mobile work. We believe that the usage of such a framework enables us to explore typical classes of mobile work with different processes and transactional support.

In [SWL+02], we have outlined the steps necessary to effectively apply our characterisation framework. These steps are as follows:

1. Select the mobile scenario
2. Identify the different roles in the scenario
3. For each role; identify tasks
4. For each task:

    a) Write a task description using the task description template
    b) Assign task weight
    c) Characterise the task using the characterisation framework

5. Apply the process characteristics:

    a) Calculate weighted mean values for each characteristic
    b) Calculate an overall scenario complexity indicator

6. Derive system requirements and priorities from the process characteristics

To measure the different characteristics in the framework, we use an ordinal scale (1-5). High values indicate more complexity in terms of system requirements, while low values indicate low complexity. Some of the characteristics do not use the full scale, but only the values 1, 3 and 5 to get a uniform representation of extreme values.

The framework should be applied to a mobile scenario focusing on one role and one task at a time. The importance of a specific task will usually vary depending on the user. This means that a task may be important for a specific user, while it is less important for another.

To define this importance, in our framework, each task is assigned a weight on a scale 1-5 (very low, low, medium, high, very high). After analysing several tasks, the mobile process characteristics can be determined based on the weights of each task. The mobile process characteristics are established by calculating the mean value of each characteristic for every task.

10.4 Requirement Indicators

From the scores of our framework we can compute different indicators that can help us analysing the mobile scenarios and help us to prioritise and extract non-functional and functional requirements. The four first indicators are characteristics computed from the four groups in our framework (see Section 10.4.1: *General, information, location,* and *time.* The other five indicators are computed by combining characteristics from different groups of the framework (see Section 10.4.2. These indicators are computed based on some key and additional characteristics. The additional characteristics are not relevant, if the average of the key characteristics have a score more than 3. A summary of all the indicators can be found in Table 10.2.

To be able to see the complexity of a whole scenario, we can compute the *Overall scenario complexity indicator*. This indicator is the average of all characteristics of our framework.

Characteristic	Possible values	Description
G1 Decomposable	*(1 No, 3 Uncertain, 5 Yes)*	Decides whether the task is composed of sub-tasks or not.
G2 Part of sequence	*(1 No, 3 Partial, 5 Yes)*	Specifies if a task has order dependencies with respect to other tasks.
G3 Pre-planned	*(1 Planned, 3 Partial, 5 Ad-hoc)*	Describes to what degree a task is planned in beforehand.
G4 Data synchronisation	*(1 No, 3 After task, 5 Within task)*	Specifies when a task has to synchronise/merge updated data with other tasks.
G5 Data exchange rate	*(1 None, 3 Once, 5 Many)*	Specifies the rate of data exchange between the current task and other tasks (during its lifetime).
I1 Information contents	*(1 NA, 2 Text, 3 Graphics, 4 Audio, 5 Video)*	Describes the complexity of the information required or produced by the task.
I2 Information streaming	*(1 NA, 3 Discrete, 5 Continuous)*	Describes whether the task requires streaming of data or not.
I3 Information required related to time	*(1 NA, 2 Low, 3 Medium, 4 High, 5 Real-time)*	Describes how important the information required to execute the task is related to time.
I4 Information produced related to time	*(1 NA, 2 Low, 3 Medium, 4 High, 5 Real-time)*	Describes how important the information produced by the task is related to time.
I5 Information transmission speed	*(1 NA, 2 Slow, 3 Medium, 4 Fast, 5 Very fast)*	Describing the expected transmission speed of information used or produced by the task.
L1 Location dependent	*(1 No, 3 Partial, 5 Yes)*	Describes to what degree a task must be executed at a specific location (e.g. that a lecturer must be in a classroom to teach students).
L2 Require services at location	*(1 No, 3 Partial, 5 Yes)*	Specifies if a task needs electronic services available at the location such as printers, network connections, video projectors, fax machines, etc.
L3 Produce services at location	*(1 No, 3 Partial, 5 Yes)*	Specifies if a task produces electronic services at the location that can be used by others at this location, such as information queries, beaming of information, network connectivity etc.
L4 Location report	*(1 No, 3 Partial, 5 Yes)*	Specifies if a task must report its location to the system.
L5 Route constraints	*(1 No, 3 Partial, 5 Yes)*	Specifies if a mobile task must follow a specific route or not when moving around.
T1 Event-triggered	*(1 No, 3 Partial, 5 Yes)*	Decides whether a task is triggered by an event or not.
T2 Time constraint	*(1 No, 3 Partial, 5 Yes)*	Describes if a task must be executed at a specific time or within a specific time-span (e.g. a task must be performed between 10:15 and 12:00).
T3 Temporal coordination	*(1 No, 3 Partial, 5 Yes)*	Describes if a task must be coordinated with other tasks (e.g. different military units must strike a the same time).
T4 Task resumption	*(1 No, 3 Partial, 5 Yes)*	Describes if a task can halt, and then later resume from where it left off (not requiring a complete restart of task).
T5 Task lifetime	*(1 Seconds, 2 Minutes, 3 Hours, 4 Days, 5 Weeks)*	Describes the expected lifetime of a task.

Table 10.1: The MOWAHS characteristics for mobile work

Indicator	Key Characteristics	Additional Characteristics
General Task Indicator (GTI)	G1 – G5	NONE
Information Complexity Indicator (ICI)	I1 – I5	NONE
Location Complexity Indicator (LCI)	L1 – L5	NONE
Time Complexity Indicator (TCI)	T1 – T5	NONE
Network Connectivity Indicator (NCI)	G3 – G5, L4, T1	T4
Network Speed Indicator (NSI)	I3 – I5	G5, I2
Energy Consumption Indicator (ECI)	L1 – L3, T5	G5, I1, I2, L4
Transactional Support Indicator (TSI)	G3 – G5, T1, T4, T5	NONE
Mobility Indicator (MI)	L4, L5	L1, L3, T3

Table 10.2: Summary of the indicators

10.4.1 Directly computed indicators

The following are indicators that are computed from how the mobile work characteristics are grouped in the MOWAHS framework (see Table 10.1):

- **General Task Indicator (GTI)** is an average of all general characteristics (G1-G5), constituting its key characteristics. A GTI score indicates the complexity of the analysed task. A high score may, for instance, mean that an underlying process and transaction system must be able to support complex – i.e., flexible and advanced – tasks. Further, this indicator specifies the basic functionality required to handle the process (tasks) and data exchange. It also determines correspondingly the complexity of the required tools (general support) – e.g., workflow tools.

- **Information Complexity Indicator (ICI)** is an average of all information characteristics (I1-I5), which constitute its key characteristics. A high ICI score indicates that the end-system must cope with complex information presentation, management, and transmission. It thus determines the complexity of information management and information exchange mechanisms, including the *quality of service*. The ICI can also be used to select the suitable mobile host device (software and hardware) and the appropriate server (software and hardware).

- **Location Complexity Indicator (LCI)** is an average of all location characteristics (L1-L5), constituting its key characteristics. A high LCI score indicates that the end-system should possibly include GIS-support and should put high requirements on the type of equipment that can be used as mobile clients. In addition, it tells how the location affects the actual task performance. It also determines the required portability of the relevant mobile devices – i.e., the weight, the size, etc. A high score also indicates that some GIS-support should be supported to define the location of the task and the need for representing location in the process modelling language. Finally, it can determine the need for the system to be aware of location.

This means that some data may be relevant at a specific location, other data may not, etc.

- **Time Complexity Indicator (TCI)** is an average of all time characteristics (T1-T5), which constitute its key characteristics. A high TCI score means that management of tasks must be timed, and that it may require more advanced transactional support. It also determines how time affects the actual task performance. Further, the TCI score indicates the (non-functional) system performance and system availability. Moreover, it specifies the required (functional) task scheduling mechanism and task synchronisation or coordination.

10.4.2 Derived indicators

We have identified some indicators that we find useful in order to specify the requirements (both functional and non-functional) for mobile work support. The following are indicators that are derived by combining some specific characteristics from different groups. They are useful in identifying the degree of complexity of both the mobile clients and the service providers, including hardware and software issues.

- **Network Connectivity Indicator (NCI)** indicates the need for being online. The key characteristics for this indicator are G3, G4, G5, L4 and T1. In addition, T4 may be used to derive NCI. A high NCI score means that the mobile client must be online most of the time. It thus determines the required networking capabilities for the actual mobile host. Also it specifies non-functional requirements for the system such as reliability and latency.

- **Network Speed Indicator (NSI)** specifies the needed transmission speed between a mobile client and the end system. In other words, a high NSI score means that the transmission speed must be high, while a low score tells that even low transmission speed may work. The key characteristics for this indicator is I3, I4 and I5, while G5 and I2 may be needed as supplements. From above, we may conclude that the NSI would be useful to set some non-functional requirements that are relevant, such as performance, quality of service and latency. It also indicates the kind of network technology that must be available for use.

- **Energy Consumption Indicator (ECI)** describes the expected required battery lifetime on mobile device. Its key characteristics are L1, L2, L3 and T5. In addition, G5, I1, I2 and L4 can be used to derive this indicator.

  A high ECI score means that the mobile device requires supplies that are able to deliver high energy, while a low score signifies low energy consumption. Based on this, the ECI indicates the energy technology needed that should be provided or

selected, considering portability versus energy cost. In other words, this indicator helps designating appropriate energy supplies in accordance with the actual energy consumption. It also indicates the required and suitable mobile device hardware. In addition, this indicator implicitly shows how long the task should be executed – i.e., task lifetime, which often corresponds to the actual energy consumption.

High-speed networks and high performance multimedia applications or hardware often has high energy consumption. Therefore, tasks requiring such types of equipment may have to consider the necessity to provide long-lasting energy supplies. In our framework, this can be specified in I1, I2 and I5. Moreover, the available energy supplies may affect task executions. Sometimes, it may be necessary to reschedule a task if the power supplied is lower than that required. For example, delegation or postponing of tasks may be useful to deal with this.

- **Transactional Support Indicator (TSI)** describes the need for flexible/advanced transactional support, and is primarily an average of the following characteristics: G3, G4, G5, T1, T4 and T5. A high TSI score indicates that the transactional support must be flexible and advanced; while a low score may indicate that traditional transactional support (ACID transactions) is sufficient.

  Hence, the TSI is useful in determining the suitable transactional support to be provided. It mainly indicates how advanced the transactional support should be in performing tasks. In effect, the TSI will help us identify the suitable transaction models or transactional frameworks, including transactional tools and applications that have to be provided. High TSI scores require advanced transactional models [ELMB92, Moh94, JK97] or even customisable transaction models [RN02]. This means that we may have to relax the ACID properties to allow the execution of a specific task or preserve these properties to ensure correct executions and consistent results. Note that in some cases we may be forced to allow temporal inconsistencies to be able to execute a task. Thus, we have to allow the transactional support to be non-ACID. How this can be achieved, has been shown in [RN02]. Further, as cooperation between tasks may be necessary, transactions must be able to cooperate, and the necessary flexibility may require customisable transaction models.

- **Mobility Indicator (MI)** describes how much mobility is involved in task execution. Its key characteristics are L4 and L5, but in addition, it may be derived from L1, L3 and T3. A mobility indicator can thus be found by computing the average of the characteristics L1, L3, L4, L5 and T3. Here, L1, L3, L4 and L5 are directly related to location and mobility of a task, while T3 implicitly affects the mobility of tasks. For example, if the value of T3 is 3 or 5 – i.e., tasks have temporally to be coordinated; and this may require changing the involved task locations. This means that a task may have to be executed at a specific location in order for it to be coordinated with another task.

  A high average score of the above characteristics indicates that a task is highly

mobile, while a low score tells us that a task may not be mobile or has low mobility. It is important to note that the value of this indicator will perhaps be affected by the value of other characteristics. More specifically, location dependency (L1) may be triggered by the available transmission speed (I5). Further, information produced at a specific location (L3) may relate to the information required related to time (I3) or the information produced related to time (I4). In this respect, MI is indirectly affected by I3, I4, and I5.

Intuitively, this indicator is useful in determining the complexity of a task based. It thus affects the type of equipments (devices) and tools that are necessary to accomplish a task. From this perspective, the MI may indicate that some specific mobile devices are necessary to perform a task. In addition it may specify appropriate tools (e.g., WAP, Lightweight Java, etc.). In this view, it may also be used to determine relevant technology that is applicable, such as network, mobile device and mobile computer technology.

10.5 Indicators applied to various scenarios

In this section we will show the results from analysing various mobile scenarios using the indicators described in section 10.4.

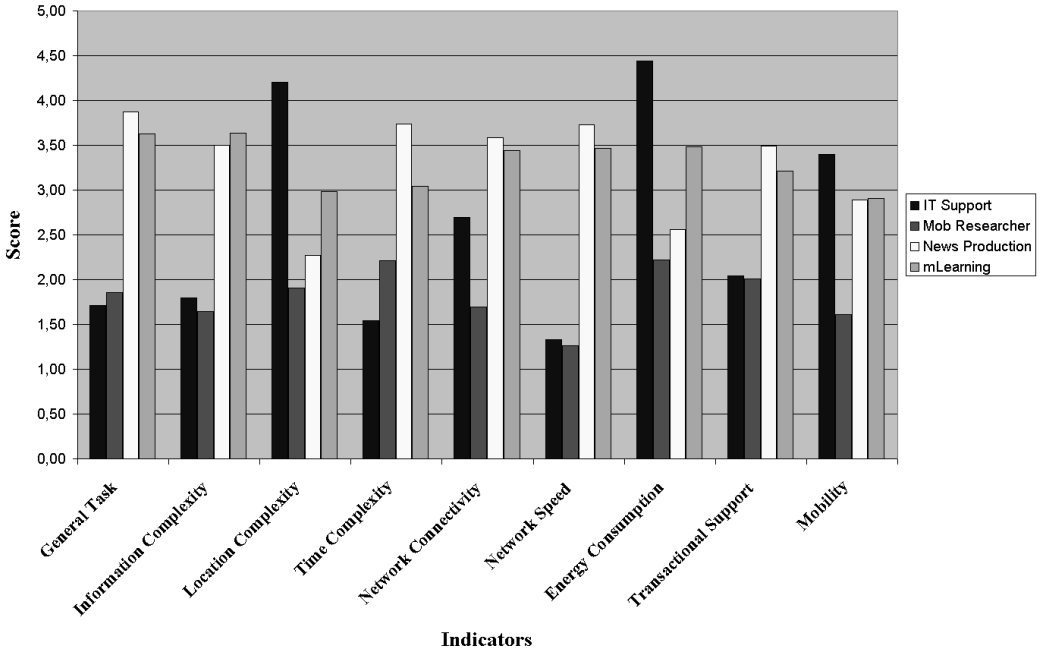

Figure 10.1: The results of applying the indicators to the scenarios

Here is a short description of the four scenarios we have analysed:

1. **IT support scenario:** IT support people handles inquiries from users about hardware and software, and install new of software and hardware. To perform their work they have to move around to specific locations (like offices and computer labs) where the computers and equipment are located. The tasks for the IT support people that we have considered in this scenario, are: *Setting up new computers*, *computer upgrades*, and *assist employees to deal with software/hardware problems*. A more detailed description of this scenario can be found in [Hof02] based on interviews with the IT-support department at our university.

2. **Mobile researcher scenario:** This scenario describes a researcher travelling to a conference and working abroad. The tasks that we have considered for the travelling researcher, are: *Prepare presentation*, *perform presentation*, *write on scientific papers*, and *miscellaneous administrative tasks* (email, coordination with colleagues, etc.). This scenario is further described in [SWL+02] and is based on our own experiences from travelling and working.

3. **News production scenario:** This scenario describes the work in a newspaper with the purpose of reporting the latest events happening in a city, involving the roles reporter and photographer. The reporter will *collect data from the site of event* (like interviews and notes), *write a report*, and then *review the report*. The photographer will *take pictures from the site of event*, and *edit pictures*. This scenario was produced based on an interview with a journalist in the biggest newspaper in Norway.

4. **Mobile learning scenario:** Mobile learning (mLearning) is an extension of eLearning making learning available anywhere at anytime. In our scenario we describe typical tasks for students and teachers performed in a mobile setting in a biology course at a university. Students perform the following tasks (using mobile devices): *Watch video lessons*, *read course material*, *do exercises*, *work with a group on a project*, and *check course status*. Teachers are involved in the following mobile tasks: *interactive field excursion reporting* (using audio/video recording), and *record live video/audio lessons*. This scenario describes typical tasks for a course at our university put in a mobile setting.

After analysing the four scenarios, we found that the news production and the mobile learning scenarios had the highest *Overall scenario complexity indicator*, with the scores 3.35 and 3.32 respectively. The IT support scenario scored 2.31, and the mobile researcher scenario scored only 1.90 on the same indicator. This result confirms our suspicion that a system to support a mobile researcher would be the least complex system, and the mobile learning and news production scenarios involving real-time video transmission would be the most complex ones. Also note that the mobile researcher scenario represents nomadic work, which normally can be supported by a standard distributed system. We can analyse

these results further by looking at the more specific indicators as shown in Figure 10.1. Looking at the *General Task Indicators*, we can see that scenarios 3 and 4 demand a more advanced (flexible) task support/infrastructure (e.g. workflow system). Advanced task support often means that you need advanced transactional support in order to allow such a flexibility. If we look at the *Transactional Support Indicator*, we can see that the scenarios 3 and 4 also score high here.

The *Information Complexity Indicator* (ICI) is used to indicate how the end-system must cope with presentation, management, and transmission of information. Also here, we can see that scenarios 3 and 4 have high scores for the ICI. This is probably because both scenario 3 and scenario 4 involve real-time video, while the scenarios 1 and 2 involve text or possibly graphics that are not transmitted frequently.

Looking at the *Location Complexity Indicator* (LCI), we can see that here scenario 1 is the most complex in this respect. This is because the IT-support scenario involves several tasks that are locked to specific locations and it is important to be able to locate the IT-support people.

The *Time Complexity Indicator* (TCI) indicates how complex the time management of tasks need to be in the end-system. Here, the news production scenario is the most complex, due to strict time constraints and coordination of tasks. The next two indicators are the *Network Connectivity Indicator* (NCI) and the *Network Speed Indicator* (NSI). Also here scenarios 3 and 4 have the highest scores due to a need to transfer big amounts of data real-time. The IT-support scenario also has a high NCI score because the need to the report location of IT-support people.

The scores of the *Energy Consumption Indicator* (ECI) shows that mobile clients required in scenario 1 consume most energy, followed by scenario 4. The IT-support scenario has a high ECI because the IT-support people needs to go to specific locations over a long period of time using a mobile client. In the mobile learning scenario, a lot of energy is consumed when watching a video on a mobile device or making video recording at a field excursion over several minutes or possibly hours. The last indicator is the *Mobility Indicator*, which describes how much mobility is involved in the execution of tasks. Also here the IT-support scenario has the highest score followed by news production and mobile learning. The mobility indicator does not indicate the mobility measured in distance, but rather how much of the work involves moving around or to specific locations. The tasks of the IT-support scenario must be performed at a specific location to be completed, thus a high mobility indicator. The mobile researcher is not so much bound to specific locations, giving a low mobility indicator score.

From these examples, we can show that our indicators are valid for several different situations. This also means that for future development of mobile systems, we can use them to set up relevant priorities and requirements.

10.6 Discussion and conclusion

The indicators presented in this paper are oriented towards different implementation aspects of a system. It is therefore interesting to discuss what the indicators mean in terms of the development of a system. We have found that the indicators represent tools to find which parts of a system are the most challenging (and uncertain) in terms of complexity, and therefore also solvability.

In agile methodologies, like eXtreme Programming (XP) [Bec99], architectural prototypes are used to delimit the uncertainties of the estimates to implement certain requirements (user story). To illustrate, a high ICI indicates that the system should be able to handle rather complex information. To make it certain that the forthcoming system is able to provide the necessary capability to handle the data, a prototype can be built to ensure that the different parts of the system is able to handle the required data, e.g., in a timely manner, on a specific equipment or a wireless network. In some cases, the hardware or network has limitations that must be taken into account. Then it might be of interest to build a prototype system that adapts to the capabilities of different devices. Here, it may be argued that the limitations are known. However, due to the constant evolution of all components of the mobile infrastructure, software solutions not only have to adapt to the current (state-of-the-art) technology, but also have to deal with future developments. An example is the availability of components – e.g., component of the shelf (COTS) – to integrate into the system to solve particular issues such as an uncertainty with respect to applicability, functionality etc.

Both the characteristics and the indicators can be used as inputs to harvest components for the system architecture. However, it may be hard to have a precise opinion of what is really needed, even if all the major requirements are captured. The indicators can be used as a means to find out whether components could be developed in-house (in the case of low indicators), or should be bought and integrated into the system (in the case of high indicators), and even investigate if similar systems are already available.

In many cases, however, it is the use of the framework on several scenarios that will be of the highest value. Certain characteristics of systems are often recurrent, and it may exist design patterns to formulate and solve similar problems. In the case of, e.g., the information complexity, it is possible to find certain kind of architectures that can provide the required level of quality of service, such as in database technology and server proxies.

This paper has presented eight complexity indicators that we believe are useful to set up and prioritise requirements for a mobile work environment. We have also shown that these indicators can be applied to real-world scenarios. This has provided us the possibility to assess the complexity of such mobile scenarios. Our indicators can also be used to group scenarios with common characteristics. This means, in turn, that scenarios of the same group may reuse similar technology and architecture.

Currently, we are using our framework and indicators to develop a mobile IT-support system. This will help us to evaluate the strengths and weaknesses of our approach in a practical setting. For example, this may reveal what indicators are important and what are less important.

Acknowledgement

This paper is part of the MObile Work Across Heterogeneous System (MOWAHS) project. The MOWAHS project is sponsored by the Norwegian Research Council's IKT 2010 programme.

CHAPTER 11

P8 – Using the MOWAHS Characterisation Framework for Development of Mobile Work Applications

Alf Inge Wang, Carl-Fredrik Sørensen, Heri Ramampiaro, Hien Nam Le, Reidar Conradi, and Mads Nygård [1]

Abstract

This paper describes an evaluation of a characterisation framework to analyse mobile work scenarios in order to make corresponding software systems. The framework identifies complexity issues to be taken into account when implementing a system. The framework can also be used to elicit requirements from a scenario. Three research questions are investigated in this evaluation: 1) Can the framework be used to identify relevant challenges in the final system? 2) Can the framework be used to identify functional requirements for the final system? and 3) Can the framework be used to identify non-functional requirements for the final system? The evaluation was performed using the framework to analyse and implement an IT-support scenario. The paper also describes a web-tool for this framework that makes the characterisation process simpler. The tool introduces consistency rules to ensure stricter characterisation of the scenarios.

[1]Dept. of Computer and Information Science, Norwegian University of Science and Technology (NTNU), N-7491 Trondheim, Norway. Phone: +47 73 594485, Fax: +47 73 594466, http://www.mowahs.com

11.1 Introduction

The explosive development the last decade in mobile computing has changed the way we communicate, learn, entertain and work. Mobile phones and PDAs have become necessary tools to make life easier through functionality such as SMS, calendars, WAP-browsers etc. As mobile devices have become more powerful, it is now possible to create software systems for mobile workers that can improve the work processes. Such systems typically consist of various mobile clients connected to a server. They provide the mobile worker with necessary information and opportunity for filling in forms and reports on various locations. Development of mobile systems is different from development of distributed systems. When designing a mobile system, we have to overcome challenges in wireless communication, physical mobility, and portability [FZ94]. Thus, it is important that these issues are examined carefully when considering the system requirements. In this paper, we review a framework that is used to identify challenges and possible requirements related to system development for mobile work. By mobile work, we consider work where the worker must move physically to some location to carry out his task. The goal of this framework is to identify the parts that are most complex and probably would be hardest to implement in a mobile support system. The framework has previously been used successfully to analyse mobile scenarios (like mobile journalist, mobile researcher, mLearning [RWS+03]) to compare their characteristics. Although we found this analysis useful, it did not imply that the framework is adequate for developing software to support mobile work. The framework it self has been published before, and the contribution of this paper is the **m** tool used to apply the framework on scenarios, consistency rules to improve the quality of the characterisation, and most important an evaluation of the framework. In this evaluation, we want to investigate three research questions:

RQ1 Can the framework identify *relevant challenges* related to the development of mobile support systems for mobile work?

RQ2 Can the framework introduce *new functional requirements* related to the development of mobile support systems for mobile work?

RQ3 Can the framework introduce *new non-functional requirements* related to the development of mobile support systems for mobile work?

The first question investigates whether the framework does what it is supposed to do i.e. to identify complexity in a system. The next two questions investigate whether new functional and/or non-functional requirements can be derived from using the framework. To answer the questions, we used the framework to analyse mobile scenarios when developing a system for mobile work.

The rest of the paper is organised as follows: Section 11.2 describes the characterisation framework for mobile work, Section 11.3 describes how the framework was applied on

the IT-support scenario, Section 11.4 describes the results of the evaluation, Section 11.5 relates this paper to similar work, and Section 11.6 concludes the paper.

11.2 The Characterisation Framework

The MOWAHS characterisation framework is a tool for analysing mobile work scenarios to create a mobile computing system supporting these scenarios. Such a system will typically be a process support system tailored for supporting mobile professions, and consists of one or more servers and some mobile clients (laptops, PDAs, mobile phones etc.). The framework can be used in at least two ways: *Firstly*, it can be used as a check-list of issues you should consider when making computer support for mobile work. *Secondly*, the framework can be used to perform a more careful examination of the requirements for making a system to support mobile scenarios. This examination will produce requirement indicators to identify complex parts of the system, type of client device, type of network, services needed etc. To use our framework, a mobile scenario must be described as a set of tasks. A task is here similar to a use case in design of software systems. To apply our framework to a mobile scenario, we use the following steps:

1. Select the mobile scenario and identify the different roles/actors
2. For each role identify tasks
3. For each task:
   (a) Write a task description using a task description template
   (b) Assign task priority (1-5 where 5 is most important)
   (c) Characterise the task using the framework (see Figure 11.1)
   (d) Calculate the requirement indicators for the task
4. Derive system requirements and priorities from the characteristics and indicators.

To measure the different characteristics in the framework, we use an ordinal (Lichert) scale (1-5). High values indicate more complexity in terms of system requirements, while low values indicate lower complexity. Many of the characteristics do not use the full scale, but use the values 1, 3 and 5 to get a uniform representation of extreme values. We are aware that it is mathematically incorrect to calculate average value of the ordinal scale, but we have found it useful to do so. We define the importance of a task by assigning a weight on a scale 1-5 (from very low to very high).

Task Characteristics		
Characteristics	**Possible values**	**Description**
GENERAL		
G1. Decomposable	(1 No, 3 Uncertain, 5 Yes)	Is the task composed of sub-tasks?
G2. Part of sequence	(1 No, 3 Uncertain, 5 Yes)	Has the task has order dependencies with other tasks?
G3. Pre-planned	(1 Planned, 3 Partial, 5 Ad-hoc)	To what degree is the task planned in beforehand?
G4. Data synchronization	(1 Never, 3 After completion, 5 During)	When in the execution will the task update data with other tasks?
G5. Data exchange rate	(1 Never, 3 Once, 5 Many)	How often will the task exchange data with other tasks within its lifetime?
INFORMATION		
I1. Information contents	(1 Text, 3 Graphics, 5 Multimedia)	What is the complexity of information required or produced by the task?
I2. Information streaming	(1 NA, 3 Discrete, 5 Continuous)	Does the task require streaming of data?
I3. Freshness of data required	(1 NA, 2 Day, 3 Hour, 4 Min, 5 Real-time)	How fresh must the data received from a server be to execute the task?
I4. Freshness of data produced	(1 NA, 2 Day, 3 Hour, 4 Min, 5 Real-time)	How fresh must data received by a server and produced by the task be?
I5. Data transmission	(1 NA, 2 Slow ,3 Medium, 4 Fast, 5 Very Fast)	What is the expected transmission speed required to execute the task?
LOCATION		
L1. Location dependent	(1 No, 3 Partial, 5 Yes)	Must the task be executed at a specific location ?
L2. Require services at location	(1 No, 3 Partial, 5 Yes)	Does the task require electronic services available at the location?
L3. Produce services at location	(1 No, 3 Partial, 5 Yes)	Does the task produce electronic services available for other systems/users at the location?
L4. Location report	(1 No, 3 Partial, 5 Yes)	Must the task report its current location to a server?
L5. Route constraints	(1 No, 3 Partial, 5 Yes)	Must the task follow a specific route when moving around?
TIME		
T1. Event-triggered	(1 No, 3 Partial, 5 Yes)	Is the task triggered by an event?
T2. Time constraint	(1 No, 3 Partial, 5 Yes)	Must the task be executed at a specific time?
T3. Temporal coordination	(1 No, 3 Partial, 5 Yes)	Must the task be timed with other tasks?
T4. Task resumption	(1 No, 3 Partial, 5 Yes)	Can the task be halted for later to be resumed from where it left off without a complete restart?
T5. Task lifetime	(1 Sec, 2 Min, 3 Hours, 4 Days, 5 Weeks)	What is the expected lifetime of the task?

Figure 11.1: The MOWAHS Characterisation Framework for Mobile Work

11.2.1 Requirement Indicators

From the scores on the twenty characteristics in our framework (see G1-G5, I1-I5, L1-L5 and T1-T5 in Figure 11.1), we can compute indicators that can help us analyse the mobile scenario and help us prioritise and extract non-functional and functional requirements. The requirement indicators we have identified for our framework are (the first four are simple aggregates and the six following are combined aggregates):

General Task Indicator (GTI) is an average of G1-G5. A high GTI score indicates that the underlying process and transaction infrastructure (e.g. workflow system) must be advanced.

Information Complexity Indicator (ICI)) is an average of I1-I5. A high ICI score indicates that the end-system must cope with complex information presentation, management and transmission. The ICI can also be used to select the appropriate hardware and software to be used as a mobile client and server.

Location Complexity Indicator (LCI) is an average of L1-L5. A high LCI score can indicate that the end-system must be location-aware, include Geographic Infor-

mation System (GIS) functionality, and use a mobile client suitable for mobility in terms of weight, size, battery power etc.

Time Complexity Indicator (TCI) is an average of T1-T5. A high TCI score indicates that time management and coordination of tasks, and advanced transaction support might be necessary. In addition, the TCI also indicates the level of performance and availability required.

Network Connectivity Indicator (NCI) is an average of G3, G4, G5, L4, and T1 and indicates the level of connectivity between the mobile client and the server. It determines the required networking capabilities of the mobile client. Further, it indicates non-functional requirements for the system such as reliability and latency.

Network Speed Indicator (NSI) is an average of I3-I5. A high NSI score means that the transmission speed and quality of service must be high between the mobile client and supporting servers. The NSI also indicates what wireless network technology can be used for the end-system.

Energy Consumption Indicator (ECI) is an average of I5, L1, L2, L3, and T5. A high ECI score means that it is likely that the mobile client device can complete the task will consume much energy.

Transaction Support Indicator (TSI) is an average of G3, G4, G5, T1, T4, and T5. TSI describes the need for flexible/advanced transactional support. A high TSI score indicates that the transactional support must go beyond ACID transactions.

Mobility Indicator (MI) is an average L4 and L5, and indicates how much mobility is involved. The MI is useful for determining the complexity of the environment the mobile client will operate in, e.g. variation in wireless networks. A high MI will affect the choice of equipment (device) and tools necessary to accomplish the task.

Task Complexity (TC) is an average of all characteristics of a task and indicates the complexity of one task. The TC is useful for finding the most complex task that should have the most attention in a further examination.

11.3 The Framework Applied to Develop a Mobile Application

In this section, we describe how we applied the framework to implement a system for supporting mobile IT-support.

11.3.1 The Mobile IT-support Scenario

In 2002, we investigated ways of improving the software used by our IT-support department to manage incoming requests from users, assigning these requests to appropriate personnel and following up these requests. The IT-support department used a system called the Request, Users, and Sys-admin To-do Ticket System (RUST) [Rue96a]. RUST is web-based, mainly used to manage requests from users by putting the requests in queues, assigning user requests to IT-personnel, tracking status of requests etc. A big problem with this system was that the IT-support staff did not update the status of tasks in the system when completing the task. This caused the number of user requests in RUST to grow uncontrolled, and it was impossible to know the actual status of user requests. We therefore proposed to extend the RUST system with mobility support, to make it possible for the IT-support staff to use mobile clients at work. This would make it possible to bring important information about the tasks to the labs or offices of concern, write a short report of how the task was solved, and change the state of the task as ”resolved” when completed. To extend the RUST system, we started to create a scenario that described the mobile tasks. We first identified the roles involved: Support manager, Support desk and Support engineers. We found that only the support engineers *had* to move around to do their job. We then identified the mobile tasks for the support engineer role:

> **Task 1**. Install new PCs (Support engineers): Install new hardware and software. Priority: High (4).
>
> **Task 2**. Upgrade such PCs (Support engineers): Upgrade existing computers with new hardware and/or software. Priority: Low (2).
>
> **Task 3**. Assist users (Support engineers): Help users to solve computer problems (like malfunctioning mouse, keyboard, software etc.). Priority: Very high (5).

11.3.2 Characterisation of the Scenario

The scenario was characterised using a web-based tool called **m**. Initially, we started to use a simple spreadsheet to characterise mobile scenarios using our framework. However, inexperienced users of the framework found it difficult to know what numbers to choose for the different characteristics. To make this easier, we decided to make a web-based system that could guide the user through the framework by providing useful information for every step of the process, and warn the user when he tried to enter inconsistent values for related characteristics. We created seven consistency rules used by the tool:

> **Rule 1**: $I3 \geq 4 \vee I4 \geq 4 \Longrightarrow I5 \geq 4$
> If the freshness of data required or produced by a task is within seconds, the data transmission must be fast.

Rule 2: $L4 = 5 \vee L5 = 5 \Longrightarrow L1 = 5$
If the task must follow a specific route or if it must report its location, the task is location dependent.

Rule 3: $T3 = 5 \Longrightarrow G2 = 5$
If a task must be coordinated with other tasks in respect to time, the task is then part of a sequence.

Rule 4: $I2 = 5 \Longrightarrow G5 = 5$
If a task requires streaming, it will require exchanging data many times with another task (the server).

Rule 5: $G3 = 5 \Longrightarrow T1 = 5$
If a task is ad-hoc, then it is event-triggered.

Rule 6: $G2 = 5 \vee L5 = 5 \vee T2 = 5 \Longrightarrow G3 = 1$
If a task has time constraints or is part of a sequence, or has route constraints, it must be planned in advance.

Rule 7: $T4 = 5 \Longrightarrow T5 \geq 2$
If a task can halt and later resume from where it left off, it must have a lifetime that can at least be measured in minutes.

The **m** tool can be used to analyse any scenario as long as the process described in Section 11.2 2 is followed. The tool allows redefining our existing framework by adding, changing and removing characteristics, but also by creating new frameworks for analysing other properties than mobility. In addition, the consistency rules can be added, changed or removed. We found it necessary for the tool to support evolution, since we have revised the framework three times up till now. These changes have been both editing characteristics and editing consistency rules.

When using the tool, the user is guided through all the necessary steps to carry out a proper characterisation. The process can be divided into two main parts:

Part 1: Defining the Scenario.

First, the user is asked to create a scenario by giving it a name and a short description. The scenario description is then extended by adding all the roles that are involved in the scenario (see Figure 11.2a).

The next step is then to describe all the tasks that should be a part of the scenario along with selecting the responsible roles for each task (see Figure 11.2b). A task is described by a name, description, responsible role, and pre- and post condition related to location, resources and other tasks. It is up to the user to decide the level of detail in task descriptions, but name and responsible role are mandatory.

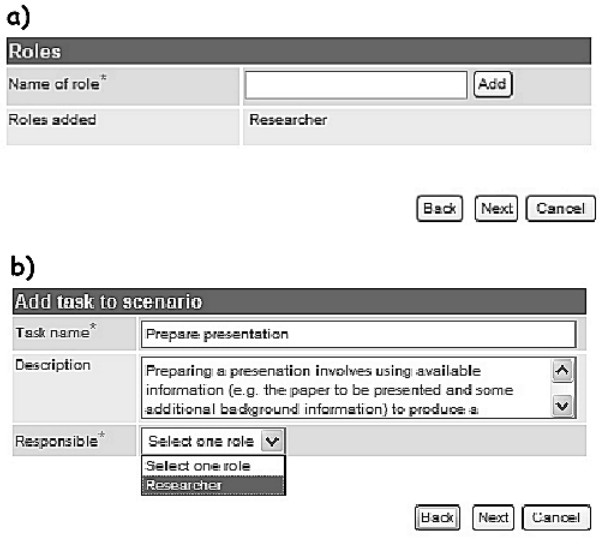

Figure 11.2: Defining a scenario in the **m** tool

Part 2: Characterising the Scenario.

The characterisation is performed by giving scores to the twenty characteristics in the framework as shown in Figure 11.3. The user interface for the characterisation consists of three main parts: The upper left area, where the user applies drop-down menus to select score for every characteristic; the right area, where the current task is described with is attributes; and the down left area, where the user is warned about violations of consistency rules.

The user can choose to ignore these rules or change his/her characteristics to comply with the rules. In the following section, we present the results of applying the **m** tool on the IT-support scenario.

11.3.3 Results of the Characterisation

The results from applying the MOWAHS characterisation framework to the IT-support scenario are shown in Table 11.1. To improve the quality of the characterisation, the scenario was characterised independently by three people before the results were compared. The three results were consistent only with small variations. Compared with previous characterisations, we see that the consistency rules and the tool have improved the precision of the framework.

We can see from task complexity (TC) scores in the table that Task 3 is the most complex of the three tasks. We can also see that Task 3 has the same or higher values on all indicators than the other two tasks. Task 3 has also the highest priority and we will

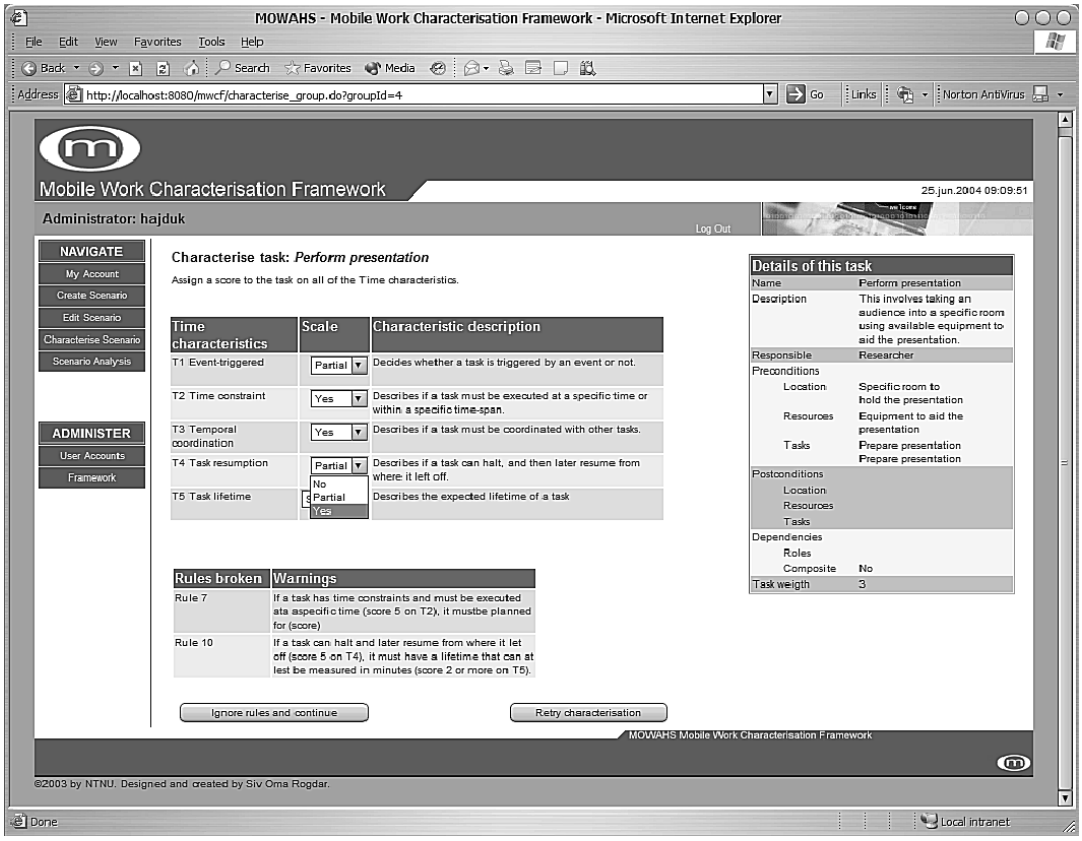

Figure 11.3: Characterising a scenario using the **m** tool

concentrate the analysis on this task, as it will most likely cause the main challenges in the system. If we first consider the GTI, ICI, and LCI, we can see that they all have the value 3,4. This means that they are all above average complexity. A more thorough analysis shows that Task 3 requires workflow/process support enabling dynamic and unpredicted behaviour, like defining new activities on the spot and redefining parts of the process during enactment. Further, it is necessary to enable synchronisation of data between enacted activities.

An analysis of ICI shows that the mobile client should be updated frequently to notify changes to others using the system. Also the mobile device should be able to display graphics, but a full-fledged multimedia device is not necessary.

A further analysis of the LCI shows that Task 3 is location-dependent, as it might utilise and produce services at the location, and that the location should be reported. This indicates that the system should provide some kind of location-awareness.

The TCI score is 4,0 which indicates that system should allow scheduling of tasks and that the process model should have a time attribute as a part of the task description.

Requirement Indicator	Task 1	Task 2	Task 3
General Task Indicator	2,2	2,6	3,4
Information Compl. Indicator	2,4	3,4	3,4
Location Compl. Indicator	2,6	2,6	3,4
Time Compl. Indicator	3,0	3,0	4,0
Network Connect. Indicator	2,2	2,2	5,0
Network Speed Indicator	2,0	3,7	3,7
Energy Consumpt. Indicator	2,5	2,5	3,3
Trans. Support Indicator	2,3	2,3	4,5
Mobility Indicator	3,00	3,00	3,0
Task Complexity	2,55	2,90	3,6

Table 11.1: Scores from the IT-support scenario

The NCI indicator is 5,0, and indicates that the mobile client must be continuously available (online) to support the system engineer. This means also that the system can only be used effectively within the actual range of a wireless network. If we consider the NSI with the value 3,7, the network speed should be between medium (UMTS) and high (IEEE 802.11b). This will also put limitations on the kind of hardware we can use in the final system.

The ECI for Task 3 is 3,3. It indicates that the mobile client to support the task will consume about half of maximum power. If the ECI has been very high, it could have indicated that the mobile client might need to recharge between every task and could not perform several tasks without access to electric power.

The TSI for Task 3 is 4,5. It indicates that such a system is likely to require a transaction support beyond standard ACID and that strict locking cannot be used. This also means that we need some mechanism/infrastructure to warn users and enable negotiation between users that try to access and modify the same data.

Finally, the MI has a value of 3,0. It shows that Task 3 has average mobility, and this indicates that the mobile client device should be rather small and light. In addition, since mobility is involved, the mobile client device should also survive a fall down on the floor or likewise. Further, this will also mean that the user interface of the system has to adapt to a small screen, and a full-size keyboard cannot be used.

11.3.4 Requirements Found by the Framework

The following requirements were found by using the framework (the indicator the requirement is extracted from is given in parenthesis):

Functional requirements for the system:

RS1 Must provide a workflow infrastructure to enable management of updates/changes of tasks during enactment (GTI).

RS2 Must enable data exchange between tasks during enactment, e.g. between two mobile clients (GTI).

RS3 Should provide location-aware services to the user (LCI).

RS4 Must enable tasks to be related to time and support for executing tasks at a specific time (TCI).

RS5 Must offer a transaction support beyond standard ACID-properties (more flexible) (TSI).

RS6 Must provide infrastructure for handling conflicts between users accessing the same data (TSI).

RS7 Must provide user interfaces adjusted for the small mobile device that only show the necessary information and widgets (ICI, LCI and MI).

RS8 Must allow buffering of necessary information on the client in case of network disconnection (ICI, LCI and MI).

Non-functional requirements for the mobile client device:

RM1 Must be able to display some graphics, but full multi-media capability is not necessary (ICI).

RM2 Should have a built-in network interface for IEEE 802.11b or better (NCI and NSI).

RM3 Must have a battery that can last several hours even if the wireless network is used continuously (LCI, MI, NCI, and ECI).

RM4 Must be small, light, and endure falling down to the floor (LCI and MI).

Non-functional requirements for the infrastructure:

RI1 Must have full coverage of a wireless network in the areas the system will be used in (NCI and NSI).

RI2 Must have a wireless network capacity of IEEE 802,11b or faster (NCI and NSI).

RI3 Must provide a way of detecting the position of mobile clients to enable location-aware services (LCI).

11.3.5 Development of a Mobile IT-support System

Our first idea of developing a system for mobile IT-support was to extend the existing RUST system [Rue96a]. However, we found that the architecture of RUST was not easy to extend and also had poor performance because all data was stored in various text files and not in a database. Since our IT-department was familiar with the RUST system, we wanted to provide roughly the same user interface and functionality as the existing system. In addition, we wanted to expand the system with mobility functionality based on the requirements found in Section 11.3.4.

The Mobile IT-Support System

From the requirements given in Section 11.3.3, we started to investigate how these could be integrated in the design of the system.

Firstly, we found that the original process model of RUST had to be extended to relate location and time to a task. In addition, we had to add a location model that could relate rooms, buildings and campuses to a location. We ended up with a hierarchical location model that can identify object position based on room, section, floor, building, and campus. The model describes every level in a distance matrix, enabling us to determine the distance between two locations.

Secondly, we started to look at what kind of location-aware services we should provide to the user. For the support manager role, we found it useful to be able to ”sort” tasks according to their location. Thus, we could assign all tasks that are geographically close to the same support engineer to improve work efficiency. We wanted to track the locations of the engineers using the system, to automatically assign new incoming tasks in the same area. For a support engineer, we wanted to provide a service for sorting the assigned tasks according to proximity. The mobile client should automatically show relevant tasks for an engineer in a specific area. This would, e.g., make it possible for a support engineer to go to a PC-lab and get a list of all the tasks related to this lab on the mobile client.

Thirdly, we found that an advanced transaction model was required for the system. Sup-

port engineers share information on pre-assigned, on-going and completed tasks. In addition, they share data concerning their experiences and knowledge about previously performed tasks. To be able to do this, a knowledge base storing such information had to be provided. Maintaining consistency and durability is important ensuring that all involved parts can have access to the correct information at any time. However, since sharing is emphasised, an engineer should be allowed to see what another engineer is doing. To enable this, transaction isolation could be relaxed. Further, as a task may last a long period of time, it is important to allow engineers to save parts of their on-going tasks. It could happen that tasks must be terminated due to some failures or other situations. When tasks are aborted we also have to be able to allow them undo only parts of their tasks rather than aborting all. This calls for a relaxed atomicity. To summarise, ACID transactional support would be too rigid for this kind of application; instead, we have to allow relaxed atomicity and isolation, but at the same time preserve consistency and durability. A transactional framework proposed in [RN04] can be used to achieve this. Unfortunately, we did not have a finished implementation of a transactional framework that could be used, so we had to leave out this part in the final system.

The Mobile Client and the Infrastructure

From our characterisation, we found that the mobile device used as a client in the system should be small and have an interface to a wireless network with high bandwidth. All the buildings the IT-support is responsible for have full coverage of an IEEE 802,11b wireless network (WLAN). This means that the mobile clients used in the system can either be laptop PCs or PDAs with WLAN capabilities. In addition, the whole campus has full coverage of a GSM wireless network allowing the use of mobile phones as client devices. The main problem of using mobile phones is the slow GSM network and the cumbersome input device. We found that the PDA would fit best as a mobile client, because of network capabilities and size. However, as most of the staff of the IT-department owns a mobile phone we wanted to provide a client for such devices as well. Our system ended up with support for three kinds of devices: Portable and stationary PCs using standard web-browsers; WLAN-enabled PDAs using limited web-browsers; and mobile phones using WAP browsers.

As these devices have very different characteristics, we did not offer all the system's services to all devices. For instance, the assignment and management of tasks is only possible in the client for PCs. On mobile phones, it is only possible to look through the relevant tasks and change status of these tasks. The functionality of the system is not tailored for each type of client, but the PDA and mobile phone provides a subset of the available functionality. The user interface, however, is tailored for the three kinds of clients. The mobile phone is supported through a WAP/WML interface, while a HTML interface is provided for the PDA and the PC. The HTML for the PDA is adjusted for small screens and does not include frames.

Figure 11.4 shows how the system looks at a portable PC (a), a mobile phone (b), and a PDA (c).

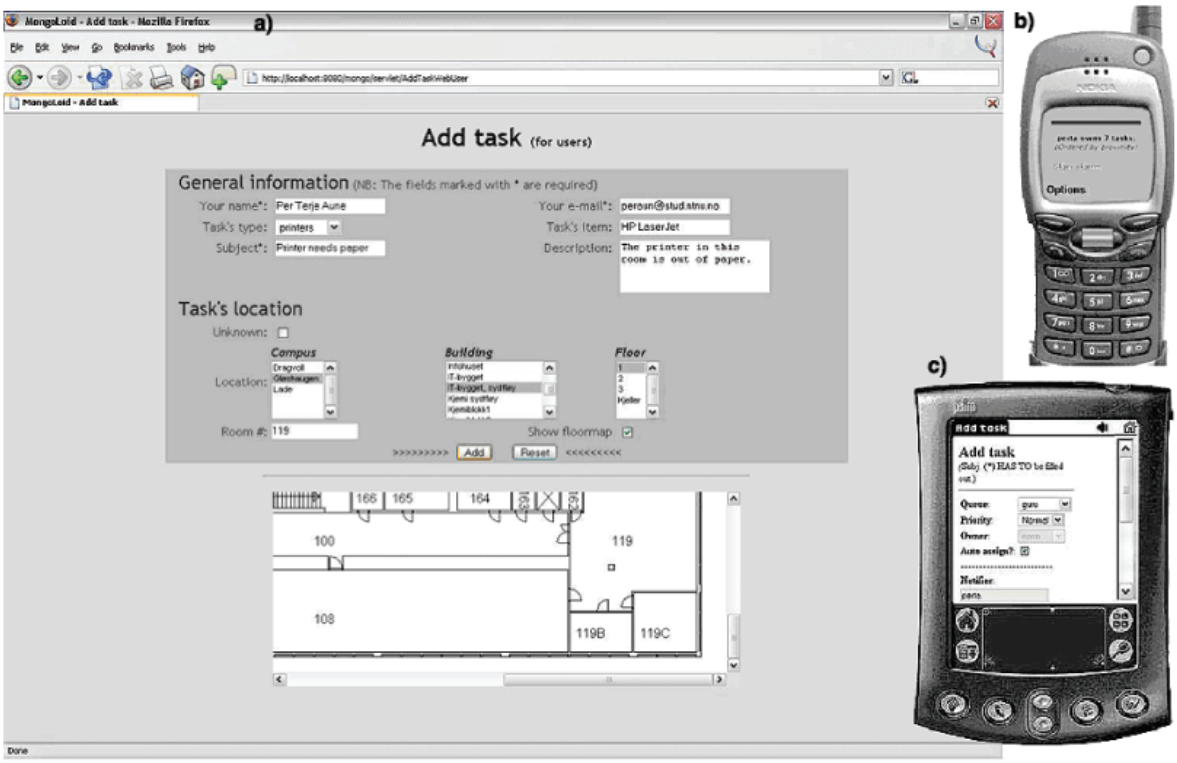

Figure 11.4: Screenshots from the system for Mobile IT-support

One problem of our system was to localise the mobile clients. We planned to use the base-stations in the WLAN network for positioning, but could not it was not supported in the network. We solved this problem by letting the user explicitly report his position through a user-interface.

11.4 Results of the Evaluation

In the spring of 2003, we completed the implementation of the prototype we called MObile Nagging Geek Organiser (MONGO) [SWØH03]. The MONGO system had all the functionality of RUST, had support for mobile clients and had mobility support through location-awareness. Two students working for about one year carried out the work. From the early beginning of the project until the project was finished, we made a note about how the usage of the framework affected the development of the system. This was done to be able to answer the research questions we wanted to investigate.

The first research question (RQ1) asked **whether the framework could identify relevant**

challenges. From the findings in Section 11.3.3, the main challenges (requirement indicators with highest value) identified were an unpredictable and dynamic process, complex timing of tasks, users updating/accessing the same data, high level of required network connectivity, and high-speed network. The three first challenges are functional challenges, while the two last are infrastructure challenges. For the latter, we discovered that this was not really a challenge for deploying the system as we had the infrastructure in place. It is perhaps wrong to say that high scores on network connectivity and network speed introduce complexity in the system. However, if the infrastructure in our campus area had not been in place, it would be quite expensive and time consuming to install such networks. For the former three functional challenges, we found these problems too complex to implement ourselves. One solution would be to acquire an advanced workflow system that handles dynamic processes, scheduling, and advanced transactions. The conclusion is that the framework indeed identified relevant challenges from the scenario, even if we did not overcome the challenges in the final implementation.

The second research question (RQ2) asked **whether the framework could be used to identify new functional requirements**. In Section 11.3.4, eight functional requirements (RS1-RS8) identified by using the framework are listed. Some of these requirements are related to the main challenges described in the previous paragraph (RS1, RS4, RS5, and RS6) and was not implemented in the final system. The rest of the functional requirements were implemented in the system. When designing and implementing the system, all the functional requirements were relevant and important to the system. However, could we have come up with this list without using the framework? The requirements that we already had identified before using the framework were RS1, RS2, RS4 and RS7. Analysing the scenario using the framework discovered the rest of the requirements. It is not only the computed numbers of indicators that lead to the new requirements, but the characterisation itself forced the designers of the system to consider various aspects. To conclude, we found that the framework is a useful tool for identifying new functional requirements.

The third research question (RQ3) asked whether the framework could be used to identify new non-functional requirements. In Section 11.3.4, seven non-functional requirements are listed (RM1-RM4 and (RI1-RI3). The non-functional requirements that we identified were relevant to the final system, but were not very useful. Most of these requirements must be tested with respect to response times and system performance before knowing for sure that the correct equipment and infrastructure is used. However, when we tested the system on a GSM phone and a WLAN PDA, we discovered that the network speed indicated by the framework was correct. To conclude, we found the framework only partly useful for identifying new non-functional requirement. This was mainly because the framework could not give any quantifiable non-functional requirements that could formally be checked.

11.5 Related Work

We are not aware of any other similar characterisation framework for mobile work. We will therefore in this section outline research that is related to characterisation of mobile scenarios.

Maccari [Mac99] presents an example of requirement engineering challenges in the mobile telephones industry due to the complexity of mobile phone architecture and performance. In addition, requirement engineering for mobile telephones has to cope with many issues such as protocols and technology standards. Also, the limitation of wireless devices, such as network connectivity and network speed, implies important challenges that developers have to deal with. The author argues that requirement engineering is a collaborative task.

Zimmerman [Zim99] suggests the ”MOBILE” framework to determine when mobile computing technology should be used to solve challenges related to mobility. This framework focuses on the current technology and software development trends in mobile computing. Further, common related scenarios are discussed, including news reporting and hotel operations. The framework provides a useful overview of necessary support needed for specific mobile environments. However, the framework does not provide any guidelines for how to develop or design systems for mobile support.

Satyanarayanan [Sat96] identifies four constraints of mobile computing concerned with limited resources, physical security (e.g. theft), and communication and durability issues. Another approach is proposed by Forman and Zahorjan [FZ94] who examine three basic features of mobile computing including wireless communication, mobility and portability. These two approaches provide different ways of addressing mobility issues. The former focuses on connectivity issues, while the latter deals with Quality of Service (QoS), such as network bandwidth and device durability.

Rakotonirainy [Rak99] discusses current and future technologies (e.g. CORBA and mobile IP), adaptable to mobile computing environments. For this, he presents a scenario revealing the limitations of the current technology. Although characteristics of mobile work can be derived from this approach, he does not provide a comprehensive framework for characterising mobile work environments.

The related work above mainly focuses on the technical aspects of mobile computing. Our framework investigates mobile computing from another point of view by focusing on the mobile scenario to be supported and identification of the various system complexities introduced by the scenarios. In addition, our framework focuses on software support for mobile work and identifies issues that are not directly related to mobility like process and transaction infrastructure.

11.6 Conclusion

In this paper we have presented an evaluation of a characterisation framework that can be applied to mobile work scenarios in order to create a software system to support such scenarios. The evaluation shows that the framework is useful for identifying challenges and functional requirements when developing systems that should provide support for mobile work. We discovered that the framework was not so useful for identifying non-functional requirements. The framework was found useful for analysing the mobility aspects of a scenario to be implemented as a system. The characterisation process is useful in itself by systematically focusing on the various aspects of mobile scenarios. Further, the computed requirement indicators will produce a profile of the characteristics of the scenario. The indicators help the designer of a system to identify the complex parts of the mobile system that could lead to decisions on buying COTS components that must be integrated in the system. The framework can also help improve the planning and resource usage in later phases of the project by outlining the hard parts.

From our experiences of using the MOWAHS characterisation framework as a tool to develop systems to support mobile work, we have found the framework useful. The system that was developed using the framework (MONGO) has been tested in actual usage with people from our IT-support department. From these tests, the users have given positive feedback on the mobile features of the system. The main complaints from the users were related to the lacking support for a more flexible process enactment (refining the process on-the-fly) and more advanced management of data updates (transaction management). Both these issues were identified by the framework, but were not implemented in the final system. The support engineers testing the system said that the most useful feature of the MONGO system was that they could bring the system along when working away from the office. This would make it easier to update the state of tasks in the system at the spot and write resolutions at problems at hand that could be reused later. The future will bring many possibilities in providing computer support for mobile work, and believe that our framework can help in developing such systems.

Acknowledgement

This paper is a result of work in a project called MObile Work Across Heterogeneous Systems (MOWAHS) [MOW04]. The MOWAHS project is sponsored by the Norwegian Research Council's IT2010 program. We would like to thank Øystein Hoftun and Per Terje Aune for their contribution in implementing the MONGO system.

CHAPTER 12

P9 – Support of Smart Work Processes in Context Rich Environments

Carl-Fredrik Sørensen, Alf Inge Wang and Reidar Conradi [1]

Abstract

The evolution of mobile and ubiquitous technologies gives promises for computational services and resources to support and influence work processes planned or performed in physical work environments. Such support should optimally provide the workers with a safer work environment for both the environment itself and the workers. The extended support can give more economic and optimal work processes through proactive and situated planning and execution. We introduce the concept of a *smart work process* to capture adaptive and context-aware process support. This combination of ubiquitous computing and workflow defines a new research direction to be investigated. This paper elaborates on research challenges related to how smart work processes can be supported. We present and discuss general cases where context information or change in context information influences mobile work activities. Finally, we propose a framework for modelling smart work processes, and present a high-level architecture to support smart work processes.

[1]Dept. of Computer and Information Science, Norwegian University of Science and Technology (NTNU), N-7491 Trondheim, Norway. Phone: +47 73 594485, Fax: +47 73 594466, http://www.mowahs.com

Keywords: Mobile work, Mobile and ubiquitous environments, Smart work processes

12.1 Introduction

The future mobile and ubiquitous computing environments hold promises for computational services and resources to become invisible but important parts of the supporting computing environment for all kinds of user activities. It can be used to improve the accessibility to information and computational services. The information sources can be very diverse, from simple sensors sampling environmental properties, to complete information systems with the ability to roam e.g. the Internet for information and services. Mobility imposes a dynamic and unstable environment that challenges how to create mobile work support environments. New opportunities to be utilised by the mobile workforce include management of ad hoc activities and cooperation with other people, as well as exchange of information and services with the surrounding environment.

Today, mobile computing and communication support is mostly directed to provide availability of services while being mobile through wireless networks and mobile computing devices. We consider this support to be an extension of existing distributed and wired services to make it possible to work distributed at different places using mobile computers.

The demanded computational support will differ from user to user, and from activity to activity. The needs are based on, e.g., what process or context information available or required by the user, and under which conditions an activity is to be performed. The conditions are influenced by temporal changes of context information as well as the behaviour of humans or other actors in the environment. Together, they constitute the state of the mobile working environment.

In this paper, we discuss and differentiate mobile work from nomadic work, and then identify issues related to how different types of context information can be supported and utilised in systems to support mobile work processes. We therefore introduce the notion of the smart work process. We further propose a process framework and architecture to support smart work processes.

12.2 Motivation

Many work places have a complex structure of participants, activities, and artefacts. This makes even simple work processes hard to plan and execute. Such environments are dynamic and unpredictable. Resources can be scarce and different participants can therefore compete to make necessary adaptation in the environment to fit their own goals. Often in such environments, safety is a very important issue for the employers and other actors in

the society.

Safety for single actors can be ensured by establishing safe working conditions to perform activities within, or to re-establish safe conditions by performing situated or planned activities. Actors can influence the working environment in different ways that can create safety-breaks for other actors as well as ”stealing” resources from others. By providing context information to the individual and to supervisory process enactment services, it is possible to initiate coordination activities to establish safer and more economic working conditions. Smart work processes can be used as a means to coordinate multiple actors. Reactiveness to environmental changes is thus important to ensure an optimal safety and production rate. In environments that are self-aware, i.e., equipped with augmented, ”intelligent” artefacts [SGKK04], activities can be initiated by the environment to ensure certain environmental goals. In addition to provide activities, coordination of actors and sequencing of activities can be performed to maintain or enhance productivity, safety, or other goals defined in the environment, by the actors or by their processes.

12.3 Mobile Work Environment

Mobile work has often been looked upon as an extension of distributed work in terms of technological solutions to support such processes. Support of distributed work includes systems to manage configuration of shared artefacts, and to manage coordination and collaboration among the participants. Mobile work support systems may have need for these properties as well, but mobile work is clearly distinct from distributed work by the motivation to be *mobile* and how the dynamic change of physical environment influences how to perform work. The supporting infrastructure suffers from unpredictability and availability to the people that work in a mobile setting. In addition, mobile workers also have to face dynamic and partly unpredictable changes in the physical environment. To establish a clear definition of mobile work, *mobility* must get more emphasis to distinguish mobile work processes from distributed, co-located or individual work processes. Mobility must be a property necessary to perform the actual work, irrespective of technology. This means that work that is independent of mobility, cannot be characterised as mobile work, even if the work is performed when mobile, i.e., change of location is not a precondition for performing the work. We can therefore split work processes performed in a mobile setting into two different categories based on context-sensitivity:

- **Work in a mobile environment**: Work processes performed in a mobile environment *independent* of context information extracted from the physical environment. That is, mobility is *not necessary* to accomplish the process goals.
- **Mobile work**: Work processes performed in a mobile environment *dependent* of context information extracted from the physical environment. That is, mobility is

necessary to accomplish the process goals.

A simple public transportation scenario can illustrate the difference: A bus is transporting people from one place to another. The driver is performing mobile work because he has to adapt to changes in the physical environment. A bus passenger working with a laptop connected to a wireless network is not performing mobile work but work in a mobile environment. Figure 12.1 illustrates the differences between nomadic work, service work, and inherent mobile work. **Nomadic work** refers to **anywhere, anytime computing**, often called **nomadic computing**. The goal is to provide users with access to popular desktop applications, applications specially suited for mobile users, and basic communication services in a mobile, sometimes wireless environment [LSG96]. Such work is not regarded as *context-dependent*. **Service work** refers to workers that need to travel to a specific location to perform work. The work is thus context-dependent, but the context can be regarded as quite stable. **Inherent mobile work** requires continuously change of location and thus a dynamic environment. All work performed when in a mobile situation may require similar technological and computational support.

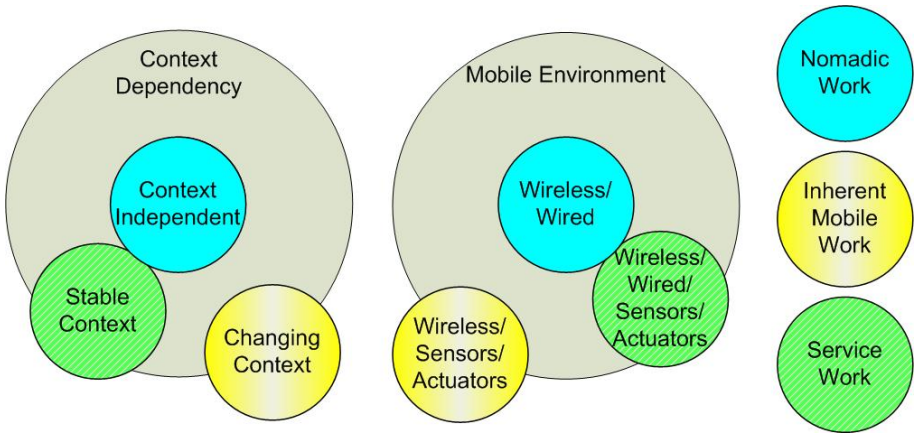

Figure 12.1: Mobile Work vs. Working when Mobile

To support mobile work, the evolution in mobile computing technology enables access to distributed workflow systems. However, the mobile environment is dynamic, and processes depending on changes in properties of the local environment are not particularly supported by such systems. Thus, for mobile work, it is necessary to have access and support related to the local environmental context. Traditional distributed workflow system with enabling mobile technology is therefore not sufficient to support mobile work processes.

Mobile work and the local working environment can mutually influence each other in different ways:

- *Mobile work can change the state of its environment*. This can be done directly by initiating and performing activities with the purpose of changing the environmental state, e.g., by introducing actuators or other equipment to control or change the environment. The environmental state can also be changed indirectly through activities performed by one or more actors (workers) that together introduce state changes.
- *The state or change of state in the environment* can influence many different aspects of how, when, by whom, and what to be done in an environment. The environmental state is thus directly or indirectly used in the planning and enactment of the work process. The state or state change can:
  - *Initiate activities that must be performed*.
  - *Decide start, duration, delay, stop, and termination of activities*.
  - *Decide which activities to be performed*.
  - *Change the content or goal of an activity*.
  - *Initiate exceptions in the current activities*.

Activities and change of state in the environment together provide dynamics to reach process goals by adjusting each other. This point relates to how an environment can be regarded as an organism with a certain amount of order (state) infused with chaos through activities, where the order and chaos must balance to successfully reach a pre-defined or situated process goal.

We will in next section go into deeper details how to support mobile work in a mobile working environment.

12.4 Smart Work Processes

In this section, we unify mobile work with environmental context-awareness to propose a framework for process models to be used to support mobile work. To be able to sense the environment, we need methods and tools to capture relevant context information that affects the mobile work process. This information will be necessary to drive the work support application; we therefore denote all externally sensed information as *context*.

To provide the required process support to deal with work processes that influence or are influenced by the mobile environment, we introduce the notion of a **smart work process**:

> *A work process that is* **sentient***, i.e., sense the environment which it performs in,* **adapts** *to relevant context or context changes through* **context-based reasoning** *to reach process goals, and* **actuates** *by providing situated activities,*

or by changing/refining the current planned activities.

Smart work processes are thus especially tailored to make use of context information to monitor and coordinate activities within a context-rich environment. Importantly, coordination is the glue for how to manage singular activities within an environment. Coordination needs to be performed between specific work processes and the work environment, between multiple actors performing possibly cooperative or competing work processes, and with respect to some stated paramount requirements like safety, time, and economy. The contextual relationships are not directly related to the normal relationships that exist between activities within ordinary, office-like work processes or plans. There is therefore a need for extracting and specifying how work support applications can be adjusted or adapted to also cover smart work processes.

12.4.1 Context-Aware Activities

An activity can be defined in a process model by providing goals, pre- and post-conditions, invariants, and the use or production of artefacts and resources. The process model elements may be affected by context information in various ways. Context or context changes can in such a model be used as pre-conditions whether to start, stop, or terminate an activity. The activity is thus directly affected by and dependent on the contextual state and hence need to include specifications and rules for how the context types are related to the activity. Context can also provide rules for how and what to do in an activity. In this case, the context is used to specify activity content and thus refining and concretising the abstract definition of an activity. Further, context can provide alternative process paths as described in Section 12.3. The process paths can either be pre-specified or need in situ specification when encountering new states not covered by the model. Context can also be used to trigger or create new activities not previously defined in a process model. This case is related to situated actions [Suc87] and can possibly be handled through situated planning as described in [Bar97]. This means that the process model needs to be extendable in runtime to cover in situ specification and enactment.

Physical work or an actuation resulting from an activity can directly or indirectly change the physical environment (post-condition). Such a change can either be predictable and enforced through the activity and thus can be specified, or be a side-effect. Side-effects must either be included into the process model in cases where such effects are preferable, or be met by reciprocal activities to counter the side-effect. In any case, side-effects must be handled by the process enactment service. In some cases the environment is in such a state that it is necessary to perform activities to provide pre-conditions to perform new or (pre- or situated) planned activities. A related case is activities to prepare the environment for a certain planned activity. This requires a process model to be extendable based on how activities are dependent on certain environmental states to be executed.

The cases above identify how a process enactment service must take into account the surroundings to effectively support the enactment of smart work processes. The process model is in addition to the changes in the process plan and goals, also affected by state changes of elements in both the physical and the computational environments. The elements in the environment may be affected by the process, but also by other factors that might be out of control from the enactment service point of view. The environment may contain other ”competing” actors, artefacts, resources and other elements that cause coordination problems between actors, the physical and computational environment itself, and the process goals of the different actors.

12.4.2 Situated actions and planning

The state of the physical work environment will often be fluctuating and dynamic, and it is thus very hard to provide realistic or detailed plans beforehand for which activities to be performed. Definition or refining of activities based on the perceived state of the environment will therefore be necessary to create executable process models. Some activities can also be defined in situ by the environment itself. This is possible if the environment is self-aware and augmented with applications and technology making the environment able to change its own state as well as ask for concrete actions by external or passing (human) actors not statically bound to the environment. The state of the environment provides means to create activities to change the environmental state to a level where a planned goal or intention may be accomplished.

Situated planning and thereby actions will therefore be a natural part of a support environment for smart work processes. Context information is thus used for both definition and adaptation of work processes in situ.

12.4.3 Challenges in context-aware process support

The support for context-aware or smart work processes is to a small degree covered in the literature and is thus a new area of research within pervasive computing and workflow/process support. Since the domain has little coverage in the research community, many challenges arise in the cross-section of mobile/wireless computing, ubiquitous/pervasive computing, and workflow:

- Specification of *contextual pre/post-conditions* related to some process goal.
- Specification of *environmental behaviour* related to an activity (adaptation).
- Specification of a *uniform representation of sensors and actuators* from a process enactment perspective to make it possible to reason about and change the context state of the environment.

- *Planning, specification, and execution* of activities in concert with the current environmental context. These challenges relate to how the dynamics are handled by the workflow enactment services, both on client and on server.
- *Managing process changes to ensure a consistent state* of the process.
- *Managing the dynamics of ad hoc activities and process changes* locally and in a central enactment service.

12.5 A Process Framework for Smart Work Processes

Process models are used to define how work should be performed and can be used for visualising processes, guiding the user through the process, automating steps of the process, educating users about new processes, analysing or simulating the process etc. To define a process model, a process modelling language (PML) must be used.

The Workflow Management Coalition [Hol95] provides a reference model for how to build a workflow management system and how to make interfaces to other applications and workflow systems. To support smart work processes, some additional or changed requirements are necessary to provide extended support of context-aware work processes. We therefore specify some of the requirements to a process support system supporting smart work processes:

The process model must be able to *adapt* during the process enactment. This is partly covered through the use of exceptions in the reference model.

The process model must be *refinable/extendable* by situated planning. Working in a dynamic environment with coarse plans is quite normal. Such plans are refined or extended based on the properties of the working environment itself. Activities resulting from situated planning are not possible to plan beforehand, but systems can use encountered processes to ”learn” about specific activities performed during specific situated work situations.

A *separate environmental or world model* must be built up to support specific processes by identifying relevant context sources that may affect or support the process.

Rules must be specified or developed for how the process model can adapt to the relevant context information. The rule model defines how the process and the environment may or should interact during the process enactment. The rules may be pre-defined, but must also be derived from the need for situated actions, the sensed state of the environment, and the process itself. In addition, the environment can provide rules based on artefacts augmented with smart sensors and some representation of ”intelligence” [SGKK04].

The working environment must have ”services” that can be given values of certain re-

quired properties or conditions to be satisfied during the process or activity execution, e.g., the environment is supposed to keep a certain temperature during a process. A break in such conditions can contradict the process goal and in the worst case endanger the environment itself or the people in the environment.

A *coordination service* that can be given values of certain required properties or conditions to be satisfied during the process or activity execution, e.g., the environment is supposed to keep a certain temperature during a process. A break in such conditions can contradict the process goal and in the worst case endanger the environment itself or the people in the environment.

Specific environments can have pre-defined activities or plans that can be put into enactment in certain pre-defined environmental states, e.g., defined safety procedures when the safety requirements are not satisfied.

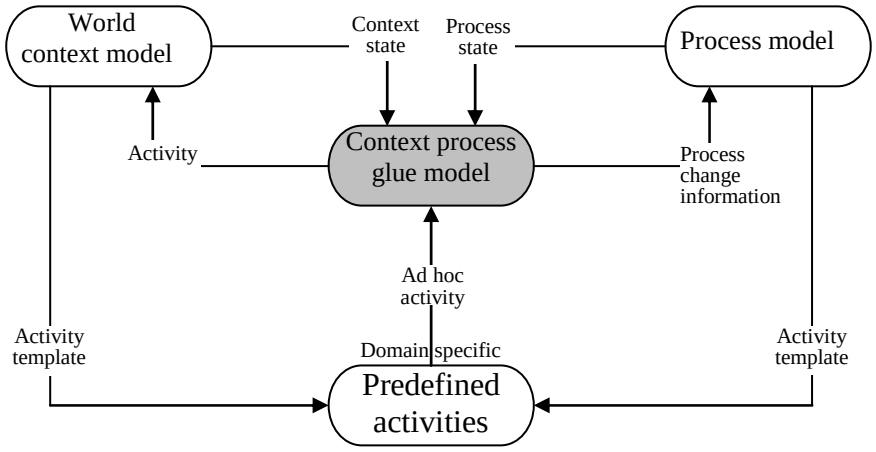

Figure 12.2: Smart work process glue model

Figure 12.2 shows a proposal of model needed to provide support for smart work processes and how these models relate. A central part of modelling smart work processes is the representation of the context relevant for the work process.

The *world context model* represents objects in a ubiquitous computing environment, often embedded with sensors and actuators. Such a world model should consist of all relevant static objects in an environment that somehow can influence the work processes. In addition, mobile objects that can provide relevant context information must also be integrated/included in such a model. Also non-tangible properties like e.g. weather, sounds, and light conditions may be relevant and therefore need to be included in the model. The construction of a relevant world model poses challenges since this means that the world model itself must be able to change when objects are approaching or leaving the environment. The static world model is, however, easier to comprehend and therefore more easily represented in a model. Other challenges are how to specify and select the relevant

objects in an environment to be included in the specification and enactment of context-aware work processes. Such a selection is necessary both when pre-planning the work process and when using situated planning. The world context model must represent the relevant objects in a way that make it possible to catch the state of the physical or execution environment. The state is needed to reason about and take according actions in the work process support system. A challenge here is to have a context model that is able to provide a unified representation of objects/sensors that can represent various physical data that can span from boolean values to complex, composite data types representing abstract properties in the environment. A *process model* is also needed that model the activities of the work process. Such a model must allow late binding and it must be possible to reconfigure and change the model during the process enactment.

A key in modelling and supporting smart work processes is the *context process glue model*. TThis model is a loose coupling between the world context model and the process model that defines rules for how the context should affect the process model and vice versa. The glue model is responsible to connect pre-planned processes with situated planning, and thus process enactment. For pre-planned processes that are context-sensitive, we proposed to use templates that can be refined during context-aware, situated planning. To deal with unexpected conditions in specific work environments, we propose to have a set of *predefined activities* used to handle such conditions. The predefined activities can dynamically be added to the process model and activated based on the new environmental state. A library of predefined activities will grow over time based on gained experiences on solving unexpected events. The predefined activities are influenced by the process model as well as the world context model.

12.6 An Infrastructure to Support Smart Work Processes

The future computing environment is predicted to realise the vision of the invisible computer [Wei02]. Computing devices are in this scenario embedded in almost every kind of physical object. Sensors are spread like ”dust” in the environment or appear as ”brilliant rocks” [Sat03]. This vision makes the computing environment extremely distributed and gives indications of an enormous amount of possible context sources and services to autonomously support people in their whereabouts. Augmented, intelligent artefacts can provide self-aware computing elements in the environment that can cooperate to provide rules and actions used by the process support system. In addition, actuators can provide services that can change the state of the environment. Such actuators can be robot-like and thus have the possibility to perform a limited number of activities (partly autonomous, mobile units with communication capabilities).

We have developed a high-level architecture as shown in Figure 12.3. The architecture consists of seven main parts which will be described in more detail below.

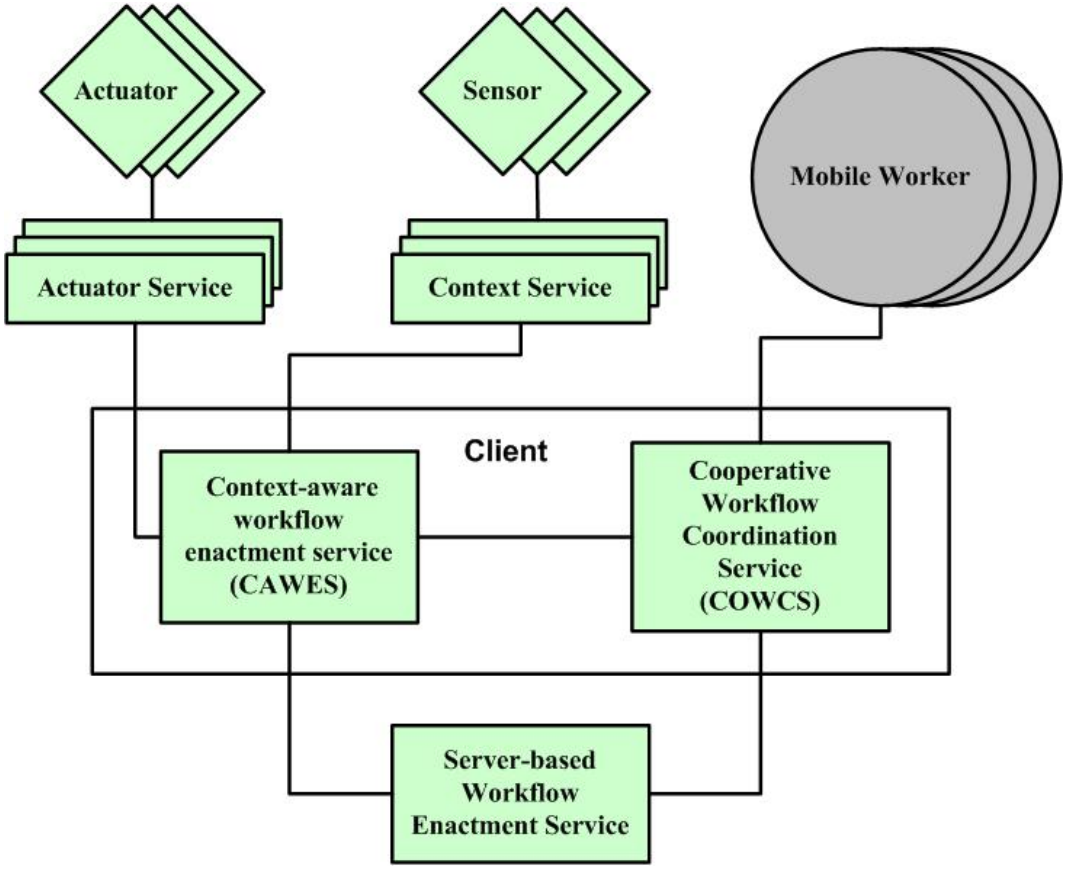

Figure 12.3: An Architecture for smart work processes

Actuator: An actuator is an entity that is able to change the environment based on digital input from some digitial equipment. In our architecture, an actuator is responsible for acting according to specifications instructed by the actuator service.
Sensor: A sensor is responsible for measuring, and transmitting sensor readings according to specifications by the context service. Sensors might vary from very simple, to autonomous, augmented entities.
Actuator Service: An actuator service is responsible to initiate actuations in an environment. The service should be able to receive instructions from the workflow enactment service and translate these to the actual actuator. The actuators can in this service be represented as abstract entities with published properties. Actuator services can hide how to communicate with the different actuators and provide a well-defined interface to be used by the service clients. The actuator service should be able to communicate with actuators through various physical communication implementations like IR, WLAN, LAN, BlueTooth, etc.
Context Service: A context service is responsible for identifying sensors, initiate sen-

sor readings, check sensors, setup/terminate sensor subscriptions, etc., i.e., provide the communication with all actual sensors. The context service can accept subscriptions from different clients and is then responsible for setting up the sensor subscription properties. The clients can specify which properties they prefer to receive. The context service is responsible for handling client sensor subscription information including receiving rules, preferences, and facts related to sensor properties; collect/aggregate basic readings into abstract context (e.g. weather conditions). Based on the properties received from the various clients, the context service should send context messages to the client subscribers related to subscriber preferences. A context service may be extended to also represent e.g. augmented artefacts. It should then be able to communicate process fragments based on templates initiated by certain contextual states/conditions either contained within the context service, or received from smart, autonomous sensors. Possible context services that can be provided are e.g. weather conditions, engine and application states, etc. (including the state of the client itself), location, and the mobility of the user and other objects/entities within the environment.

Client-Based, Context-Aware Workflow Enactment Service (CAWES):
CAWES is responsible for monitoring and executing delegated activity (ies), through surveillance of the current process/activity state, and contextual events received either from the context service, the COWCS (see below), or the Server-based Workflow Enactment Service. CAWES can also receive process/activity fragments from the context service, the COWCS, and the Server-based Workflow Enactment Service, can send its own process/activity state as contextual events to the COWCS, and the Server-based Workflow Enactment Service. If it is a need for actuations, CAWES initiates actuations using the Actuator service. Further, it can display/inform about environmental state wrt. the process itself, safety state, etc. It can also display/inform about information related to the activity performed. Such information can be based on the activity itself (manuals, checklists, or multimedia information), location-based information, safety information, etc. In additions, CAWES should also provide a traditional workflow user interface for filling in information, create/update artefacts related to the activity, e.g. interactive checklists (can also be filled automatically using the context service), manage/integrate audio/video/textual information collected through the device or an actuator service), etc. An extension of CAWES is to provide virtual/augmented, location-based information (manuals, multimedia) related to the working environment, process type, available tools and sensors.

Client-Based Cooperative Workflow Coordination Service (COWCS):
COWCS is responsible for managing and coordinating activities in a multi-actor environment. This includes a coordination service that based on policies can send messages to all involved parties about coordination needs, competitive resources, etc. COWCS receives process goals, and currently performed activities from the other actors. Similarly, it can send its current state of executing activity and possibly prepare other users about new activities to be started within a time limit, and coordination messages to other users including needs for artefacts, services, actuators, and a preferred environmental state. Internally on the client, COWCS sends contextual events to the CAWES based on coor-

dination reasoning. Such events may include information about the need for artefacts, services, actuators and an environmental state with priorities, time, demand, etc. It can also send new process fragments derived from the coordination service or received from other clients based on demand/need from other users. The fragments are sent both to the CAWES, and to the Server-based Workflow Enactment Service.
Server-based Workflow Enactment Service: The Server-based Workflow Enactment Service is responsible for traditional workflow management including sending activities to the users, based on delegation/plans, and managing the complete, high-level process for the involved participants. The architecture has not been completely validated, but we have developed some prototypes to start the process of validating the architecture and the concepts behind it. Unfortunately, few work environments have been sufficiently instrumented to try out the proposed technology in practice. This situation, we hope will be improved in a few years.

12.7 Related Work

[WL01b] states that ”anytime, anywhere” does not necessarily mean ”everytime, everywhere”. The ideal mobile situation is not to work continually without any stops. Further, true mobility goes beyond mobile support for ”here and now”. There is also a need to support the place to go, and the place left behind as well as to make plans for the future or backtracking earlier events. Mobile work is in many cases a kind of stationary work because the worker has to stop to perform any real work when work is physically oriented.

Situated actions [Suc87] and situated planning [Bar97] are terms that relate to our approach. It is hard to make predictions of which actions to perform in dynamic environments, i.e. the environment itself is influencing the actions based on the state of the environment. This motivates for a broader use of contextual information to support both the execution pf actions, and also the planning of them. Situated planning can be used to refine coarse-grained plans based on the contextual properties of the environment and the worker itself. Situated actions can be used to create immediate plans to be used in situ.

The CORTEX project proposes the sentient object (SO) programming model [FBCC02] for pervasive and ad hoc computing applications. Smart work processes can be looked upon as sentient objects. A smart work process can be decomposed into smart activities or process fragments that also can be regarded as sentient objects.

All the work presented above is important to enable sound architectures and designs for smart work processes. Thus, both conceptual and technical issues can be addressed by using similar design frameworks to create a workable infrastructure to support mobile work where context information plays an important role for a successful enactment of smart work processes.

12.8 Discussion

We have in this paper not discussed all challenges that may arise when working in mobile, ubiquitous computing environments. We have, however, identified a few issues that may be hard to address to ensure a safe and dependable support of smart work processes:

Supporting process state transitions and process enactment when disconnected from a central work process support server: When disconnected we need to establish local enactment services in addition to a central enactment and coordination service. It must also be possible to coordinate the clients decentralised in an ad hoc fashion.

Ad hoc activities based on environmental context: The definition of ad hoc activities can be done in at least three ways: Such activities can be automatically defined by inferring activities through context-sensing and reasoning, they must be specified semi-automatically (using partly pre-planned templates), or be specified manually. This requires knowledge about the relationship between the context state and activities to be performed to reach an either pre-planned, or a situated goal. The activity must also be directed to the correct worker and then be coordinated with other activities in the environment. The state initiating the ad hoc activity can at the same time also influence the other activities in the environment creating conflicting goals or incompatible activities. This requires coordination activities to solve inconsistencies as well as optimising the activity throughput in the environment.

Ad hoc collaboration between environment and users, user to user: Collaboration between the different actors in a working environment is often not specified explicitly. In some cases, the collaboration is specified in the PML, but often collaboration can be initiated by the current situation or state in the environment. It is a challenge to create support systems that are able to create such process ad hoc relationships.

Human considerations: Context-aware systems [[BE01]] must consider properties like intelligibility and accountability to give dependable systems providing added value to the user. We envisage the need to provide intermediate evidence of the applicability and dependability of the solutions, and that the number of different kind of context sources to be included in the systems must be slowly increased to keep the systems manageable and accountable. The importance of *which* context information to include *when*, will therefore be a paramount issue to evolve applicable rules. Learning processes should therefore be included to evolve the inference engines embedded in the systems.

Technical issues and solutions related to mobility, mobile devices, wireless networks, and the inherent properties of these, have not been covered in this paper. Such issues are important to address to successfully implement support for smart work processes.

12.9 Conclusion

We believe that the concept of a *smart work process* is useful as an abstraction of adaptive, context-aware work processes used to model and support mobile work in future ubiquitous computing environments. We have in this paper clearly distinguished mobile work from nomadic work to better illustrate how future process support should be developed. Current process modelling languages do not support situated processes directly influenced or created by the environment. Such processes are mainly unknown to the work process support systems until the situation has occurred in which the process is to be performed in. Context information captured in the working environment and used in work support applications can in the future be important to keep control of dynamic work environments. Coordination and implicit cooperation between the environment and participating actors can ensure a safer and more economic work environment directing and coordinating work activities using all relevant context information that can be captured using sensors and reasoning techniques. A lot of work remains before the vision of smart process support systems can be realised. Especially in multi-actor environments and environments with autonomous equipment like robots, the need for supervision and adaptation of work processes is important to ensure safety, and economy through better utilisation of the available resources.

Part IV

Enabling Technology Papers

CHAPTER 13

P2 – Experience Paper: Migration of a Web-based System to a Mobile Work Environment

Carl-Fredrik Sørensen, Alf Inge Wang and Øystein Hoftun [1]

Abstract

It is a large collection of software supporting the planning, execution and reporting of work activities, but these systems are mainly created to be run in a distributed wired network, or as standalone systems. The systems do not directly support the mobile part of work. The cost and effort to make such systems available to mobile users are not just an issue of making adjustments to the network protocols, but also to be able to adapt the system to a heterogeneous set of hardware spanning from small mobile phones to portable PCs. This paper presents experiences from migration of a Web-based task administration system to a mobile environment. Different solutions of system adaptation are discussed, and a prototype of a system to support mobile work is presented and evaluated.

Keywords:

Mobile work, mobile devices, application evaluation, mobile software systems, technology migration

[1]Dept. of Computer and Information Science, Norwegian University of Science and Technology (NTNU), N-7491 Trondheim, Norway. Phone: +47 73 594485, Fax: +47 73 594466, http://www.mowahs.com

13.1 Introduction

The functionality, usability and availability of mobile electronic devices are improving, both professional and for pleasure, as the software, hardware and network technologies are evolving. This gives opportunities to exploit the technologies to support a greater part of the mobile work force in their daily work. The software to support the planning, execution and reporting of work are mainly created to be run in a distributed wired network, or as standalone systems. Most of the systems do not directly support the mobile part of the work. The cost and effort to make such systems mobile are not just an issue of making adjustments to the network protocols, but also to be able to adapt the systems to heterogeneous hardware, spanning from mobile phones to portable PCs. These may have different screen sizes, input capabilities, CPUs, memory sizes, network connectivity, and operating systems. The diversity of devices and their capabilities create substantial challenges for software providers wanting to create or enhance systems that can be used by several different types of mobile devices [BFJ+01, THSK01].

The systems must provide adaptable user-interfaces to the different screen sizes and presentation paradigms [GWH02]. This is not only a question about how the graphics and text is formatted, but also of how much information and functionality should be available for the different devices. The functionality provided by the systems should also be dependent on the input capabilities of the device, such as the provided keyboards (mobile phones keyboard vs. a full-fledged keyboard), capability of touch-based input, and voice recognition. The format of the graphics and text is not only dependent on screen size, but also on the ability of the screen to display the graphics (resolution, colours, etc.).

The types of network connectivity offered by the devices must also be considered by the software providers. Some types of mobile devices have only indirect network connection through other devices like mobile phones or PCs. A software system must be able to store the necessary information at the device to enable off-line operation. Other devices such as GPRS-enabled mobile phones have almost a fixed network connection and do not in that extent need to buffer the information. The software providers should also be aware of other limitations of the devices such as the battery lifetime and network connectivity only within limited areas (e.g. Wireless LAN) [FZ94, Sat96]. Software engineering for mobility must allow heterogeneous software and hardware to live side by side, and make the necessary bridges and adaptations to facilitate the view of one system.

This paper focuses on a scenario to support the administration and execution of mobile tasks at the IT-support group at our department. We present experiences from a student research project to make a Web-based system available for heterogeneous mobile devices. Further the paper briefly evaluates different technical solutions to support mobility, and some experiences from the implementation of a prototype of a mobile task administration system for the IT-support group at our department.

The rest of the paper is organised as follows: Section 13.2 describes the Web-based task

administration system, Section 13.3 describes issues related to offer a mobile task administration system, Section 13.4 briefly discusses our prototype of a mobile task administration system, and Section 13.5 discusses experiences from use of the prototype. Further, in Section 13.6 we relate our work to similar research, and finally Section 13.7 concludes the paper.

13.2 Description of a Web-based Task Administration System

The daily routines of the IT-support staff involve mobile tasks where they have to go to specific locations (offices, computer labs etc.), e.g. to repair or change hardware, or to fix or install software. The staff is supposed to use a system called RUST (Request, Users, and Sys-admin To-do Ticket System) [Rue96b] to handle inquiries (tickets) from users about e.g. faulty hardware and software. RUST is a Web-based application where users of the computer systems can report problems or other support needs to the IT-support group. RUST was initially developed around 1995-1996. The Web interface, called WebRUST, has increased the worldwide use of RUST. RUST is used to allocate people to perform support tasks, and to make them able to solve the requests.

Efficient organisation and management of to-do lists with the user-reported problems are important tasks for the IT-support managers. As the to-do lists and problem reports increase, so does the time required to organise and maintain them. Without an effective method of managing these lists, also when mobile, important issues may be ignored or forgotten.

RUST is implemented in *Perl* [Per02] and is platform independent. The Web interface is a CGI-program written in Perl that requests different queues from a TCP socket, and generates HTML documents to present the data. The server program is a Perl-script used to read the system database (files).

To provide data protection, RUST implements a client/server framework using TCP sockets. The client interface makes a TCP connection to the server to retrieve the requested information.

The IT-support group started to use RUST in 1997, and to fit their needs, they have adjusted RUST at several occasions.

13.2.1 Evaluation of RUST

The IT-support group appreciates RUST as an important tool in their daily work, but experiences also some problems. These are mainly related to the support of ad-hoc and

mobile tasks.

One of the main problems is the amount of unsolved tickets. The obvious reason is the large amount of requests and problems reported every day. Another significant reason is according to the manager at the IT-support group, missing completion of tickets in RUST. When an IT-support person has solved a ticket, she is supposed to do two things: 1) She should write a few words to describe the work done to solve it, and 2) set the status of the ticket to *Resolved*. After solving a ticket, the IT-support person often starts to work on another ticket at the same location/area before returning to her office. The fact that several tasks are performed when away from RUST, it is understandable that it is easy to forget which tickets have been solved, and what kind of work that actually has been done to a specific ticket. When solved tickets are not reported to RUST, two more problems arise:

- The consequence of not commenting how a ticket is solved is a poorer knowledge database. The information and experience attached to a solved ticket could have been very useful when analysing and solving similar tickets in future inquiries. Further, characterising problems and solutions might be useful to be able to predict services required in a system. The services related to printers are often related to the time since the last similar service, like e.g. change of toner.

- The consequence of not closing a solved ticket is growing queues containing both solved and unsolved tickets. When queues are growing, it is difficult to get an overview of the tickets, i.e. it is hard to know which tickets have been resolved and which have not. This will further result in a less effective and over-complex system, which leads to unnecessary administration costs.

The many unsolved tickets in RUST leads occasionally to performance problems. This is mainly because of the use of plain ASCII text files to store the data.

13.3 A Mobile Task Administration System

As described above, the IT-support staff has a lot of mobile tasks. The lack of a mobile support system to discover new tasks or additional information, leads to a forced mobility for the support staff. They must move between the problem locations around the campus and locations with access to RUST. A mobile system could both improve the current work processes, and increase the work efficiency.

The problem of closing solved tickets can be better handled if RUST supports mobile clients. A report on how the ticket was solved, as well as closing the ticket, can be done at the problem location after performing the task, instead of later at their offices (usually not done anyway).

This paper investigates two options of how to offer a mobile support system. The first option is to integrate a mobile front-end to the existing RUST, while the second option is to implement a corresponding system with built-in mobile support in addition to the current functionality.

Initially, we tried to access RUST directly with mobile devices such as PDAs and WAP phones. This solution did not involve any modifications of the system, and testing was easy. Two different mobile devices were used in the test. The devices were a Palm m505 PDA with an AvantGo Web browser [Ava03], and a Compaq IPAQ with a Wireless LAN [The02] card and Internet Explorer. The test detected that this solution is difficult to use. There are several reasons for this conclusion:

- The HTML pages in WebRUST are naturally adjusted to PC displays, and not adjusted to the small displays on mobile devices. Another problem is that the PDAs do not support HTML frames (it is no point in dividing an already small display in even smaller parts).
- RUST uses the HTTPS protocol. HTTPS encrypts and decrypts user requests as well as the information returned by the Web server. An IPAQ with a WLAN card is supposed to be able to access a Web page directly. The message ”You are not allowed access to this page”, indicates that the IPAQ has trouble with the HTTPS protocol.
- WebRUST is protected behind a firewall. This causes problems by using Web-based technology like AvantGo to synchronise the content on PDAs. Our AvantGo server operated outside the firewall.

13.3.1 The extended RUST solution

Another solution was to extend the existing RUST system with new components adjusted to the mobile environment. RUST could be extended with a new PDA interface based on WebRUST. Even though the PDARUST has the same foundation as WebRUST, there are some important differences. An adjustment to the PDA environment is of course required for a potential PDARUST. Parts of the WebRUST functionality are neither applicable nor useful in a PDARUST solution. The PDARUST should mainly be used as a tool when working on mobile or personal tasks. Administrative tasks are probably not very important in mobile scenarios, mainly because these tasks are not individual.

The users of the data systems usually report problems or other support needs to RUST by using email. It is important to be able to communicate with these users in order to inform about incidents related to the tasks, as well as request feedback or additional information about the tasks. The communication with the users is an integrated part of RUST, and is done by generating emails to task originators. The fact that these generated emails can

be read and answered by any email client is a major advantage of the extended RUST solution.

As discussed above, RUST uses the HTTPS protocol as well as being protected behind a firewall. An adjusted PDA component may encounter problems with the HTTPS protocol and the firewall.

There are issues related to the transactional management of a mobile solution. An example of a typical transactional problem is when two people are working on the same ticket. One may work off-line using a PDA, while another is working online on the same ticket. It complicates the transactional management that RUST uses plain ASCII text files to store the data.

13.4 A Mobile Task Administration System Prototype

A prototype of a mobile task reporting system [Hof02, HB01] was implemented to provide support to mobile work processes in a heterogeneous environment. The system was called MONGO (MObile Nagging Geek Organizer). The work processes in the IT-support environment are supported by provision of adaptable client solutions based on client type (Web, PDA and WAP).

13.4.1 Adaptable Functionality

PCs, PDAs and WAP phones were chosen as mobile clients for MONGO. These devices have different characteristics and are suitable to provide different kinds of support to the mobile IT-support staff.

[WSHB02] describes an evaluation of using PCs, PDAs and mobile phones as clients to a Mobile Task Administration System. It was evaluated how well the system has worked on the different devices based on several evaluation criteria like screen size, input device, physical device size, battery life, connectivity etc. During the evaluation it was concluded that the system should provide adaptable functionality depending on the type of device. Because of the limited input and display capabilities, the mobile phone was found only to be useful for checking task states, and for limited information browsing. The PDA was very useful for writing task reports when finishing tasks at the actual location. For managing tasks, which involve getting overview of the different tasks and states, and delegating tasks, a PC with a Web browser was found most suited. The MONGO system was therefore created to be adaptable based on client type for what functionality to provide. Figure 13.1 shows some of the proposed use cases to be offered for the different types of clients.

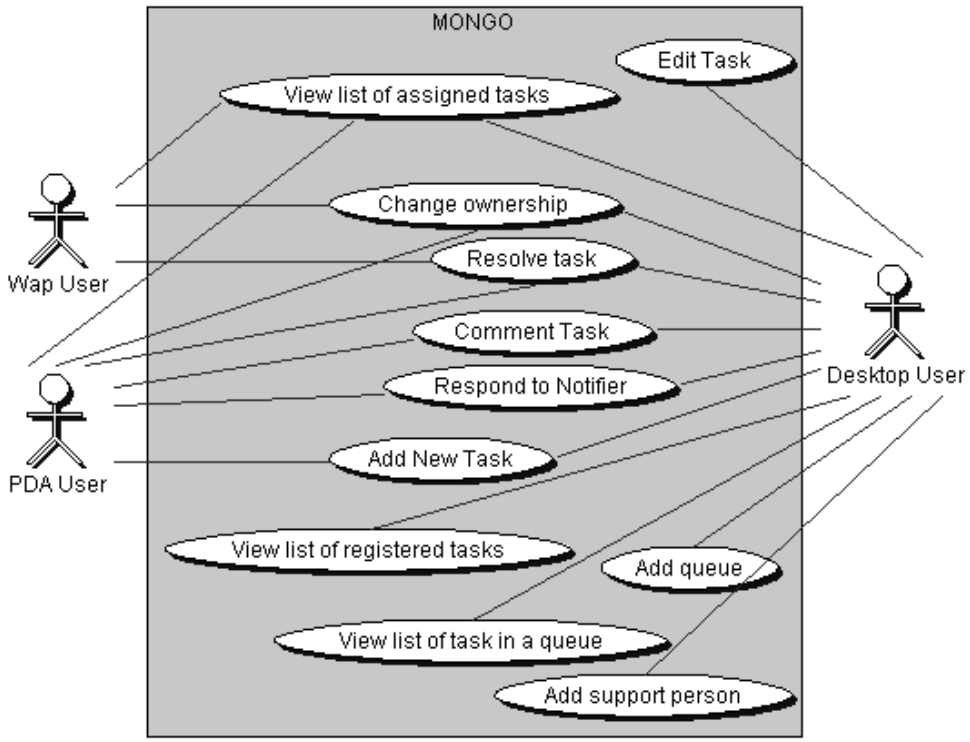

Figure 13.1: Adaptation of functionality to different types of mobile equipment

Figure 13.2 shows the basic system architecture of the Mongo system.

13.4.2 Technology

This section describes the choices of technology used in the prototype.

Database vs. XML [Wor02]

In the first prototype version, TRS [HB01], an XML file was used to store the data. XML was chosen due to the small amount of data involved in the prototype. MONGO handles more data than TRS, a database was therefore decided to be more suitable to use than XML files. It is easier to manipulate and manage data in a database than in several XML files, mainly due to performance, amount of code needed, testability and the availability of tools to manually manipulate data in a consistent manner.

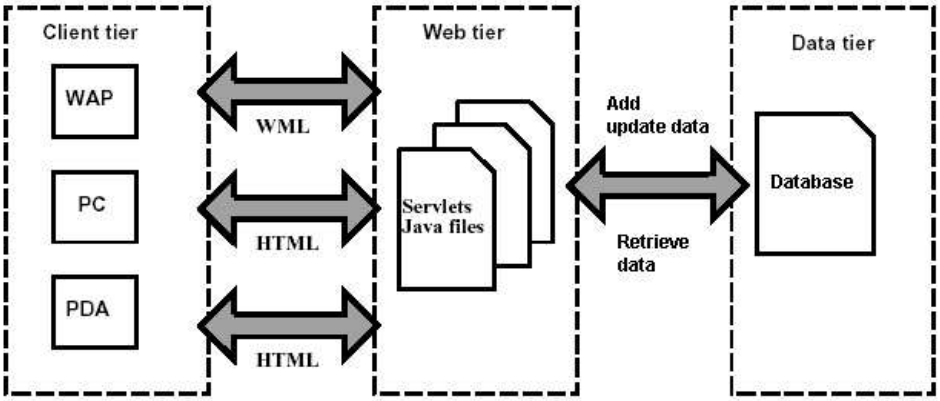

Figure 13.2: The basic MONGO System Architecture

Servlets vs. Perl

Perl and CGI loose the position as the most used Web technologies. Technologies such as Java, ASP and .NET handles issues like scalability and maintainability better, and provide frameworks to develop Web-server solutions with a lot of built-in functionality/-components.

Servlets vs. JSP (Java Server Pages)

Servlets was selected instead of JSP, because of the difference in performance. Servlets are pre-compiled Java files. JSP files are during the execution, interpreted and compiled into servlets, before being run.

Problems experienced using WML and WAP

The main problem encountered with WML pages for WAP phones, was the varied behaviour of the WML documents on the different browser and client types. Functionality that seemed to work well on one emulator, did not work satisfactory on another emulator. The varying behaviour of WML on different browser and client types forced us to test our implementation on several WAP emulators to find a common denominator.

13.4.3 Discussion of Mobile Device Types

WAP phones

WAP [Wap02] is a totally different paradigm than the Web. WAP has been designed to build services for mobile phones and has some limitations in use [BFJ+01]: 1) Displays

on the WAP phones are small; 2) Entering text is difficult compared to a desktop PC; 3) Navigation is cumbersome, requiring many ”clicks”; 4) They have limited bandwidth; and 5) The online time is relative expensive.

The small display makes it difficult to design good and usable WML-applications, and makes the usability even more important in a WML-application than in a Web-application. Good usability can only be achieved if the application is customised for each specific type of phone. It is important to minimise the need for textual entry, and the text should be terse. A wide WML-application often results in bad usability, and requires often too much user effort. People buy WAP phones primarily for making calls. Limited usability and limited useful functionality do not motivate the user to pay for many of the available WAP services. In order to become a real success WML applications have to solve real user problems in a simple way.

A favourable usage of WAP phones in our scenario is when the ownership of a task is given to a mobile support person already in the same location as a newly arrived task (maybe working on another task). The new task can be discovered and examined on the WAP phone, and hopefully be solved without returning to the office to get the task information. This may result in a more effective use of the support people, resulting in saved time and resources.

PDA

PDAs have one major disadvantage compared to WAP phones. Most of the PDA types available do not have the ability to be online when mobile. In order to get an updated version of e.g. the content at a Web site, a PDA has to synchronise with an online client, which can be a PC or a WAP phone connected to the Web. The PDA solution can therefore result in sharing violations. If the support people have different versions of the tasks in their PDAs, it may, in the worst case, result in several persons working independently at the same task. By using a PDA with a WLAN card connected to the wireless network, or by using a combined device, the problem described above will be solved. A smartphone (combined WAP phone and PDA), have the advantages of both devices and would be perfect for use in the IT-support environment. The disadvantage with such a device is however the purchase price.

13.5 Experience of Use

We executed a small experiment in order to evaluate the functionality and usability of the MONGO system. To perform the experiment, the IT-support group had three trial persons to work on real support cases using the MONGO system. Two of the test persons used a Palm Vx equipped with a keyboard. One person used a Compaq IPAQ H3850.

AvantGo was used as the client software for the Palm users, while Internet Explorer combined with AvantGo was used for the IPAQ. All users were given a task beforehand. The task involved visiting a specific office to give support on a software problem. The task descriptions were downloaded to the mobile equipment by the test persons, to make the content, nature, and location of the tasks accessible. During their visit, the work process provided by the MONGO system was used as the guideline. As part of the experiment, each test person was given a new task when mobile. The users gave, using the mobile device, information of the performed work, and the state of the task back to the system. After completing the mobile tasks, the system was synchronised (updated) with the offline information on the devices.

We experienced some issues during the experiment:

Working Offline

The system did not give any update opportunities of already submitted task information. The users wanted to add/change task information of already submitted information. In AvantGo, we found a workaround by using the *Form Manager* functionality of the Web browser. This was however not a satisfactory solution, i.e. this functionality was not accessible by using the MONGO client. We did not find a workaround for this problem in Internet Explorer (IPAQ).

Transactional Problems

The MONGO system does not give any specific transactional support to multiple and asynchronous updates of the same information item (task). Everyone is able to make changes to the state of the tasks by using the system online. Another problem is that updates from the offline devices are processed in the same order as they are synchronised with the system. This causes problems with the committed sequence of work, i.e. the timestamp follows the synchronisation time, and not the actual time of offline updates. Another related problem was that the time sequence of submitted information on the mobile device was not reflected in the synchronisation, i.e. non-deterministic sequence of the updates.

To cope with some of these problems, we observe a need to give timestamps to every offline update. When synchronising, the system must be able to sort the information (forms) based on the timestamp before processing the forms. There are however still issues of how to apply unique and correct timestamps to the information. Users may potentially be spread in different time-zones (for other kind of workflow systems).

Information availability

The users had mostly positive experiences from the use of the system. The mobile solution was regarded as valuable enhancement of the existing RUST system. However, they considered an always online solution to be even more important in order to discover new tasks in the vicinity of the current location, and be able to access knowledge bases related to the current task.

13.6 Related Work

Buchanan et al. [BFJ+01] discusses the usability aspects of the mobile Internet, and suggests that a reason for the failure of WAP, as well as potential failures of other approaches, is that not enough time is spent to think about the human factors of such systems. In our project, we decided to reduce the functionality offered to the WAP-phones due to the usability issues. We also wanted to personalise the available content on the different device types based on the need to access such content in a mobile environment, and based on the constraints of the wireless networks and devices.

Trevor et al. [THSK01] discusses different approaches to make Web-content available on mobile devices. The usability of the browser model depends on characteristics (user interface, computing resources, network bandwidth) of desktop PCs. Small mobile devices possess few of these characteristics. The gap between desktop and mobile devices must be considered while constructing the web content. There are a number of methods available to put Web content onto small displays. These methods rely mostly on the notion of *transducing* (conversion and transformations) to fit small screens. [THSK01] developed a complementary technique to not only transform content, but also to transform the integrated activity of Web browsing into separate and simpler modes. We have in our system used AvantGo [Ava03] to transduce the web content, but we have also made adaptations of the content itself to fit into the three different types of mobile devices (WAP, PDA, Laptop).

Al-bar et al. [AbW01] discusses adaptation spaces to enable applications to function in different environments. Content adaptation is one space that has the goal to allow users to access the contents they want on any display terminal using any mobile device. User profiles are connected to each device to hold capability information about the device. This information is used to determine the adaptation level. In our project, we have delivered content based on the device type, and have not used any specific profile to identify the device capabilities. Other adaptation spaces are not considered in this paper.

We are aware of the existence of research and systems that handle similar issues that are reported in our paper. Lamming et al. [LEF+00] describes the Satchel system supporting mobile document work. In [VR02], Vantroys et al. describe a workflow system using

mobile devices in coordinating learning activities. [SK01] describes an architecture for executing an agent-based workflow system on PDAs. We envisage our system, and the IT-support scenario to be baselines for a workbench to explore issues related to the system support of mobile work in heterogeneous (wireless) networks, with heterogeneous devices, and in virtual as well as ordinary organisations.

13.7 Conclusion

The investigation of technologies to give fast, reliable, user-friendly, maintainable, and cheap systems, is important to provide services to the mass market with relative small budgets and special needs. One of the goals of our research is to provide extended mobile support for work processes based on scenarios like that from the IT-support group. This paper has presented and evaluated a system to support mobile activities/tasks in mobile environments. To deliver content to mobile devices it is necessary to take into account that the displays, computing, and networking capabilities are different from device to device. It is necessary to make adaptations to every type of mobile device and software in order to get usable results. Usability and effectiveness are of a greater concern in mobile systems compared to desktop systems because of the device limitations. There are further challenges e.g. to provide the necessary security to the system, especially since we have decided to keep the Web-based interface to support all the different kinds of equipment. To improve the work efficiency of mobile workers, we are investigating the possibilities to extend the system with functionality to support planning/assignment of tasks based on the current location of the staff. This functionality can be used to optimise the assignment of the support tasks. This includes finding the most optimal worker to newly arrived tasks based on both their current location and their capability to solve the tasks (in available time and competence). The use of SMS as a communication channel was not tried out in our prototype. We do however envisage that SMS can be used as a polling mechanism to get more information on certain tasks, add new tasks by sending messages to the system, and change the state of specific tasks. This will require an SMS gateway at the server, and a model to be able to parse the messages. SMS could be partly integrated into the client application, but we have not made any attempts to implement this issue.

Acknowledgement

This paper is a result of work in a project called MObile Work Across Heterogeneous Systems (MOWAHS) [MOW04]. The MOWAHS project is sponsored by the Norwegian Research Council's IKT-2010 program.

CHAPTER 14

P5 – Novel Component Middleware for Building Dependable Sentient Computing Applications

Maomao Wu, Adrian Friday, Gordon Blair, Thirunavukkarasu Sivaharan, Paul Okanda, Hector Duran Limon [1] **Carl-Fredrik Sørensen,** [2] **Gregory Biegel and René Meier** [3]

Abstract

With advances in sensor-based computing and mobile communication, people have started to explore ubiquitous or pervasive computing systems that aim to have computing devices literally available everywhere, making them disappear into the physical environment. Novel ubiquitous computing applications such as intelligent vehicles, smart buildings, and traffic management have special properties that traditional computing applications do not possess, such as context-awareness, massive decentralisation, autonomous behaviour, adaptivity, proactivity, and innate collaboration. In this paper we argue that such applications require a new computational model and middleware that can reflect the autonomy

[1]**Department of Computing, Lancaster University, Bailrigg, Lancaster, UK, LA1 4YR.**
{maomao, adrian, t.sivaharan, okanda, gordon, duranlim}@comp.lancs.ac.uk

[2]**Department of Computer and Information Science, Norwegian University of Science and Technology (NTNU), NO-7491 Trondheim, Norway.**
carlfrs@idi.ntnu.no

[3]**Department of Computer Science, University of Dublin, Trinity College, Dublin 2, Ireland.**
{Greg.Biegel, Rene.Meier}@cs.tcd.ie

and spontaneity of cooperative entities. The EU funded CORTEX[4] project proposes the sentient object model to support the construction of such large-scale applications. We report on a flexible, run-time reconfigurable component based middleware that we have used to engineer the sentient object programming paradigm. We demonstrate the appropriateness of the novel computational model and validity of the middleware by constructing a proof of concept demonstrator based on the notion of autonomous cooperating vehicles.

Keywords: *Context-aware, Component, Middleware, Sentient object.*

14.1 Introduction

With advances in sensor-based computing and mobile communication, researchers have started to explore ubiquitous computing (UbiComp) systems [Wei93, Wei02] that aim to have computing devices embedded literally everywhere, while making them disappear into the physical environment (e.g in our cars, buildings, soft furnishings, appliances, clothing etc.). Novel applications are possible in these environments, but many of the scenarios we can envision require elements to operate independently of direct human control. Among the most popular examples of these kinds of application are based around intelligent vehicles, traffic management systems, and smart buildings or working/living environments. We believe that there are a number of special characteristics that differentiate these classes of application from traditional computing applications, such as:

- **Sentience:** These applications are context-aware, i.e. have the ability to perceive the state of the surrounding environment, through the fusion and interpretation of information from possibly diverse sensors.
- **Autonomy:** Components of these applications will be capable of acting in a decentralised fashion, based solely on the acquisition of information from the environment and on their own knowledge.
- **Decentralisation:** There is no single central server that does intensive computation for the clients. Typical applications consists of components that might be scattered across geographical regions, e.g., street, buildings, cities, countries, and continents.
- **Proactivity:** These applications are able to act in anticipation of future goals or problems without direct human intervention. They should have a certain degree of intelligence, and be able to decide what action to take from gathered sensor data.

[4]The CORTEX (CO-operating Real-time senTient objects: architecture and EXperimental evaluation) project is supported by the Future and Emerging Technologies programme of the Commission of the European Union under research contract IST-FET-2000-26031, http://cortex.di.fc.ul.pt/.

- **Adaptivity:** These applications will have to cope with changing conditions during their lifetimes. Not only must the applications be designed to evolve, but their underlying support must be adaptable as well.
- **Time and Safety Criticality:** These applications interact with physical environments and are required to provide real-time services to human users. It is important to provide real-time guarantees and dependability assurance through some system or middleware modules, e.g., resource management and configuration, timing failure detection and Quality of Service (QoS) management.

These characteristics make it extremely challenging for application designers and system engineers to design and implement ubiquitous computing systems and applications. We postulate that an appropriate computational model, programming abstraction and supporting middleware are crucial for realising the vision of UbiComp and assisting in constructing these forms of novel application.

In the EU funded CORTEX [Uni01] project, we propose a programming model that we believe contributes an important abstraction for supporting the construction of these forms of application. In our programming model, applications are constructed from large numbers of software components which accept input (construct their view of the world) via a variety of sensors, and autonomously react, acting upon the environment using a variety of actuators. These components must have a certain level of "intelligence" to allow them to act autonomously based upon this acquisition of information, and should be able to cooperate with each other using a range of different networking technologies. We named these intelligent software components "sentient objects", and define a programming model for the development of such objects.

In this paper we describe our programming model in more detail (section 14.2) and report on our experiences of building middleware to support this model (sections 14.3 and 14.4). Our component based middleware in particular offers a number of important features including flexible, run-time configuration and reconfiguration, and a novel unified mechanism for supporting both context-aware and QoS adaptation. To demonstrate the appropriateness of the paradigm and the validity of the middleware, we have also constructed a proof of concept demonstrator based on the notion of autonomous cooperating vehicles (section 14.5). Section 14.6 contrasts our approach with existing approaches in the area of middleware for ubiquitous computing. Finally, we summarise our experiences in our concluding remarks and propose some future work.

14.2 Sentient Object Programming Model

In the sentient object programming model [BC04, COR03], software entities are categorised into sensors, actuators, and sentient objects. Sensors are defined as entities that

produce software events in reaction to a stimulus detected by some real-world hardware device. An actuator is defined as an entity that consumes software events and reacts by attempting to change the state of the real world in some way via some hardware device. Both of these may be a software abstraction of actual physical devices. A sentient object is then defined as an entity that can both consume and produce software events, and lies in some control path between at least one sensor and one actuator.

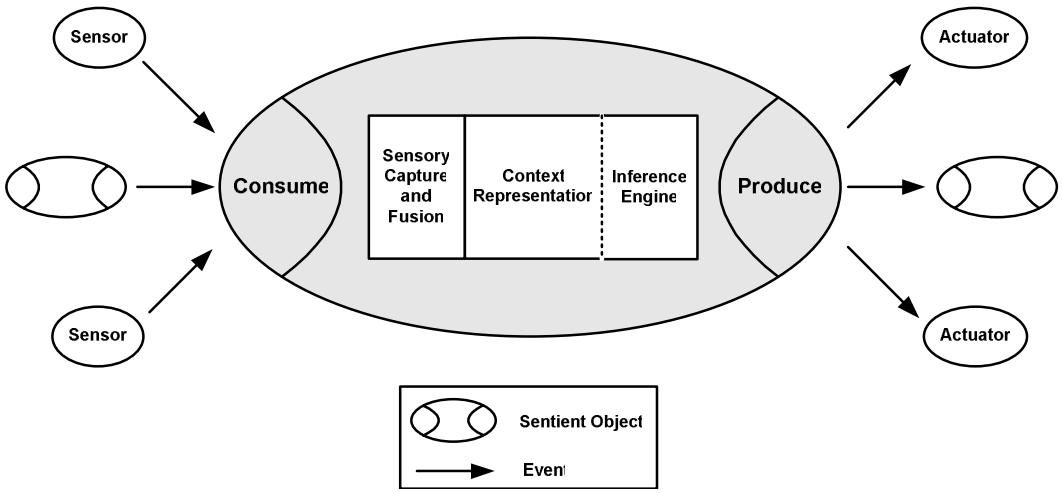

Figure 14.1: Sentient Object Model

Sentient objects are cooperative and communicate with each other and with sensors and actuators via an anonymous generative event based communication paradigm [MC03], permitting loose coupling between objects which supports component reconfiguration and application evolution. A sentient object and its internals are illustrated in 14.1. The novel event-based communication mechanism incorporated in the model is specifically designed for mobile ad-hoc wireless environments such as those typified spontaneous device interactions and networking found in embedded systems (e.g. mobile computing, sensor networking, cooperative vehicles, etc.).

Three main components have been identified [COR03] in order for a sentient object to be context-aware: sensory capture, context representation, and an inference engine. The major issues in the area of sensory capture are data filtering and sensor fusion. Probabilistic sensor fusion techniques can be used to provide more accurate estimates of contexts, and dead-reckoning can be employed when there is insufficient traffic or delays in sensor data. The context representation component deals with the representation of context information in a way that is useful to the sentient object and may be easily exchanged among sentient objects. The inference engine is analogous to the brain of a sentient object, and all the intelligent behaviour of the sentient objects are attributable to the inference engine.

It has the capability of generating high-level context from low-level context or sensor data, and making actuation decisions on how to react based on this contextual data.

14.3 Middleware Design Challenges

Middleware can conceal programming complexity from application developers, and provide services to facilitate communication and coordination of distributed software components. Instead of realising the sentient object programming model directly, we decided to design and implement a flexible middleware so that it can be reused in different circumstances. Some key research challenges we aimed to address in building our middleware were as follows:

- **Context-awareness:** We follow the sentient object paradigm to handle inputs from diverse sources, e.g., different sensors or other sentient objects. Uncertainty is a major problem in sensing the environment due to the inherent limitations of sensors with respect to accuracy and precision. This has led to a crucial requirement for our middleware that it provides uncertainty management for software components whose actions are based on environmental perception. Additionally, "intelligent" software components that reason based on context are required in order to make sentient objects autonomous and proactive.
- **QoS management and fail safety:** Due to the real-time nature of ubiquitous computing applications, the middleware needs to take into account the provision of incremental real-time and reliability guarantees. QoS properties need to be expressed as a metric of predictability in terms of timeliness and reliability. For distributed objects coordinating in uncertain environments, the timing bounds for distributed actions could be violated because of the timing failures. This requires a reliable timing failure detection service for distributed operations.
- **Communication model:** Traditional communication models, such as client-server and the RPC paradigm, are not well suited to mobile ad-hoc environment, because there is no fixed infrastructure to host centralised services. Since disconnections are common in the wireless communication environment, the communication paradigm should be decoupled and asynchronous. Moreover, in novel applications with mobile or context-based elements, the scope of information dissemination is dynamically determined by spatial parameters. For example, in the cooperating cars scenario, one might wish to limit dissemination to those vehicles directly affected by an obstacle on the road, and the information is only valid in a restricted geographical area.
- **Routing in mobile ad-hoc environment:** In mobile ad-hoc networks, the senders and receivers move constantly so that the network topology frequently changes.

This poses a challenge for routing packets in such dynamic environments. Multicast protocols based on proactive and reactive ad-hoc routing, using shared state kept in the forms of routes and adjacent information, is useful in environments with low node mobility. However, this shared state and topology information can quickly become outdated in the highly mobile environments. Hence, it requires a new type of routing protocol in highly dynamic ad-hoc network environments.

We believe middleware support encourages the widespread development of aforementioned novel applications. It is always important for middleware designer to bear in mind the issue of reusability, configurability and reconfigurability. Previous research experience [CBCP02] in reflective middleware also motivates us to use reflection, component technology, and component frameworks to create a flexible and dynamically configurable middleware platform. In the following section we discuss how these challenges have been addressed by the component frameworks that are used to compose our sentient objects.

14.4 Component-oriented Reflective Middleware

14.4.1 Component Model and Component Frameworks

The autonomous feature of sentient objects requires that they adapt based on contextual information, either communicated as high level contexts (e.g. as software events) or derived directly from raw sensor information. To move from low level sensing to discernable contexts requires both conditioning of the data (e.g. fusion and interpolation) and higher order reasoning (rule based inferring). Clearly the demands of a given application and its associated environment will vary over time and situation, both in terms of fine grained adaptation (tuning parameters on a particular fusion algorithm for example), and at a courser grained level (switching behaviours or between information sources). To support this level of configuration and reconfiguration we have chosen to build our middleware based on a reflective component model called OpenCOM [CBCP02]. This technology allows use to introspect the running system and adapt any aspect of the system at run-time, permitting us ultimate flexibility.

OpenCOM is built atop a subset of Microsoft's COM. Ignoring COM's advanced features, OpenCOM keeps its core aspects including the binary level interoperability standard, Microsoft's IDL, COM's globally unique identifiers and the IUnknown interface as the basis of its implementation. The fundamental concepts of OpenCOM are interfaces, receptacles and connections (bindings between interface and receptacles). An interface expresses a unit of service provision while a receptacle describes a unit of service requirement. OpenCOM deploys a standard run-time to manage the creation and deletion of components, and to act upon requests to connect and disconnect components. Moreover, a system graph of

the running components is maintained to support the introspection of a platform's structure.

Component Frameworks (CFs) were originally defined in [Szy97] as "collections of rules and interfaces that govern the interaction of a set of components plugged into them". CFs constrain the design space and scope for evolution, so that they are targeted at a specific domain and embody "rules and interfaces" that make sense in that domain. To realise sentient objects, the OpenCOM component model has been employed to construct families of middleware, each of which is in turn created as a set of configurable CFs. We have designed and implemented a number of CFs, including a Publish-Subscribe event channel CF, a Context CF, and a Resource and a QoS CF. As a result, reflection can be used to discover the current middleware configuration and behaviour, and middleware configuration can also be changed at run-time.

14.4.2 Publish-Subscribe CF

The Publish-Subscribe (P-S) CF is a componentised prototype of the STEAM P-S system [MC03]. STEAM is based on an implicit event model and has been designed for mobile applications and ad hoc networks. STEAM differs from other P-S systems in that it does not rely on the presence of any separate infrastructure and supports distributed techniques for identifying and delivering events of interest based on location. STEAM supports a decentralised approach for discovering peers, for routing event notifications using a distributed addressing scheme, and for event filtering based on combining multiple filters. Filters may be applied to the subject and the content of events, and may be used to define geographical areas within which events are valid. Such proximity-based filtering represents a natural way to filter events of interest in mobile applications. The P-S CF support both publisher and subscriber side filtering, by using a query or subscription language called Filter Event Language (FEL) [Siv02] to express their preferences. FEL can be used to create subject, content, and context filters, which are also componentised and can be dynamically reconfigured.

Publishers and subscribers are anonymous, and subscribers should be able to interpret the events without a priori knowledge. Therefore, we have the need for a generic event dialect which can be understood by all publishers and subscribers in the system. The events in the P-S CF are represented in XML, and a generic XML profile defines the generic event dialect. The XML based events provide for easy interoperability and extensibility of the event dialect.

Events are published to a notional event channel allowing one-to-many distribution of events. The event channel is supported by an underlying communication mechanism supported by a group abstraction. The Group Communication CF provides a range of group communication protocols. The Publish-Subscribe CF can flexibly select a group commu-

nication protocol from Group Communication CF, which currently supports a probabilistic ad-hoc multicast protocol, an IP multicast based protocol and a local (shared memory based) group communication protocol. The P-S CF is used to construct various event channels, which are required by the Context CF (as describe below) and inter-sentient object communication.

14.4.3 Context CF

The Context CF consists of two parts: sensor capture and fusion, and the inference engine.

Sensor Capture and Fusion

The sensory capture and fusion components are able to receive sensor data through an event channel and perform some data fusion algorithm in order to manage uncertainty of sensor data and derive higher level context information from multi-modal data sources. We have explored three different techniques for our middleware component design.

One of the widely-used methods in sensor fusion and machine learning is the Gaussian modelling and multivariate Gaussian modelling [vVS+03]. Raw sensor data samples associated with certain target values are gathered first to establish some multivariate Gaussian distribution functions [5]. We collected ultrasonic sensor' readings at certain target values, i.e. various distances from the sensors to the obstacle ahead. When new sensor readings arrive they are fed into the established Gaussian distribution functions. The outputs of these functions are compared, and the target value associated with the distribution function generating the highest output is the most probable target value. In the ultrasonic sensor example, we can obtain the most probable distance to the obstacle from the current sensor readings by feeding the current sensor readings to the various Gaussian distribution functions associated with different target values. Multivariate Gaussian is the one of the most important methods to get the best statistical target value for current sensor readings.

Another probabilistic sensor fusion scheme is also employed, based upon Bayesian networks [Jen01], which provides a powerful mechanism for measuring the effectiveness of derivations of context from noisy sensor data. A Bayesian network is a directed acyclic graph in which nodes represent random variables and the absence of arcs represents conditional independence. A Bayesian network graph allows us to manage the exponential increase in the size of a joint probability distribution over a set of random variables, by ex-

[5] A Gaussian distribution is also known as a normal distribution, which has the form of

$$G(x;\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^d|\Sigma|}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$$

where μ is the mean or average vector, and Σ refers to the covariance matrix

ploiting conditional independence: $P(X_1, ..., X_n) = \prod_{i=1}^{n} P(X_i|pa(X_i))$ where $pa(X_i)$ are the parents of node X_i and $P(X_i|pa(X_i))$ is the conditional distribution of X_i given its parents. Using Bayesian networks to model the uncertainty of sensor data and the dependencies between a set of sensors, gives us the ability to efficiently reason that a hypothesis is true, given available evidence from sensors.

Finally, dead-reckoning has been used to compensate for insufficient data and latency in transmissions. Instead of relying on the latent current sensor readings for fusion, historical data can be processed during run-time to generate more reliable and real-time values. This technique is especially useful for sensors, such as GPS, that may have unpredictable delays. In the case of GPS, objects should be able to calculate their own dead-reckoned position and update their own trajectory using extrapolation algorithms to make up for this latency. They could compute their current position based on predictive algorithms that have a basis on a history log such that there is continuity and timeliness in otherwise sporadic and delayed data from a source.

Inference Engine

The inference engine is a key part of the sentient object paradigm, and the intelligence of a sentient object is realised by the inference engine and its associated knowledge base. An inference engine in artificial intelligence terms, refers to a program that reasons about a set of rules (a knowledge base) in order to derive an output [COR03]. An inference engine is actually a part of an expert system, which also contains a domain-specific knowledge base. The knowledge base contains the knowledge required to solve a certain problem, encoded as a set of production rules. As a result, contexts can be represented and stored as some facts [Ril03] within the inference engine.

We chose the well-known rule-based inference engine CLIPS — C Language Integrated Production System [Ril03] as a base of our middleware component design. CLIPS provides a cohesive tool for handling a wide variety of knowledge with support for three different programming paradigms: rule-based, object-oriented and procedural, and the internal implementation of CLIPS is based upon the RETE net [For82]. Using the rule-based programming paradigm provided by CLIPS, we can specify rules to generate high-level contexts from fused sensor data or derived low-level contexts. A CLIPS rule consists of two parts: an *if* part and a *then* part. The if part of a rule is a series of patterns which specify the facts (or data) that cause the rule to be applicable. The then part of a rule is the set of actions to be executed when the rule is applicable. Here is an example of CLIPS rule named will be triggered when obstacle distance is near. The if part of the rule contains two facts, “car-id” and “obstacle”. CLIPS automatically matches facts against the patterns specified in the if part of the rules, and executes the actions (then part) of applicable rules. The sample rule activates two actions: “retract” or delete the fact referenced and call user defined function “publish”. CLIPS also provides ways for the programmers to

define their own functions, which can also be specified in the rules. This makes it possible to automatically perform actuation when certain contexts are derived from the inference engine. The user defined function "publish" in the sample actually makes use of the P-S CF to publish the "stop" command to the car speed control event channel. This use of inference on the basis of contextual information makes it possible to do Context-based Reasoning (CxBR) [GA99].

```
(defrule rule-obstacle-near "CLIPS rule for obstacle near"
    (car-id (id ?id))
    ?f1 <- (obstacle (distance near))
  =>
    (retract ?f1)
    (publish ?id stop)
)
```

One of the important features of our approach is that the paradigm facilitates uniform treatment of both context and QoS. End-to-End QoS violations are provided by the Timely Computing Base (TCB) [CV01]. The TCB provides the facility to monitor timeliness of event delivery on distributed event channels, thus providing estimations and awareness of timing failure probability for a given required stability coverage. The detection of timing failures by TCB can be used to trigger the timely execution of fail safe-procedures and adaptation strategies. QoS rules can be included in the inference engine that trigger adaptations based on changes in measure QoS or coverage stability from the TCB (see the following sample rule below that slows the vehicle in our demonstrator when intra-vehicle communication becomes too unpredictable).

```
(defrule rule-network-coverage-bad "CLIPS rule for network coverage bad"
    (car-id (id ?id))
    ?f1 <- (network-coverage (value bad))
  =>
    (retract ?f1)
    (publish ?id slow)
)
```

Note that this uniform approaches allows QoS to become part of context descriptions. Moreover, both changes in context and/ or changes in QoS can trigger adaptations and actuations.

We have wrapped the CLIPS into a DLL and created OpenCOM components for both WinNT and WinCE. This component can perform adaptation at the rule-level (by loading and unloading different rules at run-time).

14.5 Autonomous Cooperative Vehicles Application

To demonstrate the appropriateness of the sentient object model and validity of the componentised middleware, we constructed a proof of concept demonstrator based on the notion of autonomous cooperating vehicles. The vehicles have the objective of travelling on a given path (a virtual circuit), predefined by a set of GPS waypoints. Each vehicle needs to build a real-time image of its surrounding environment within some bounded error to make informed decisions regarding its next move. The cooperation between vehicles is critical to avoid collisions, to follow a leading vehicle and to travel safely. The vehicles also need to obey external traffic signals and give way to pedestrians who cross the road by sensing their presence, and adapt to QoS data such as network coverage stability.

We have built a small number of autonomous cooperative vehicles by modifying remote controlled robot cars and augmenting them with multiple sensors and driven them by our software running on iPAQ PocketPC2002s. A GPS receiver is used to sense location, a digital compass is integrated to sense direction, and eight units of ultrasonic sensors are fixed to detect the presence of neighbouring physical objects. The PocketPC has two WaveLAN cards configured in ad-hoc mode. One is exclusive to the TCB control channel, and the other is for the event channels (payload) used for inter-vehicle communications. Each vehicle uses an onboard Controller Area Network (CAN) [Rob05] to connect the hardware devices, e.g, sensors, actuators and iPAQ. The onboard network is a bespoke ring topology with single break failure resilience, and the design enables addition of further devices on to the ring in a plug and play fashion.

The demonstrator application — "autonomous cooperating sentient vehicles" is implemented using instances of our middleware CFs, and the detailed configuration of the software components is illustrated in Figure Figure 14.2. Different sensing service components are responsible for receiving related software events from a particular sensor and fusing the sensor data. These sensing service components make use of data fusion algorithm components, which can be plugged in and out at run-time. For example, the obstacle distance sensing service can dynamically swap between two different ultrasonic sensor fusion components: Gaussian modelling and a customised fusion algorithm. Context information is derived from these fusion components, and is fed into the inference service component. The inference service component makes use of the CLIPS inference engine to derive higher level contexts and decide actuation actions. The CLIPS rules are predefined in CLIPS script files that can be dynamically loaded into and unloaded from the inference engine. We have three sets of independent rules to do speed control, steer control of the vehicle, and derive network coverage stability context. By matching the current contexts to the actuation rules, the inference engine component decides how to actuate the vehicle by transmitting software events to actuators, which in turn interact with the hardware to fulfill the task.

The rulesets can send special adaptation "commands" using the event channel to trigger

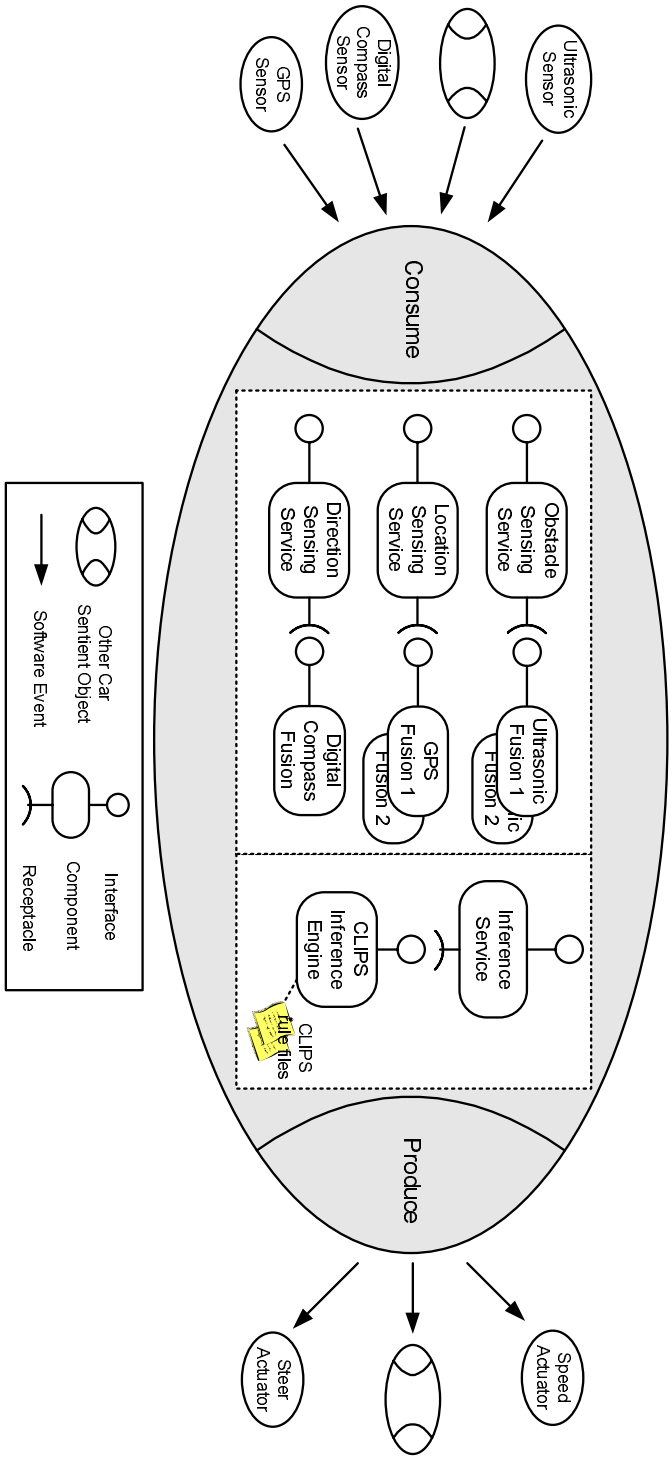

Figure 14.2: Middleware Configuration for Autonomous Cooperative Vehicles Application

middleware component level adaptation (both swapping components to change algorithm and finer grained internal tuning of algorithms) and CLIPS rule level adaptation (which changes behaviour of the system). We have found this simple yet elegantly consistent approach to context and QoS adaptation provides us with with a great deal of expressive power and flexibility in our demonstration.

14.6 Related Work

A tremendous amount of activity, particularly from the car manufacturers themselves, has been taken under the former category of application. In BMW's ConnectedDriv [New03], research vehicles can communicate with each other by using ad-hoc network networking technology, e.g., Wireless LAN. Each vehicle acts as a sensor registering and monitoring the road traffic, and it can recognise congestion traffic and transmit a local traffic report to other vehicles in the nearby area, requiring no human intervention. Motorola laboratory in Arizona, US, Delphi Safety & Interior Systems in Michigan, US and DaimlerChrysler research centre in Ulm, Germany [Jon02] are all trying to build very similar "smart cars" that will make the road travel safer. By monitoring the driver's physical status, such as body movements, eye-blink pattern, and respiration, as well as driving behaviour, e.g., the number and the severity of steering corrections, smart cars are proposed to be able to either alert the driver by some means, e.g., sound or vibration, or even take over some tasks from the driver, such as braking or steering. The Intelligent Transportation Systems (ITS) program at the General Motors [Ash98] is more ambitious, which proposes to harmonize traffic flow, e.g., reducing speed fluctuations and traffic shock waves, and maximising highway capacity by not only constructing closely coordinated vehicles but also developing automated highways. Such complex real time dependable applications are difficult to engineer using traditional systems development techniques and paradigms. We believe the sentient object paradigm may highlight a useful approach that can be adopted by the developers of systems such as these. Moreover, our paradigm offers intrinsic scope for flexibility, reuse and reconfiguration to offer new behaviours and applications.

Different approaches of reflective middleware in Ubicomp have been investigated, e.g., Gaia and UIC. Gaia [RHC+02] is a metaoperating system built as a distributed middleware infrastructure that coordinates software entities and heterogeneous networked devices contained in a physical space. Gaia supports development and execution of portable applications in *active spaces*. Active spaces are programmable ubiquitous computing environments in which users interact with several devices and services simultaneously. Users, services, data, and locations are represented in the active spaces, and are manipulated dynamically and in coordination. Context is important to Gaia's applications, which have access to context via a "context file system". However, context is externalised and is not a first class entity that drives the behaviour of the active space systems. We believe that active space applications could easily be modelled as sentient objects and potentially

benefit from our paradigm.

Reflective middleware has been adopted by other projects interested in adaptation and reconfigurability. UIC (Universally Interoperable Core) [Ubi03] is targeted to mobile devices as a minimal reflective middleware. Heterogeneity issues are solved by UIC in a set of pluggable components that allow developers to specialise the middleware to different devices and environments. The configuration of the middleware can be updated both at compile- and run-time.

The Mobile Platform for Actively Deployable Service (MobiPADS) system [CC03] is another reflective-based middleware designed to support context-aware processing by providing an execution platform to enable active service deployment and reconfiguration of the service composition in response to environments of varying contexts. MobiPADS supports dynamic adaptation at both the middleware and application layers to provide flexible configuration of resources to optimize the operations of mobile applications. To alleviate the adverse conditions of a wireless environment, services (known as mobilets) are configured as chained service objects to provide augmented services to the underlying mobile applications. The reflective model in MobiPADS provides metainterfaces to make applications able to directly participate in computation adaptation in response to the changing context. A mobile application can access contextual information, the service configuration and adaptation strategy, and examine and modify these entities to obtain optimal service provision through the metalevel object representation of the internal event system and service reconfiguration mechanism.

Also other approaches have been developed that focus on meta-data representation for services and application-dependent policies. Capra et al. [CEM02] have e.g. developed mechanisms for dynamic adaptation and conflict resolutions. Several approaches to data-sharing middleware for mobile and ad hoc environments have been developed the last decade. Many of these are based on tuple spaces (e.g. LIME [PMR00], TOTA [MZ04], and L^2imbo [DFWB98]). The approaches have in common that they aim to let data be de-coupled and de-centralised. Data sharing though XML-based structures have been proposed in e.g. XMIDDLE [MCZE02] to support more complex data models than what a tuple represent.

14.7 Concluding Remarks

This paper has described a novel componentised reflective middleware for engineering dependable sentient computing applications based on the sentient object model. A proof of concept demonstrator based on the notion of autonomous cooperating vehicles has been fully implemented to investigate the appropriateness of the novel computational model and validity of the middleware. Based on the experience of designing such a flexible middleware and engineering the fully-fledged demonstrator, we can identify a number of

key results:

- The sentient object model has proved to be an excellent programming abstraction for the development of real-time, cooperative, context-aware applications, particularly because of its intrinsic support for decoupled event channel communication and context-awareness. The demonstrator shows it can be used to handle both context adaptation, e.g., responding to the traffic light signals, and QoS adaptation, e.g., reacting when network coverage stability is low, in a self-consistent and uniform way.
- OpenCOM component model and CFs are the two key enabling technologies underpinning the construction of the middleware. The component-oriented approach for engineering the middleware offers benefits of flexible configuration and reconfiguration of the middleware components, supporting the development of context and QoS adaptive applications. For example, the sentient object instance constructed from CFs can dynamically reconfigure its internal architecture by responding to events from the event channels, e.g, the sensor fusion service components can plug out and in different sensor fusion components at run-time.
- The middleware architecture also provides the management of non-functional concerns such as timeliness and reliability properties. The integration of these modules is greatly eased by following the sentient object model. For example, the data from TCB is distributed using the anonymous communication paradigm, and it in turn goes through the inference engine to derive high-level network coverage stability context and trigger actuation events being published to the car speed control channel.

We believe that our middleware is reusable, and we are keen to investigate the generality of our approach by applying our middleware to other application domains involving embedded autonomous components. This is a key aspect of our future work.

Acknowledgements

We would like to thank Joe Finney and Angie Chandler for the construction of the robot cars augmented with a token ring network with an array of sensors. Trimble Component Technology Europe also supported the construction of the demonstrator by providing us free GPS development kits.

CHAPTER 15

P6 – A Context-Aware Middleware for Applications in Mobile Ad Hoc Environments

Carl-Fredrik Sørensen[1], **Maomao Wu, Thirunavukkarasu Sivaharan, Gordon Blair, Paul Okanda, Adrian Friday and Hector Duran Limon**[2]

Abstract

Novel ubiquitous computing applications such as intelligent vehicles, smart buildings, and traffic management require special properties that traditional computing applications do not support, such as context-awareness, massive decentralisation, autonomous behaviour, adaptivity, proactivity, and innate collaboration. This paper presents a new computational model and middleware that reflect support for the required the properties. The sentient object model is proposed by the CORTEX [3] project to support the construction of ubiquitous applications. A flexible, run-time reconfigurable component-based middleware has been built to provide run-time support to engineer the sentient object programming paradigm.

[1]Department of Computer and Information Science, Norwegian University of Science and Technology (NTNU). NO-7491 Trondheim, Norway

[2]Computing Department, Lancaster University,Bailrigg, Lancaster, LA1 4YR, U.K. Tel: +44 (0) 1524 593315

[3]This paper is part of the CORTEX (CO-operating Real-time senTient objects:architecture and EXperimental evaluation) project. The CORTEX project is supported by the EC, through project IST-FET-2000-26031. http://cortex.di.fc.ul.pt

An application infrastructure using sentient objects to enable cooperation between autonomous and proactive vehicles has been implemented to demonstrate the appropriateness of the computational model and the validity of the middleware for pervasive mobile ad hoc computing.

Categories and Subject Descriptors
D.2.2 [**Software Engineering**]: Design Tools and Techniques– *Modules and interfaces*; C.2.4 [**Computer-Communication Networks**]: Distributed Systems–*Distributed applications*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design– *Wireless communication*

Keywords: *Middleware, components, context-awareness, sentient objects, ad hoc wireless network*

15.1 Introduction

The technical evolution in wireless networks, mobile and sensor technology has made the vision of ubiquitous computing coming closer to reality. Autonomous applications that in a proactive manner operate independently of direct human control are increasingly being developed to provide a more seamless and invisible support to the users. One of the challenges of this evolution has been to create ”intelligent” middleware and to design appropriate computational models for this new generation of applications. Such middleware should address the basic challenges raised by wireless mobile ad hoc network (MANET) applications, and support the growth and adaptability to new technologies, and provide applications with ways to enforce non-functional quality attributes like reliability and timeliness.

Academia and industry are both showing interest in research to exploit the new possibilities opened by the technological evolution. Car manufacturers, e.g., have initated projects to develop similar applications as investigated in this paper, e.g., the ConnectedDriv project by BMW [New03] and the Intelligent Transportation Systems (ITS) program at the General Motors [Ash98]. Such complex real-time dependable applications are difficult to engineer using traditional systems development techniques and paradigms.

The CORTEX project has identified the key characteristics that the new generation of applications may possess: **Sentience** – the ability to perceive the state of the surrounding environment; **Autonomy** – components should be capable of acting in a decentralized and autonomous fashion; **Time criticality** – applications will typically interact with the physical environment, and will have to cope with timeliness constraints; **Safety criticality** – typical applications will interact with human users, whose well-being will frequently rely on them; **Geographical dispersion** - typical applications will integrate components

that are scattered over buildings, cities, countries, and continents; **Mobility** – applications residing in mobile devices must possess the ability to physically move and discover new neighbours and interact and share information; **Evolution** – applications will have to cope with changing conditions during their lifetimes.

It is unlikely that any application at the same time will possess all these characteristics. The most challenging applications are probably those that mix autonomy of application participants, derived from sentience, with the need to maintain a consistent view of the application environment while cooperating with other participants.

In this paper, we present a middleware platform which addresses the challenges raised by the application domain involving autonomous mobile physical objects that cooperate with other objects, either mobile or static, by capturing information in real-time from sensors event messages propagated in a MANET.

The rest of the paper is organised as follows: Section 15.2 presents the application domain, Section 15.3 presents the sentient object model, and Section 15.4 presents the component framework and the middleware developed. Section 15.5 presents related work, and finally, Section 15.6 concludes the paper.

15.2 Cooperating Cars

To investigate the unique challenges posed by ubiquitous applications built upon MANETs, we decided to build an application based on intelligent vehicles. This facilitates to investigate the feasibility and appropriateness of the programming model and middleware platform based on a concrete prototype application. Using real hardware for the demonstration ensures that our proposals are directly usable and do not depend on hidden idealistic assumptions. Moreover, the intelligent vehicle applications are an active research area. Cars are increasingly being equipped with different types of embedded sensors ranging from position and obstacle sensors to sensors indicating road and weather conditions. Cars with embedded sensors thus provide a rich set of context information. By making the sensor information available to other cars, a car can derive certain traffic conditions directly from the provided sensor information. Car-to-car communication thus has three major goals: *1. Dissemination of traffic information derived from the embedded sensors*; *2. Cooperation of cars to assist the driver in critical situations*; and *3. Interaction between remote cars*.
Traffic scenarios are inherently safety critical. An ultimate level of predictability and safety is particularly necessary in the tight coordination tasks in and between the cars. To model various forms of the aforementioned cases and to validate the middleware platform using real hardware, we have constructed a small number of autonomous robot cars. The

robot cars have been instrumented with GPS[4], compass, and ultra-sonic sensors[5] and it is controlled by an iPAQ mounted on the car. Sensor information is collected by the iPAQ and used to provide actuations, e.g., change of course, reduction of speed, and in the case of obstructions or other safety critical events, to stop the car. Relevant information is notified to other cars using the ad hoc mode of IEEE 802.11b WLAN network.

15.3 Sentient Object Model

The CORTEX project proposes the sentient object (SO) programming model for pervasive and ad hoc computing applications. The SO programming abstraction allows developers to design distributed applications in terms of sentient objects, instead of decomposing them into components parts such as messaging protocols, sensor fusions, context representation and inference engine. Those applications that have already been built has shown to facilitate good application design and hide the complexity from the application designer. The cooperating car demo application is designed using the sentient object model as described in [FBCC02]. A sentient object (SO) is a mobile (not mobile code), intelligent software component that is able to sense its environment by consuming events via event channels from sensors and/or other SOs, fuse the sensor data and derive higher level context, perform context-based reasoning using some control logic and actuate physical aspects by publishing events via event channels to respective actuators. SOs are context-aware; aware of both their internal state and the state of their local environment; and are cooperative by exchanging higher level context information via event channels.

15.4 Component Framework of Sentient Objects

Sentient objects (SO) provide an abstraction for very common behaviour in pervasive ad hoc computing: Sensing and viewing the behaviour of neighbour entities and based on a RT-Image of the environmental context, reason about it, and actuate environmental objects. E.g., a car modelled as a SO can view the location of its nearest car or a traffic light signal state, reason about it, and then actuate or alter its navigation. This behaviour involves spontaneously discovering neighbour nodes and interacting with them to share context data, and possibly to cooperate with each other to attain common goals (e.g., cars cooperate to avoid collisions). The SOs capture useful, light weight mechanisms that match the essence of many distributed computing mechanisms in pervasive ad hoc computing domain. For the application designer to address a specific domain in terms of SOs, an underlying goal of the middleware is to provide a suitable interface that provides

[4] A Trimble Lassen LP GPS module is used in our cars.

[5] Thanks to Joe Finney and Angie Chandler for construction of the cars augmented with a token ring network and an array of sensors.

the basic functionality identified in Section 15.3. We have designed and implemented a component framework-based middleware platform to provide the run-time support for the SOs to fulfil this goal. The middleware consists of several component frameworks (CF) where each of the CFs addresses certain research areas. The *Publish-Subscribe CF* is used for discovery of mobile entities in its proximity, communication and data sharing data among distributed entities forming MANETs. The *Group communication CF* is used to provide a group communication protocol suite that provides the support to route events in mobile ad hoc networks. The *Context CF* provides the facility for sensor fusion and the inference engine. The *QoS management CF* arbitrates the allocation of resources and provides facilities for monitoring and adaptating the QoS. The *Timely Computing base (TCB)* [CV01] is used to monitor the QoS of the event channels operating over MANETs; The goal of the middleware platform is to provide support for CORTEX applications and to support the sentient object (SO) abstraction. Figure 15.1 shows the SO model of the cooperating car demo application. In this model, in each car, the external environment is sensed by different sensors and consumed by the SOs as sensor readings. The execution platform (iPaq) and the wireless network are in the model also regarded as external environments by the SOs. The external environment produces environmental events that can be captured by the sensors. The sensor readings are sent as internal events to the subscribing SOs. The SOs can also subscribe to events from other SOs. Such events are received as external events. Two types of SOs have been modelled in our cooperating car application: The *car SO* and the *QoS Management SO*. Both receive events from "sensors" or virtual sensors, and actuate by sending internal events to the other SOs which would trigger adaptation of a certain component configuration, and to the actual actuators (e.g. the speed control).

Previous experiences in construction of reflective middleware [BCA+01] have motivated us to use reflection, component technology and component frameworks (CF) to create a middleware platform for MANETs. In pervasive mobile ad hoc networks, a dynamic environment is the norm rather than the exception. This mandates the middleware platform to be deployment time configurable and run-time reconfigurable. Clearly the demands of a given application and its associated environment will vary over time and situation, both in terms of fine grained adaptation (tuning parameters), and at a courser grained level (e.g. switching components). To support this level of configuration and reconfiguration, we have chosen to build our middleware based on the reflective component model OpenCOM [CBCP01]. This technology allows introspection of the running system and adaptation of any aspect of the system at run-time, permitting ultimate flexibility. OpenCOM is based on Microsoft COM, and is adaptive and reconfigurable. The middleware platform is implemented for Windows CE

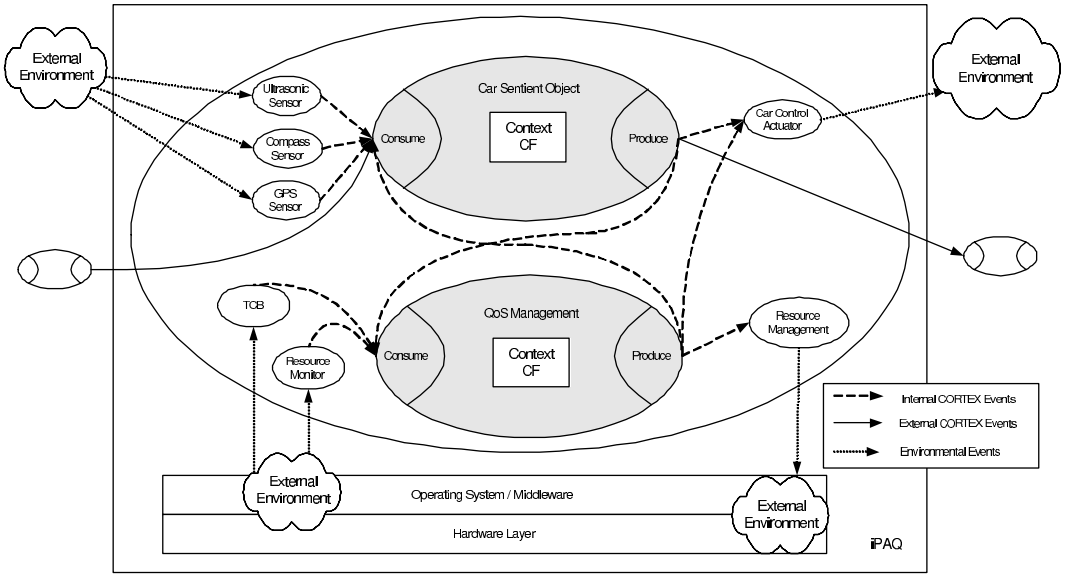

Figure 15.1: The Sentient Object Model of the Car Scenario

15.4.1 Context CF

The *context CF* in Figure 15.1 provides the functionality of the control engine of a sentient object and consists of a sensor fusion and an inference engine component. It offers gaussian modelling algorithms and dead-reckoning functionality. The *context CF* subscribes to events from the sensors and possibly from other sentient objects, and fuse the multi-source sensor data and derives higher level context, the inference engine constitutes the expert logic which reason about context and take autonomous decisions. The decisions are in-turn published to respective actuators and sentient objects.

Sensory Capture and Fusion

Uncertainty is a major problem while sensing the environment due to the inherent limitations of sensors with respect to accuracy and precision. Several methods have been explored to handle this problem. Statistical methods, such as Gaussian modeling and maximum likelihood estimation can give more accurate estimation of target values from the imprecise raw sensor data; dead-reckoning can be used if there is insufficient traffic or delays from the sensor capture; Bayesian networks are also proposed to do sensor fusion and to measure the effectiveness of other context derivation methods given inherently uncertain and noisy sensor data. The combination of low-level sensors to a sensor detecting more complex events is denoted as sensor fusion, the resulting sensor is called a virtual sensor. It should be noted that the correlation of multiple sensors has to be performed in the receivers, which requires the general availability of sensor information.

A widely-used method in sensor fusion and machine learning is the Gaussian modelling and multivariate Gaussian modelling. Raw data samples from the sensors at certain target values need to be gathered to establish some multivariate Gaussian distribution functions. Ultrasonic sensor readings were collected at certain target values, i.e. various distances from the sensors to the obstacle ahead. When new readings arrive while the car moves, they are fed into the established Gaussian functions, and the highest value of the current readings will be the most probable target value. We can thus obtain the most probable distance to an obstacle using Gaussian modelling. This technique was used in the car demo to detect pedestrians and non-event publishing obstacles within 3m range from the car. The ultrasonic sensors on the robot car had the capability to detect objects within 3m range.

Dead-reckoning techniques were used to compensate for insufficient data and latency in transmissions. Instead of relying on the latent current sensor readings for fusion, historical data is processed during run-time to generate more reliable and real-time values. This technique was especially useful in the car application for sensors, such as GPS which have unpredictable real positional update delays. The cars were able calculate their own 'dead-reckoned' position and update their own trajectory using extrapolation algorithms to compensate for the GPS latency.

Inference Engine

The autonomous behaviour of a SO is supported by an instance of an inference engine embodied in the SO. Low-level context, either directly derived from sensor readings or received through the event channels is fed into a rule-based inference engine to derive high-level context according to some rules specified in the rule-base inside the inference engine. When high-level context has been derived, the conditional rules (specified by the application programmer) are used to reason about the context and execute external functions that have been defined by the user to do e.g. actuations. Context data is represented as "facts" within the CORTEX inference engine. Internally, SOs can interact with the inference engine by asserting and retracting these facts. A rule-based inference engine CLIPS - C Language Integrated Production System [Ril03] has been chosen as a base for the inference engine. CLIPS provides a cohesive tool for handling a wide variety of knowledge with support for three different programming paradigms: rule-based, object-oriented and procedural. Using the rule-based programming paradigm provided by CLIPS, rules can be specified to generate high-level context from fused sensor data or derived low-level context. CLIPS also provides ways for the programmers to define their own user functions, and these functions can also be called from the CLIPS rules script file. This makes it possible to automatically perform actuations when a certain context is derived from the inference engine. One of the important features of our approach is that the paradigm facilitates uniform treatment of both context and QoS data. E.g, CLIPS script based rule files are written by the application programmer to control the navigation

of the cars in the Car SO. Similarly QoS management policies are written in CLIPS script file for the QoS management sentient object.

15.4.2 Publish-Subscribe CF

The key to the implementation of a sentient object is the publish-subscribe (P-S) communication model used for both discovery of neighbouring nodes and data sharing. When a SO wants to share its context data, the data is published using the P-S communication model. The receiving SOs subscribe to the events of interest. The P-S CF takes care of event routing and event filtering from a publisher to subscribers. This decouples the publisher and the subscribers, they are anonymous and the communication style is asynchronous. These properties are well suited for MANETs. However, the state-of-the-art P-S middleware are based on centralised event brokers that mediate between publishers and subscribers. The MANETs cannot support centralised fixed servers or fixed system wide services. Thus, we have specifically designed the P-S CF for MANETs. The design was inspired by STEAM [MC02]. The P-S CF is based on an implicit event model. The P-S CF differs from other P-S systems in that it does not rely on the presence of any separate infrastructure and supports distributed techniques for identifying and delivering events of interest based on location. The P-S CF support both publisher and subscriber side event filtering, by using a query or subscription language called Filter Event Language (FEL) to express the preferences. FEL can be used to create subject, content, and context filters, which are also componentised and can be dynamically reconfigured. In the implicit event model, there are thus no event brokers or mediators; instead event filtering functions are implemented at both the subscriber and the publisher side in a decentralised manner.

Publishers and subscribers are anonymous, and subscribers should be able to interpret the events without a priori knowledge of the exact event data structure. A generic event dialect is therefore needed, which can be understood by all publishers and subscribers in the system. The events in the P-S CF are represented in XML, and a generic XML profile defines the generic event dialect. The XML based events provide for easy interoperability and extensibility of the event dialect. Events are published to a notional event channel allowing one-to-many distribution of events. The events are routed from publisher(s) to subscriber(s) using multicast protocols. The multicast protocols are addressed in the *Group Communication (GC) CF*.

The *GC CF* provides a range of multicast protocols. The P-S CF can flexibly select a group communication protocol from the GC CF, which currently supports a probabilistic ad hoc multicast protocol, an IP multicast based protocol, and a local (shared memory) group communication protocol. The probabilistic ad hoc multicast protocol was implemented for MANETs with high node mobility, where keeping shared routing information within the nodes is unsuitable as the network topology is rapidly changing.

15.5 Related Work

Different approaches of middleware in UbiComp have been investigated, e.g., Gaia and MobiPADS. Gaia [RHC+02] is a metaoperating system built as a distributed middleware infrastructure that coordinates software entities and heterogeneous networked devices contained in a physical space. Gaia supports development and execution of portable applications in *active spaces*. Users, services, data, and locations are represented in the active spaces, and are manipulated dynamically and in coordination. Context is important to Gaia's applications, which have access to context via a "context file system". However, context is externalised and is not a first class entity that drives the behaviour of the active space systems. We believe that active space applications could easily be modelled as sentient objects and potentially benefit from our paradigm.

The MobiPADS system [CC03] is a reflective-based middleware designed to support context-aware processing by providing an execution platform to enable active service deployment and reconfiguration of the service composition in response to environments of varying contexts. MobiPADS supports dynamic adaptation at both the middleware and application layers to provide flexible configuration of resources to optimize the operations of mobile applications. The reflective model in MobiPADS provides metainterfaces to make applications able to directly participate in computation adaptation in response to the changing context. A mobile application can access contextual information, the service configuration and adaptation strategy, and examine and modify these entities to obtain optimal service provision.

15.6 Conclusion

In this paper, we have presented a middleware for mobile ad hoc environments. The prototype implementation of the autonomus car application scenario provides a very rich application domain to explore context-awareness. The sentient object programming model has proven to be very usable as a programming abstraction, for development of such applications, particularly because of the intrinsic support for context-awareness. The context reasoning engine and middleware developed have successfully been used in an autonomous car application in a real mobile ad hoc wireless environment. There is a need for middleware in this area to ease the burden on the application developer and also to provide support for the management of non-functional concerns such as timeliness properties. The properties of configurability and re-configurability inherent in our approach are highly suited to this domain, for example to select configurations suitable for a given embedded device and also to encourage the construction of adaptable or autonomic systems. The middleware developed is reusable and may potentially also be (re)used in other application domains like smart buildings and retail systems. Investigating the generality

and application of the approach in other domains is interesting for future work.

CHAPTER 16

P7 – Mobile Peer-to-Peer Technology used to Promote Spontaneous Collaboration

Alf Inge Wang[1] , **Carl-Fredrik Sørensen**, and **Thomas Fossum** [2]

Abstract
Mobile devices have in the recent years been able to form peer-to-peer networks making it possible for people to share information and interact. This technology makes it possible to create new tools that initiate and improve human collaboration. The paper presents experiences from creating a prototype for spontaneous collaboration, supported through the appliance of peer-to-peer applications on wireless mobile devices with the ability to form ad-hoc wireless networks. The paper gives an example of an application that promotes human collaboration by using peer-to-peer technology, and describes the technological challenges faced when implementing such applications. The paper reflects on some user tests that were carried out using this prototype.

Keywords: Spontaneous collaboration, peer-to-peer networks, mobile environments.

[1]Dept. of Computer and Information Science, Norwegian University of Science and Technology (NTNU), N-7491 Trondheim, Norway. Phone: +47 73 594485, Fax: +47 73 594466, http://www.mowahs.com
[2]Norwegian Institute for Air Research, NO2027 Kjeller, Norway. thomas@devbox.no

16.1 Introduction

Humans are said to be social animals. We seek contact with other humans, and we are used to collaborate to achieve our goals. By interacting in various ways, we can share information and exchange ideas and opinions. Both in professional and private settings, we use communication as a mean for collaboration. Collaboration is vital for productivity at workplaces. Many research projects have investigated how collaboration can be improved using computers.

Most of the earlier research studying human collaboration is based on the premise that interactions between colleagues happen in formal meetings where several persons have long interactions on pre-planned topics. Isaacs [Fin97] states that the interaction patterns between people at workplaces are indeed different. Most interactions are short, and tend to be spontaneous. Such interactions can occur at any location at the workplace when two or more co-workers meet by chance. The discussions will usually be informal, and tend to be continuations of prior conversations. Informal, spontaneous interaction at work places has important functions both socially and for collaboration. Incidental encounters between co-workers are a natural setting for collaboration where a lot of important information, ideas, inputs and updates on various issues can be exchanged.

With hardware and bandwidth becoming increasingly cheaper, new devices, platforms and applications are constantly being developed. According to Ken Dulaney of Gartner Group [Dul02], this can be expected to continue, and devices for computing and communication will become faster, smaller, more accessible, more affordable, and easier to use. This trend is already clearly visible through the fusion of PDAs and mobile phones.

The same level of innovation can be found within network technology. Wireless Local Area Networks (WLANs) have become common, and new standards such as Bluetooth and other implementations of Wireless Private Area Networks (WPAN) are being implemented in many new devices. Other areas of network technology are also subject to extensive research. Among these is ad-hoc networking, which enables devices to discover other devices autonomously and to communicate without the need for a central server or other existing infrastructure. This technology is very interesting for use in applications supporting spontaneous collaboration.

All these technologies and the new and improved ways of communication, provide new opportunities for collaboration. In this paper, we describe how mobile peer-to-peer technology can be used to promote spontaneous collaboration between co-workers. Our approach makes it easier to find the right person to collaborate with, and to share ideas and information utilising new technologies.

The rest of the paper is organised as follows: Section 16.2 describes two scenarios that were the motivation for building the tool. Section 16.3 describes the concepts of the tool. Section 16.4 presents the evaluation of potential technologies that could be used to

develop the tool. Section 16.5 describes how the tool was developed, tested and experiences from building the tool. Section 16.6 describes related work, and finally Section 16.7 concludes the paper.

16.2 Spontaneous Collaborative Scenarios

In this section, we will present two scenarios that promote spontaneous collaboration utilizing mobile peer-to-peer networks. The first scenario describes a university working environment with a large number of employees that are *knowledge workers*. The second scenario describes a large conference where a lot of unfamiliar people are gathered, and where interaction and collaboration are essential for gaining the maximum benefits of being present. The last part of this section presents the main characteristics and challenges of the two scenarios.

16.2.1 A Day at the Office at the University

John and all his colleagues just recently installed an application called ProMoCoTo (Proactive Mobile Collaboration Tool) on their mobile devices. The main objective of this application is to promote and support spontaneous collaboration. John is currently having some problems with Java RMI, and he adds this problem as keywords in a search criterion in his professional profile in ProMoCoTo. With ProMoCoTo installed, the mobile device now has the ability to match profiles with other users, and in this way initiate interaction between co-workers with matching queries and skills. The professional profile contains personal details and a detailed list of skills and current projects.

John walks through the corridor and his mobile device points out a person present with some experiences with Java RMI. He decides to talk to this person, a new PhD student, and the student suggests a solution to his problem. The use of ProMoCoTo has disclosed hidden knowledge at the university. The social barriers for initiating interaction with strangers have been reduced, since they now know that the other party can be a useful collaborative partner.

16.2.2 At the Conference

John travels to a large conference, with researchers from all over the world attending. The conference organisers have asked all attendants to install ProMoCoTo on their mobile devices in advance of the conference. When John arrives, he discovers that most of the others have done so. Since the conference has many parallel tracks, it is rather difficult for John to meet all the people working in his research areas. John configures his ProMoCoTo

with the keywords "*distributed architecture and context-aware applications*". Thanks to ProMoCoTo, John meets several persons on his first day of the conference with the same research interests. ProMoCoTo also enables John to exchange contact information with all his new research friends. A particular useful feature of ProMoCoTo is the ability to show the picture of the owner of the device with matching profiles. This makes it a lot easier to pick the correct person in a crowd.

16.2.3 Characteristics of the Scenarios

These scenarios are characterized by dispersed locations, where it is impossible to know experiences of all others. Further, both scenarios illustrate environments where people do not know each other and social barriers can prevent useful interaction. In such environments, a lot of "hidden" knowledge is unused because others do not know about it. Thus, a tool like ProMoCoTo will enable the workers to take the first step to utilise the hidden knowledge and to improve collaboration between colleagues.

16.2.4 Challenges Found in the Scenarios

The scenarios described above introduce some social and technological challenges that must be managed for the application to be useful.

The **social challenges** can be summarized in the three words: *Use, update* and *trust*. To efficiently share knowledge in an organization, most employees must *use the ProMoCoTo application*. If, e.g., large groups of the organization do not use the tool, their knowledge will still be hidden for others. In the same way, it is important for users to frequently *update their knowledge profiles* in order to reflect newly acquired knowledge. Lastly, it is important that the users *trust the application*, and that it is easy to choose what the users want to disclose to others.

There are also a number of **technical challenges** that must be managed to make the ProMoCoTo application useful in real environments. The application must be able to discover devices through the use of ad-hoc networks. Most wireless networks like infrared, WLAN, and BlueTooth have support for device discovery. Further, the application must be able to react to changes in the network to discover new peers or detect when peers are leaving. The application must also be able to perform search for matching peers to find the correct peers based on some specified search criteria. It is important that the information exchange between the mobile devices is very quick, since the ProMoCoTo application should be used in areas where people come and go. In some cases, this may cause that the information exchange will be interrupted and only parts of the information is exchanged. The application must be able to handle such situations. Lastly, it should be possible to run the application on various mobile devices, to increase its applicability and

user base.

16.3 The Concept of Pro-active Mobile Collaboration Tool (ProMoCoTo)

The initial idea of the ProMoCoTo was to implement a mobile pro-active tool. This means that the tool can operate on its own without user intervention. The pro-active part of the tool includes discovery of possible partners for collaboration and the creation of ad-hoc networks where spontaneous collaboration can be conducted.

Since the tool should be used in a mobile environment, it should be possible to run the application on mobile devices. These mobile devices must support wireless networking that can work in peer-to-peer mode.

The main purpose of the ProMoCoTo is to promote collaboration that can occur when co-workers meet by chance, and engage in spontaneous, informal interaction and also information exchange. The application holds a profile containing owner's name, employer, current projects and other useful information. It must also include a detailed list of the owner's professional skills/interest. The user should set own preferences, where the search criterion is a vital part.

When two or more co-workers meet, their mobile devices will discover each other and create an ad-hoc network. This is done without any user intervention. The two or more devices in the ad-hoc network are now treated as peers in the peer-to-peer network. The various entities of ProMoCoTo will immediately start to exchange messages. Figure 16.1 shows how the messages are exchanged. First a query will be sent, and this query is run against the local profile of each peer. If a match is found, the profile will be sent to the remote peer who sent the query. When a profile has been received, the user is notified, and becomes aware of the possibility for interaction and collaboration with nearby users.

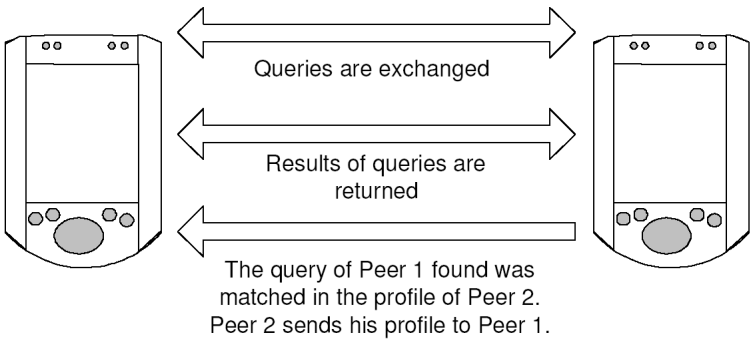

Figure 16.1: Message exchange in ProMoCoTo

16.4 Evaluating Technology

The initial criterion for selecting technology was that the tool should be possible to run on various mobile platforms. To enable portability, Java was chosen as the development and execution environment. In addition, the following requirements were used to determine suitability of existing peer-to-peer frameworks:

- Must be able to run on available resource-constrained devices.
- Must be able to support wireless ad-hoc communication.
- Must provide autonomous peers, and the ad-hoc network must be truly de-centralized. This means that the entire framework and tool must be able to run on the mobile device, without any need for external resources such as servers.

16.4.1 Evaluating Peer-to-Peer Frameworks

One of the main objectives of our peer-to-peer project was to evaluate existing peer-to-peer frameworks that can be used to implement peer-to-peer applications on mobile devices. The following candidates were evaluated:

- **JXTA** [BW02] is a framework for developing P2P applications, originally initiated by Sun Microsystems. JXTA is now an open-source project supported by Sun. JXTA provides a set of protocols and APIs for general-purpose, computer-to-computer communication. The motivation behind the JXTA project is to create a common framework for P2P systems. JXTA provides a fully decentralized environment that enables user applications to connect to each other in an ad-hoc fashion. This enables application developers to create robust P2P applications with a minimum of effort. JXTA is implemented in a wide variety of programming languages, but the reference implementation is in Java. The framework is platform and network independent.

- **JXME** [AP05] is JXTA for Java 2 Micro Edition (J2ME) and is a lightweight implementation of JXTA for mobile devices. It is specifically aimed at devices without sufficient computation and/or communication resources to participate in the network on their own. JXME is currently implemented in part on the CLDC/MIDP platform, and provides full JXTA functionality through the use of a relay host. The J2ME devices (mobile phones and PDAs) in the JXTA network are not connected directly to the JXTA network, and the relay hosts act on their behalf. All messages intended for the mobile devices are received and managed by the relay host. This means that the mobile devices do not communicate through direct peer-to-peer networks. There is also a JXME proxiless initiative, but no working implementation exists.

- **Proem** [Kor02] is a framework for developing and deploying P2P collaborative applications in a mobile ad-hoc networking environment. The framework is developed at the Wearable Computing Laboratory at the University of Oregon. The main objective of Proem is to provide a common framework for rapid development of applications for ad-hoc network environments. The framework is implemented in Java, and can be run on various wireless mobile devices. Proem is designed to be independent of underlying network transport protocols, and can be implemented on top of TCP/IP, HTTP, Bluetooth and others.

  The P2P applications that run on top of the Proem framework are called *Peerlets*. These are hosted in the Peerlet engine, which is responsible for instantiation, execution and termination of Peerlets. The Peerlet engine also controls the discovery of peers and users, the communication between Peerlets, and the Peerlet user interfaces. The Peerlet engine must run on every peer in the Proem network. Peerlets react to and communicate via events. The Peerlet engine fires events to Peerlets as a reaction to changes in its internal state or as a reaction to messages received by remote peers. Peerlets handle events asynchronously.

- **RockyRoad** [Alg04], has a set of features similar to JXTA. The main difference is that RockyRoad can be deployed on wireless devices with Palm OS and Pocket PC operating systems. However, this framework only provides limited support for mobile devices.

The peers involved in our prototype had to be truly autonomous. JXME is intended for mobile devices without sufficient computation and/or communication resources to participate in the network on their own. Also, previous experience with this framework concluded that the framework was not yet mature, and was missing important features in the API.

JXTA and Proem are similar in their overall approach, but their goals are different. Both JXTA and Proem are P2P frameworks as opposed to specific applications. JXTA has a broader scope and is more generic; it is intended as a common communication infrastructure for P2P applications, covering a large number of hardware platforms, networks, and applications. The Proem framework has a more narrow focus on ad-hoc networks and collaborative networks. Both frameworks are protocol-based; they define application-level communication protocols. There are also some fundamental differences between the two frameworks. The JXTA framework has its strength in support for developing P2P applications for semi-stable environments like the Internet. Proem, on the other hand, focuses on the person-to-person collaboration aspect of communication, and is designed for highly dynamic environments, such as those of ad-hoc networks with their constantly changing topology and available resources. Since JXTA is more general, this framework is more low-level than Proem, and requires more time to develop P2P applications.

All in all, we found Proem to be the most suitable P2P framework for our prototype. As the Proem framework was chosen for developing the prototype, most of the other choices of technology were given. The original version of the Proem framework runs on Java2 Standard Edition (J2SE), but a mobile version provided can run on PersonalJava or JDK 1.1.8. Our choice of framework also meant that we could only use mobile devices that have virtual machines that support the Java versions mentioned above. Our choice of framework made it impossible to use mobile phones and PDAs running the operating systems Symbian or Palm that only support Java2 Micro Edition (J2ME). However, for Compaq iPAQ we found the Jeode JVM, which is a PersonalJava distribution that could run Proem. Another option was to install Linux on a Compaq iPAQ or use a Linux-based PDA that could run a minimal version of the JDK 1.1.8 version.

16.5 The ProMoCoTo Prototype

This section describes how the prototype was developed, tested and some experiences we gained from building it.

16.5.1 Developing the Peer-to-Peer Application

The ProMoCoTo tool was implemented as a Peerlet in the Proem architecture as shown in Figure 16.2. The prototype was implemented in one package to avoid problems with the Jeode JVM's inability to handle complex package hierarchies. Our prototype uses the underlying layers in Proem to handle discovery of nearby devices, message passing, identifying users and groups, event handling, and debugging. The Proem framework also provides support for GUI, but this feature was not used because of its limited functionality.

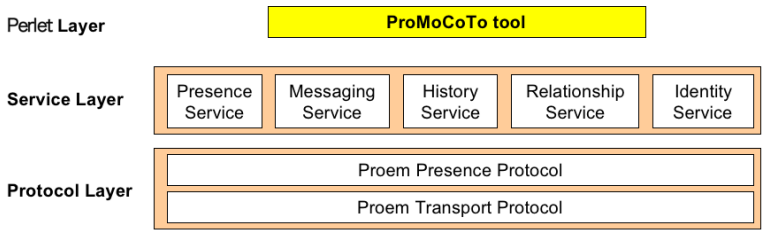

Figure 16.2: The ProMoCoTo tool shown in the Proem architecture

To handle the communication between peers, we implemented five message types:

- **PROMO_QUERY:** All peers send a query when they encounter another peer.
- **PROMO_CONFIRMATION:** The remote peer sends *CONFIRMATION* if the query returns a result.

- **PROMO_REJECT:** The remote peer sends a *REJECT* if the query does not return a result.
- **PROMO_REQUEST_PROFILE:** If a confirmation is received, the remote peer's profile will be requested.
- **PROMO_SEND_PROFILE:** This message contains the profile of the local peer.

Figure 16.3 shows the three screenshots of the user interface of the prototype. Screenshot **a** shows the GUI for displaying users that match the query of the current user. Screenshot **b** shows the profile details of one of the persons found that match the query. Screenshot **c** shows the preferences screen where the user can specify personal information (name, company, profession, skills etc), entering queries to search for matching users, and choosing between passive and active usage mode. In the *passive* mode, ProMoCoTo just collect information about other users that match the query without notifying the user. In *active* mode, ProMoCoTo will notify the user if there is another user nearby that matches his query.

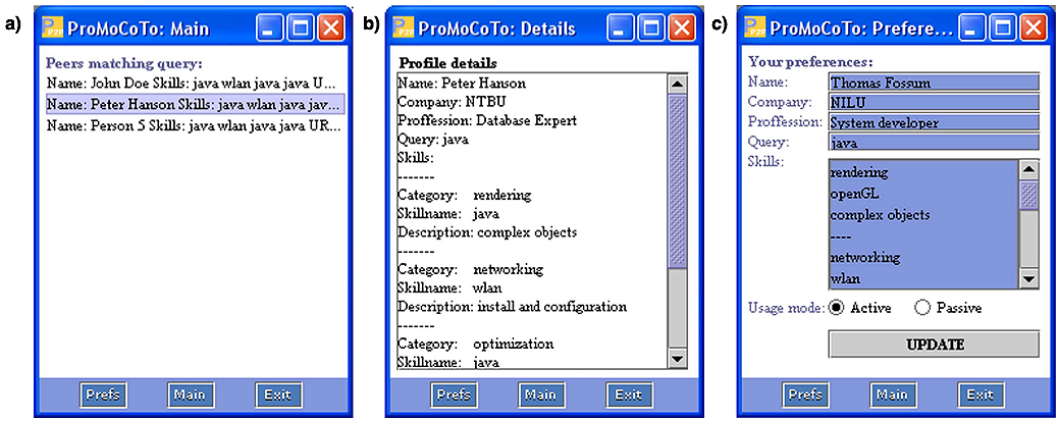

Figure 16.3: Screenshots from matching in ProMoCoTo

16.5.2 Testing the Peer-to-Peer Application

This section describes experiences from running the peer-to-peer application. The test was based on the university scenario described in Section 16.2.1.

The tests were performed in the hallways of Department of Computer and Information Science and the Norwegian University of Science and Technology (NTNU). The test persons were employees and students of the department since they were the target users for the application developed (see Section 16.2). This meant that all participants in the tests were computer experts and able to provide useful feedback. We helped the participants to enter their profile in the ProMoCoTo, and then asked them to solve a particular programming problem in a project. This programming problem was entered as a query in

the tool. To make this test interesting, we picked the programming problems to fit the skills of the participants. During the tests, the authors of this paper observed the involved participants and they were finally questioned about their experiences. Figure 16.4 shows two participants running the tool (left), and the tool running on an iPAQ (right). In this test we wanted to look at user and implementation issues as described in the paragraphs below.

Figure 16.4: Testing ProMoCoTo peer-to-peer application

The first aspect we wanted to investigate was if the user experiences of using such a tool would promote spontaneous collaboration. We found that the participants with matching queries and profile found each other when they passed in the hallway, and they started to talk about the programming problem when the tool had notified the match. Initially, it was rather cumbersome to match participants as there was no sound notification when matches where discovered. This caused the various participants to walk around staring at the PDA screen to see if there were any matching persons around. In a later version of the application, a beep was introduced when a match was found. The users found the possibility to view the profile of other users that matched the query useful. This information was used to initiate the conversation and made it easier to discuss the problem (as they knew each others background). The participants said that the prototype would be even more useful if it could provide other collaborative tasks like file exchange. The prototype was judged to be easy to use and understand because of the limited functionality. Thus, if the ProMoCoTo should be extended, the more advanced functionality should be hidden in separate screens to avoid confusion. The main problem we discovered was to identify the correct user when several users were located in the same room. A picture in the profile could have solved this problem. Another approach could be to use unique sounds to identify matching devices, but this could disturb other people.

The prototype was deployed on Compaq iPAQ PDAs, which at that time were among the most powerful PDAs. The Compaq iPAQs were equipped with WLAN cards used for

peer-to-peer communication. We found that this PDA was powerful enough for running the Java-based peer-to-peer application. The application was running fast enough to handle profile search of several nearby devices and providing the user with responsive user experience. One minor problem with the Pocket PC OS on the iPAQs was that it was hard to get the debug information since there was no easy way to redirect the standard output stream when running Java applications.

We tested the network performance to see if it was fast enough for two or more devices to exchange the necessary data. The time for data exchange between to devices can be decomposed into the time to perform the discovery of nearby devices, to create an ad-hoc network between the devices and to exchange the data. Within the office building of the department, the range of the wireless cards was about 20 meters. The network performance was acceptable for typical situations where people are passing each other in a normal walking pace to create an ad-hoc network, search for profile, and exchange information before user were out of reach. The average time for performing the necessary network communication between two iPAQs running ProMoCoTo was about 10 seconds. We also tested the number of users that could be handled without any problems. We found that the prototype worked well with five simultaneous users, but more could cause performance problems. We also found that we could improve the performance drastically by sending the profile directly instead of discovering nearby users first.

Before we got the prototype to run correctly, we had to solve some initial problems. The mobile device would not have an assigned IP address if the wireless card were not in contact with a network. This meant that if the PDA were powered on in an area without any wireless access points or other peers, the prototype would not work. With no IP address assigned at start-up, the application would throw an exception because it was unable to join a MultiCastGroup. In order to prevent this problem, IP-addresses were set to 10.0.0.1, 10.0.0.2, 10.0.0.3, etc., on the participating devices to get consistent networking performance. The wireless cards had to be inserted before application start-up.

We also tested portability of the prototype by deploying the tool on two different platforms, namely the Pocket PC OS using the Jeode JVM (running on the iPAQ), and the Windows XP OS using the JDK 1.1.8 JVM (on a Acer Travelmate laptop). Both versions were compiled with JDK 1.1.8, and the class-files were deployed on the iPAQs. Some modifications in the code were necessary to make the prototype run on the iPAQ. These modifications were all related to the fact that relative paths to files and images were not handled well on this platform. The conclusion was that our prototype could run on different mobile devices with minor changes of code.

16.5.3 Experiences

The actual implementation of the prototype was a time-consuming effort, due to an unfamiliar framework and initially unsolved problems. The poor documentation and lack of relevant sources for information on development using this framework made the early stages of the development a somewhat frustrating experience. But important lessons were learnt by trial-and-error. The choice of using an existing P2P framework for development instead of building everything from scratch proved in the end to be a sensible choice. The proof-of-concept could not have been realized without the use of the Proem P2P framework. Well into the development process, we realized that more investigation of the built-in profiles in Proem could have provided an alternative solution to what we were implementing. Still, the implementation proved its worth through the experiences gained and the test results it provided.

The Proem framework proved to be a sensible choice although the documentation was not too extensive. The framework offered all the functionality the prototype needed, and it ran on the iPAQs without too much hassle. It was in reality the only real choice, as other frameworks had a somewhat different scope and focus. The team behind Proem was very supportive, and was of great help during the development process.

16.6 Related Work

In [HFW99, Fil04], the concept of IPADs is presented. IPAD is used as a collective term for mobile, portable devices intended to improve the opportunities for face-to-face interaction in a group of people. IPADs aim to provide awareness solutions that combine the advantages of desktop-based awareness applications; constant, non-disruptive awareness, with the mobility provided by mobile devices. The IPAD is used as a contact facilitator, initiating contact and extending the human range of awareness. Several experimental prototypes have implemented this concept.

The Hummingbird [HWF98] has its name because it ”hums” whenever another hummingbird is nearby. The prototype implements the concept of an IPAD, and is meant to promote presence awareness and collaboration between people who are in the physical vicinity of each other. Constant awareness information is supplied to the users that are close to each other, without any reliance on an underlying infrastructure. When another device has been detected, a sound is generated, and the name of the owner of the other device is displayed on the device’s screen. Studies show that the Hummingbird did increase awareness between group members, and that it had the potential to complement other forms of communication such as phone and email. The Hummingbird prototype was found to be particularly useful when a group of people were in an unfamiliar location, with no other form of communication available.

The main objective of Proxy Lady [DLS02, LDS00] is to promote informal, opportunistic face-to-face communication. Proxy Lady is intended to be used by people who meet frequently, such as co-workers, and needs to share information when opportunities occur. The system supports these informal interactions by providing location awareness, as well as information items such as email messages and other files. These items are stored on the mobile device during synchronization with a desktop computer. The concept behind Proxy Lady assumes that these information items can be used as the basis for informal interaction. Users may want to discuss the information items they carry. This implies that the users of the system already know each other, and store information items on their mobile devices with later interaction in mind. As the Hummingbird, the Proxy Lady prototype provides presence awareness [DS00], but it adds the possibility to filter the candidates for interaction by what information items they carry.

AIDA [NFR02] is a prototype that combines the features of an Instant Messaging application with those of a Presence Awareness application. The prototype is deployed on a mobile device and is used to interact with devices that come within reach of the user as he moves within a pervasive computing environment. AIDA uses an infrastructure called DoMo, for opportunistic interaction with services available in a pervasive computing environment. When users join the ad-hoc network, agents representing services are registered in a list held on each mobile device. The user has access to a given set of services offered by other users of this network. The application also provides presence awareness through an Instant Messaging-part of the application.

The prototype Hocman [EJÖ02] is a mobile HTTP P2P application, which supports social interaction and enabled sharing of multimedia content over an ad-hoc network. The prototype has been deployed on mobile devices and tested in the very mobile environment, namely motor bikers riding their bikes. A rapid peer discovery algorithm was invented in order to cope with the very brief opportunities for peer discovery and the creation of ad-hoc networks.

The concept and prototype described in the paper was inspired by several of the concepts described above. The aspects of awareness and the initiation of interaction through the use of mobile devices are central. The developed prototype share many similarities with the concept of IPADs. It is intended for raising awareness of co-worker's skills, and to initiate interaction that can lead to useful collaboration. Also, the prototype is independent of existing network infrastructure, and only need the presence of other devices to be truly useful.

Our prototype shares the feature of providing information on nearby peers with implemented concepts such as the Hummingbird and the Hocman. This aspect is also implemented by the AIDA prototype. Both the Proxy Lady and the Hocman prototype add the concept of exchanging information items. This can however be done without additional software once the devices have established an ad-hoc network, but it could nevertheless be a useful extension of our prototype, providing increased convenience for the users.

The filtering of nearby users based on what information items they carry, used in both the Proxy Lady and Proem, closely resembles the concept that will be used in the future prototype to determine what peers that are interesting for collaboration.

Our prototype combines the strengths of several of the described prototypes. In addition to the presence awareness, our prototype uses the concept of filtering interesting remote devices, through matching search criteria to fire a notification and further exchange of information. The information on the owner of the nearby device is accessible on the local device through the exchange of profiles. The main technical difference with our approach is that our prototype can run on general-purpose mobile devices and that it uses Java.

16.7 Conclusion

In this paper we have presented some experiences from implementing a collaborative application using peer-to-peer technology in Java. We have found a promising application area for peer-to-peer technology that can promote spontaneous collaboration and improve knowledge sharing in big organizations. We have also found that peer-to-peer frameworks for Java are still immature, which makes it difficult to make stable and useful peer-to-peer application in Java. In addition, the lack of peer-to-peer frameworks in J2ME causes Java-based peer-to-peer applications to be limited to only the most powerful PDAs. Such applications have a much broader audience if they can be run on mobile phones. The authors of this paper are involved in a project developing a new framework for developing cooperative applications in J2ME called Peer2Me [WS05]. We have successfully tested an early version of this framework on mobile phones (Nokia 6600 and Sony Ericsson P900) using Bluetooth. The framework provides opportunities to create a palette of collaborative applications for mobile devices utilizing peer-to-peer communication.

Acknowledgement

This paper is a result of work in the project MOWAHS [MOW04] sponsored by the Norwegian Research Council's IKT2010 program.

CHAPTER 17

P10 – The Nidaros Framework for Development of Location-aware Applications

Alf Inge Wang, Carl-Fredrik Sørensen[1] , Steinar Brede[2] , Hege Servold[3] , and Sigurd Gimre[4]

Abstract

This paper presents the Nidaros framework for developing location-aware applications that provide location dependent functionality based on the current location of the user. The framework can be used to develop location-dependent advertisement, city guides, guides for tourist attractions, etc. The framework consists of three main components: *A runtime system* that manages user locations and the interaction with the user clients; *a creator tool* that is used to map information and multimedia content to locations; and *a logging tool* that is used to log the movement of users to monitor the interest for certain locations. The paper also describes an implementation of a location-aware tour guide for the Nidaros Cathedral in Trondheim that can run on different mobile devices. Further, the paper describes experiences from installing, configuring, and running a location-aware

[1] Dept. of Computer and Information Science, Norwegian University of Science and Technology, Sem Sælands vei 7-9, NO-7491 Trondheim, Norway, Phone: +47 735 94485/90731, email: alfw/carlfrs@idi.ntnu.no

[2]Telenor R & D, NO-7004 Trondheim, Norway, email: steinar.brede@telenor.com

[3]Bekk Consulting AS, NO-0102 Oslo, Norway, email: hege.servold@bekk.no

[4]CIBER Norge AS, NO-0103 Oslo, Norway, email: sigurd.gimre@ciber.no

tour guide in a real environment. A demonstration of the tour guide was tested on PDAs and mobile phones. The event received a lot of publicity.
Keywords: *Context/location, case studies and experience, applications, tour guide*.

17.1 Introduction

In 2005, it is estimated that there will be more than 1.5 billion wireless subscribers worldwide. Although mobile computing gives challenges to the application developer like handling wireless networks, heterogeneity, limited screen size, input device CPU, memory and battery [Sat96], it gives possibilities to develop new types of applications. One such type is location-aware applications. Location-aware applications are useful for several reasons: Firstly, the limited screen size of mobile devices can be better utilized by providing user interfaces that are related to the context of the user. By using the location, the parts that are not relevant to the current location can be left out, making better use of the limited size. Secondly, the user experience can be improved by providing the user with information and interaction that are relevant to the current context. Here the context is necessarily not only location, but can also be time, weather, temperature, altitude etc. Thirdly, a system can collect context information from users to further improve the location-aware system. For instance in a tour guide system for an art gallery, the system can log which paintings the visitors spend most time by. This information can be used to publish additional information about the most popular paintings in the location-aware guide system.

Several location-aware systems for tour guiding have been developed like the Lancaster GUIDE [CDMF00], CyberGuide [AAH+97] and MUSE [GCP03], but most of these systems are tailored for a specific location-aware application. In 2003, the Norwegian University of Science and Technology (NTNU) started together with Telenor, the largest telecom company in Norway, to develop a general framework for creating, running, and analysing mobile location-aware systems. The motivation for this work was to enable rapid development of location-aware systems that can provide the user with information or multimedia content dependent on the user location. Another important aspect of the framework was to enable support for different mobile clients with different characteristics from the same system. From similar projects, we have found that location-aware systems use various client devices from rather big portable PCs down to small PDAs [SWØH03]. Also we noticed that some location-aware systems use customized hardware to get the required characteristics. In addition, the evolution of mobile devices makes it necessary to be able to adapt to future devices with new and useful capabilities. From talking with people managing a PDA-based tour guide (not location-aware) at the Nidaros Cathedral, we understood that theft was a serious challenge. Letting people use their own mobile equipment for such services was found to be very interesting. Based on this observation, we found it important to be able to support many different mobile devices including

mobile phones not previously been used in such systems.

Another shortcoming for many of the existing systems is that they are tailored to support only one type of positioning technology like GPS, GSM, Bluetooth, IR or WLAN positioning. We used in the Nidaros framework a location server to fetch the user positions from various sources and to send this information back to the system when needed. The location server examines position technologies available to return the most accurate position. The use of a location server also enables change of coordinate system for the location-aware application or can provide different coordinate systems for various parts of the system.

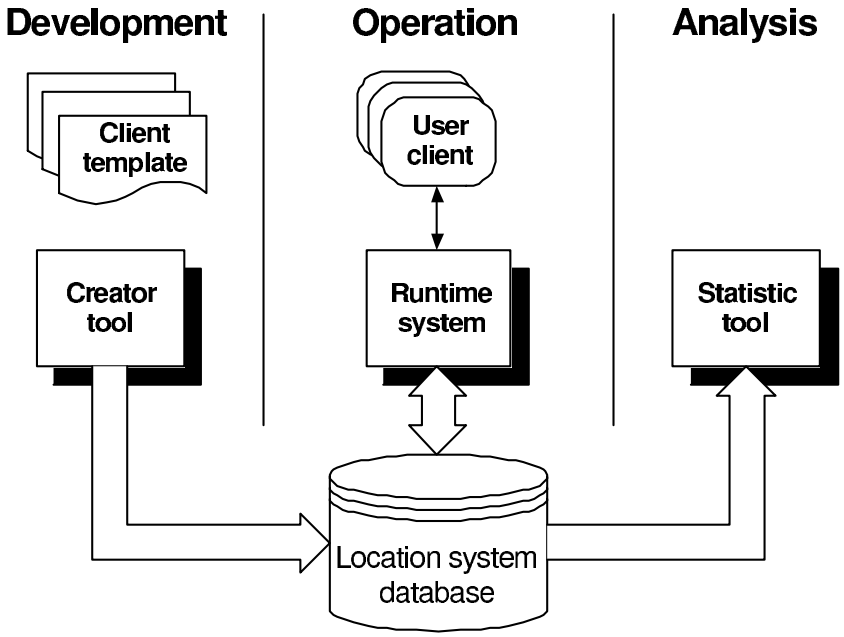

Figure 17.1: The Nidaros framework for development of location-aware applications

Figure 17.1 shows an overview of the Nidaros framework for development of location-aware applications. The framework covers three lifecycle phases of a location-aware system: Development, operation and analysis. In the *development phase*, the Nidaros framework provides a creator tool to map information and multimedia content to locations used by location-aware applications. The output generated from the creator tool is stored in the location system database. In addition, we provide client templates that are used as a basis to implement client applications that can interact with the location system database. In the *operation phase*, the framework provides a runtime system that manages every aspect of the location-aware application. The runtime system interacts with multiple clients (of various types) and the location system database. In the *analysis phase*, the framework provides a statistic tool used to analyse the use of the location-aware application. The statistics are computed based on data stored in the location system database.

The rest of the paper is organized as follows: Section 17.2 describes the Nidaros frame-

work in more detail including design considerations, Section 17.3 describes how we implemented a tour guide for the Nidaros Cathedral using the framework, Section 17.4 describes our experiences from creating and using the framework, Section 17.5 relate our framework to similar frameworks, and finally Section 17.6 concludes the paper.

17.2 The Framework

This section presents the Nidaros framework, divided into the three phases (development, operation and analysis), a location server developed by Telenor R & D, and a location system database.

17.2.1 The Development Phase

The development phase of the framework is supported by three components: A creator tool, client templates, and the XML interface between client and server.

The Creator Tool

The creator tool is a system for managing information and multimedia content related to locations that are to be part of a location-aware application. The tool provides a simple user interface that facilitates describing areas hierarchically into maps, zones and objects as a tree structure. A map represents the whole area of the location-aware application. A map can be divided into one or more zones that represent some specific areas in the map. Within a zone, you can add several objects. These data objects typically represent physical objects and may contain information, audio-clips, video-clips and similar. The maps, zones and objects are mapped to a local Cartesian coordinate where the x- and the y-axis are represented with a 32-bit integer ($2^{32} = 4,294,967,296$). The coordinate system can be mapped to various geographical positioning systems and makes the framework independent of the actual coordinate systems. In the local coordinate system, a map is represented as a rectangle ($X_{min}, X_{max}, Y_{min}$, and Y_{max}). A zone is represented as a polygon of four coordinates: $(X_1, Y_1), (X_2, Y_2), (X_3, Y_3)$, and (X_4, Y_4). An object is represented by a specific coordinate, (X_{obj}, Y_{obj}), but also with a hotspot area represented by a polygon similar to a zone. The hotspot area is used to determine if a user is close to an object to trigger some location-aware event. Figure 17.2 shows a screenshot from creating an object using the creator tool (the text is in Norwegian).

In the tour guide application we developed for the Nidaros Cathedral (described in Section 17.3), the map represented the whole cathedral, the zones represented the main parts of the cathedral, and the objects represented physical objects like alters, the pipe organ,

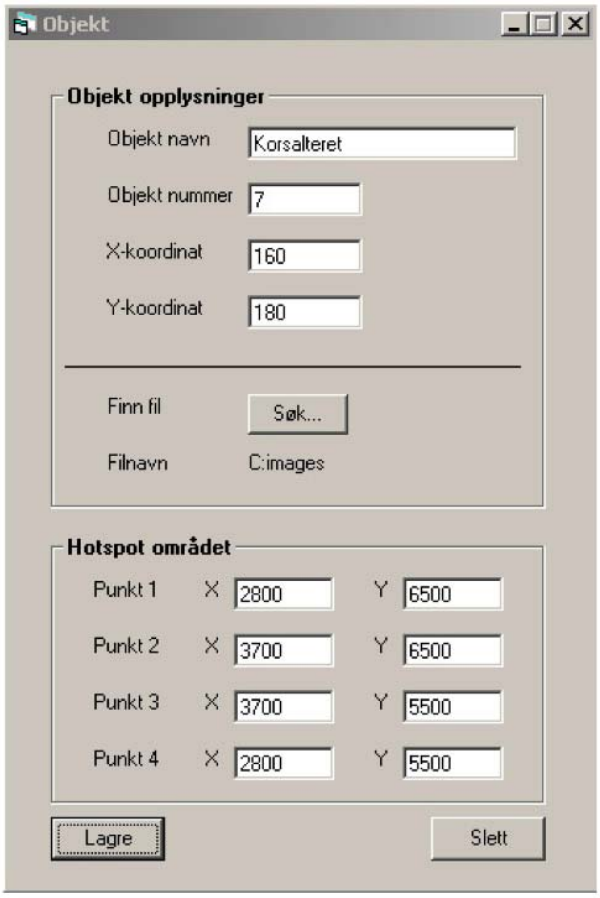

Figure 17.2: A screenshot from the creator tool

paintings, etc.

The creator tool offers a user interface for how various devices like laptops, PDAs and mobile phones are mapped to the local coordinate system. The mapping identifies the type of device by name, the size of the device, an offset coordinate (X_{offset}, Y_{offset}), a rotation coordinate $(X_{rotation}, Y_{rotation})$ and a rotation angle R_{angle}. These data are used to compute the transformation from real world coordinates to the local coordinate system.

The creator tool can be used to graphically visualize the results of using the tool. The visualization shows the map you have created with named zones and objects. The zones and the objects are shown in different colours. Hotspot areas are shown for the objects.

The Client Templates

The framework provides a template for reducing time to create clients for location-aware applications. The template is intended for clients that run Macromedia Flash applications.

Flash makes it possible to make advanced and dynamic interfaces, and can run on various operating systems like MS Windows, Mac OS, Linux, Unix and Pocket PC. The Flash player is also capable of audio and video playback.

The template offers basic functionality like server communication using XML, graphical highlighting of zones and objects, management of hotspots (event-triggered actions), and initialization of multimedia playback. In addition to the template, the user interfaces and additional functionality must be added and integrated.

The XML Interfaces

The Nidaros framework provides an open XML interface to the runtime system that makes it possible to create clients for different devices using different technologies like Flash, Java 2 Micro Edition, HTML, .NET compact framework, etc. The only requirement is that the client is capable of managing XML communication and adheres to the specified XML interface. The client application can send a request to the runtime system in several different ways, but it has to follow a predefined XML syntax. The response will likewise follow a predefined syntax. If the client application requires downloading of multimedia content, this is done by an HTTP-request to a file server (see Section 17.2.2).

A summary of the document type definition (DTD) for a request is shown below (the definition of data types of the elements have been left out).

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT locationRequest (mac, get+, ,mapping*)>
<!ELEMENT get (#PCDATA | name | startX | startY |
                stepX | stepY | stopX | stopY)
...
```

The root element in every XML request is a *locationRequest*. This root element can contain several different elements depending on what kind of information that is wanted. Every *locationRequest* must contain an element called *mac* that holds the mac address of the client device. The runtime server needs the mac to identify and find the position of the user. The *get* element is optional and is used when the client application wants to receive some specific information from the runtime server. We have defined six values in the *get* element: **Position** will get the current position of the user, **simulatedPosition** will get a simulated position of the user (useful for demonstration and testing), **friendsPosition** will return positions of other users registered as friends, **dynamicInfo** will return objects that the user is within the hotspot area of, **tracks** will return some predefined routes that the user might want to follow, and **messages** is used for sending and retrieving messages between users.

The *mapping* element is used when the $\langle get \rangle$position$\langle /get \rangle$ and the $\langle get \rangle$friendsPosition$\langle /get \rangle$ element are involved. The mapping element specifies a name (profile) of how real world

coordinates are to be mapped into the local coordinate system. If this element is omitted, a default mapping value is used. The creator tool sets this default value when configuring the runtime server. Here is an example of a request from a client to the runtime server:

```
<locationRequest>
   <mac>00:0B:FD:C6:5C:AB</mac>
   <get>position</get>
   <mapping>PDA</mapping>
</locationRequest>
```

Similar to the *get* element, the *send* element is also optional. Currently, *send* is only used for sending messages between client users. However, the *send* element can be extended to provide support for new services when needed. A summary of the DTD of the *send* element is shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT locationRequest(mac,send+)>
<!ELEMENT send (message)>
<!ELEMENT message (note+)>
<!ELEMENT note (to, from, header, body)>
<!ATTLIST note id CDATA #REQUIRED>
...
```

When a request is sent to the server, the client will get a response from the server depending on the request type. Below is a summary of the DTD of a response (all definition of data types have been left out):

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT locationResponse(position, dynamicInfo,
           friendPosition, tracks, message, error)>
<!ELEMENT position (user, map, zone, hotSpotObject)>
<!ELEMENT user (mac?, mapping?, nick?, x, y, map?)>
<!ELEMENT map (name, number, floor, zone*)>
<!ELEMENT zone (name, number, floor?, located?,
               object*)>
<!ELEMENT hotSpotObject (name, number, x, y)>
<!ELEMENT dynamicInfo (map+)>
<!ELEMENT object (name, number, x, y, hs, url)>
<!ELEMENT friendPosition (user+)>
<!ELEMENT tracks (track)>
<!ELEMENT track (name, number, coord)>
<!ELEMENT coord (x | y)+>
<!ELEMENT message (note+)>
<!ELEMENT note (to, from, header, body)>
<!ATTLIST note id CDATA #REQUIRED>
<!ELEMENT error (type, text)>
...
```

The root element in every XML response is the *locationResponse*. The root element can contain several different elements dependent on what kind of response that is required.

If the requested information for some reason is not available, the server will give these elements the value "noinfo" or "-1" depending on the data type of the element (string or number). Every *get* element in the client request receives a separate element in the response.

The *position* element is the response of the $\langle get \rangle$position$\langle /get \rangle$ request. The *position* element contains information about the user and the position of the user described by the current map, current zone and current hotspot objects. It is then up to the client application to use this information e.g. to display the user position in a map with highlighted zone and objects.

The *friendPosition* element is the response of the $\langle get \rangle$friendsPosition$\langle /get \rangle$ request. The *friendPosition* element contains information about all persons registered as friends of the user and their positions.

The *dynamicInfo* element is the response of the $\langle get \rangle$dynamicInfo$\langle /get \rangle$ request. The *dynamicInfo* element will give information depending on specified preferences. The *dynamicInfo* element contains information about all available maps, zones and objects. This information can be used to show dynamic menus. It is possible by using this service to make changes to the location of objects, zones and maps both in the attraction area and in the database without having to make changes in the client application.

The *tracks* element is the response of the $\langle get \rangle$tracks$\langle /get \rangle$ request. This service can be used to suggest a specific route for the user to follow. The response is a list of coordinates describing the track. The client decides what to do with this information and how to present the track to the user. Such tracks can for instance be used to guide users in a museum to all objects that are related to a specific person or historical time period.

The *message* element is the response to the $\langle get \rangle$message$\langle /get \rangle$ request. The *message* element contains all unread messages to the addressed user.

The *error element* is used by the runtime server to notify the client when something goes wrong, e.g., a request with illegal syntax.

Below is an example of a response from the server:

```
<locationResponse>
  <position>
     <user>
        <mac>00:0B:FD:C6:5C:AB</mac>
        <mapping>PDA</mapping>
        <x>100</x>
        <y>130</y>
     </user>
     <map>
        <name>Nidaros Cathedral</name>
        <number>1</number>
        <floor>1</floor>
     </map>
     <zone>
        <name>Entrance</name>
        <number>3</number>
     </zone>
     <hotSpotObject>
        <name>Statue St. Olav</name>
        <number>1</number>
        <x>100</x>
        <y>120</y>
     </hotSpotObject>
  </position>
</locationReponse>
```

17.2.2 The Operation Phase

The main components used in the operation phase are the user clients, the runtime system and the location system database. The user clients are developed using the client template and the XML-interface described in Section 17.2.1. The runtime system is the heart of the Nidaros framework that brings location-aware applications alive. The runtime system manages the information in the database including maps, zones, objects, users, and etc. The main task of the runtime server is to feed the clients with correct information according to the client position.

The architecture of the runtime server was developed according to the 4+1 view model [Kru95] and the IEEE 1471 recommended practice for architectural description [IEE00]. We will in this paper only focus on description of the physical and the logical view in this paper. The architecture of the runtime system was especially constructed to enable extendibility and maintainability of the system. This is reflected in both the physical and logical view described below.

Figure 17.3 shows the physical view of the runtime system. The runtime system supports several types of clients and identifies three client types we have implemented. The figure shows that wireless LAN is used between the server and the clients, but also other types of wireless networks have been used. Currently, our mobile phone client uses GPRS for communication between the client and the server.

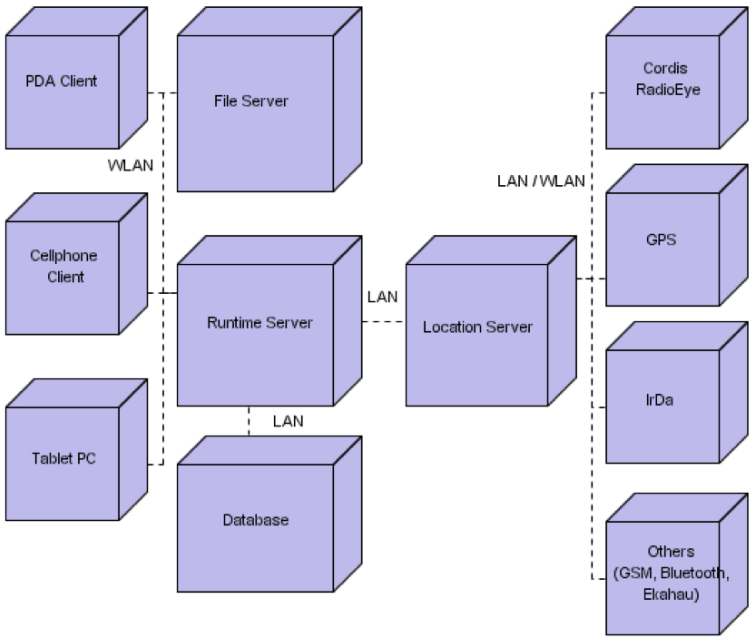

Figure 17.3: The Physical view of the runtime system

The runtime system itself consists of four main components:

- **File server:** The file server stores media files accessible for mobile clients. A file server is used because mobile devices are not likely to store all media files locally because of the limited memory. For some location-aware applications, it is not necessary to use the file server, but for tour guides that feature audio or video playback in multiple languages that a file server is needed. However, the mobile clients should have the most used multimedia elements stored locally on the device for faster access and to avoid delay when initialising multimedia playback. Some clients also use local caching on the device to store the most used multimedia elements on the device.
- **Runtime server:** The runtime server is responsible for handling requests from the clients and providing the services requested. A more detailed description of this component is given later in this section.
- **Location system database:** The location system database stores all information used by the clients and the server. This database is also used by the creator tool when creating location-aware applications, and by the statistical tool for analysis of user patterns.
- **Location server:** The location server gets the current positions of the clients through various interfaces to different position technologies like WLAN positioning, GPS, IrDA, Bluetooth etc. A more detailed description can be found in Section 17.2.4.

We have not made extensive tests for how well our system scales for many users. However, we believe that the wireless network between the clients and the server will be the main bottleneck of the system. If the GSM network is used, only audio streaming can be supported. If Wireless LANs like IEEE 802,11b or even better IEEE 802,11g are used, video streaming can be supported. The number of simultaneous users to be served at the same time depends on how well the physical network is implemented. Another possible bottleneck can be the file server. However, such servers can be duplicated to achieve better performance.

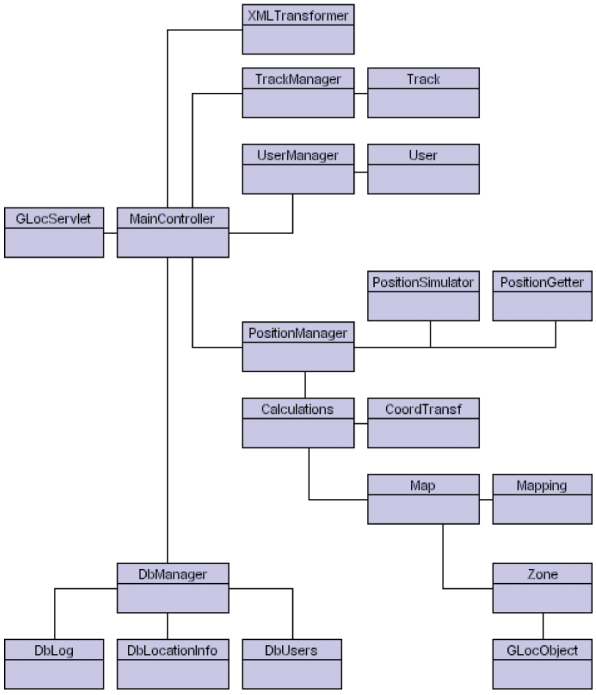

Figure 17.4: The Logical view of the Runtime system

Figure 17.4 shows the logical view of the runtime server architecture. The logical view identifies 20 main classes. The system is communicating with client applications through the *GLocServlet* class. Data is exchanged as XML, and the *XMLTransformer* class interprets and transforms the information sent between clients and the server. For the runtime server, we decided to use an architecture based on a centralized control model. This means that all information flow through the *MainController* class. By using centralized control, it is easy to analyse control flows and get the correct responses to the given client requests. It also makes it easy to substitute the servlet class with another class for handling the client communication and to add new interfaces to the system as needed. The *UserManager* class is responsible for maintaining information about the users. This task includes storing the user's last position and deciding whether a person is allowed to communicate with another person (defined as friend). The *PositionManager* class is responsible for returning the user position, adjusted to the type of mobile device used.

The *Calculations* and *CoordTransf* are classes that manage transformation from world coordinates to map local coordinates. The *Zone*, *Object*, *Map* and *Mapping* classes are used to store information about the location. The *TrackManager* class is responsible for keeping information about the available predefined tracks. Each track has a unique name, so the client application can either request all tracks or one particular track. The runtime system has two different operation modes: Runtime and testing. In runtime mode, the system will use the *PositionGetter* class and return real position information of the clients using the location server. In testing mode, the system will use the *PositionSimulator* class and return simulated position information of the client. This means that the system can demonstrate its functionality without use of any positioning technology. The *DbManager* class is responsible for all communication with the database. This class uses the *DbLog*, *DbLocationinfo* and *DbUsers* classes to perform tasks when requested.

17.2.3 The Analysis Phase

The statistical tool is useful for analysing the use of location-aware applications. The tool uses data logged by the runtime system to look at user behaviour and what objects that are most popular. Figure 17.5 shows the logical view of the statistic tool.

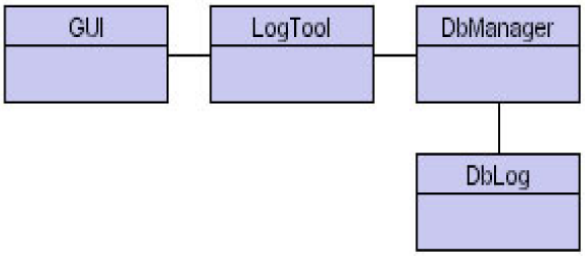

Figure 17.5: The logical view of the statistic tool

The logical view consists of four classes: GUI, LogTool, DbManager and DbLog. The GUI class presents the different services available and interacts with the user. The LogTool class is responsible for calculation and manipulation of the data stored in the database. The DbManager and DbLog classes handle database issues.

The users of the system must agree that they will permit logging of their use.

17.2.4 The Location Server

The location server, developed by Telenor R & D, is a framework for uniform, geometric-based management of a wide variety of location sensor technologies. The goals of this framework are to have one server that can get positions of mobile devices through multiple

location sensing technologies, and to provide this position information to other systems. By using the location server in the Nidaros framework, we do not need to tailor our system to use a specific positioning technology. We can also use different positioning technologies within the same application.

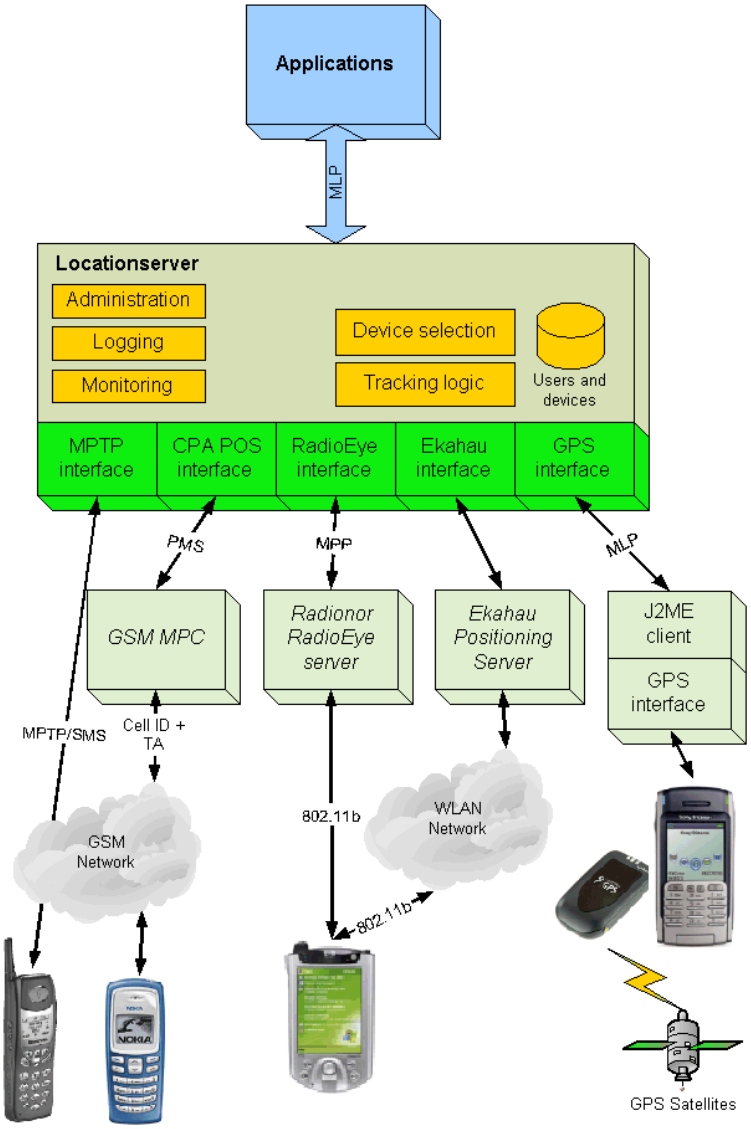

Figure 17.6: The architecture of the location server

The server includes a middleware protocol specification and a specification of quality-of-service parameters. Further, the server has support for event-driven position reporting (i.e. for change of position) and support for methods for merging and position enhancements. The architecture of the location server is layered and shields the application from the details of collecting and merging location information from a variety of sources. The

devices that are positioned by the server are identified by the MAC-address of the device. The location server also manages authorisation for accessing the position of a device.

One advantage from using the location server is that this server handles the complexity of merging locations that might include partly overlaps of positioning systems and seamless transitions between different position systems. This is especially useful for location-aware applications that cover both indoor and outdoor areas.

The location server currently support positioning using GPS, GSM, and wireless LAN using the RadioNor RadioEye server, but can be extended to also support others. Figure 17.6 shows an overview of the location server architecture.

17.2.5 The Location System Database

The location system database stores location data, log data, and user data.

The *location* data is represented in five tables describing maps, zones, objects, preferences, and mapping. The motivation for these tables is that one location can have several maps covering different territories, each map can cover several zones, each zone can contain several objects, and each object can be connected to several preferences. The preferences of an object are used to provide the dynamic menu service. The user can state the preferences in the attraction, and only objects matching his preferences will be displayed in the client menu. In addition, there must exist a table with mapping information to transform locations from world coordinates to coordinates adjusted to fit the client map.

The *user* data is represented in three tables describing users, user groups and user preferences. The user table contains an MAC-id of the user device and other information. The *user group* table is used to group users that are friends to allow services like tracking friends and messaging. The *user preferences* table stores information about the user's main interests.

The *log* data is represented in two tables. One table is used for storing user movements and one table for storing what kind of objects the user is interested in. The statistical tool uses the log data.

17.3 Implementing a Tour Guide Using the Framework

We created a location-aware tour guide for the Nidaros Cathedral in Trondheim to test the Nidaros framework in a real setting. The cathedral had an existing PDA-based tour guide, called Nidaros Pocket Guide (NPG) that was not location-aware. The NPG was implemented in Flash MX by the multimedia company *Klipp og Lim Media*.

To show the flexibility of the Nidaros framework, we decided to implement support for two different types of clients: A PDA and a mobile phone. The rest of this section briefly describes how these two clients were implemented and the testing environment and configuration of our application.

17.3.1 The PDA Client

We decided to develop our location-aware PDA client in Flash MX. This made it possible to reuse code from the NPG and the client template. Our PDA application provided functionality for selecting language, enable trace of own movement, show friends on a map, show zones and objects on a map (with highlighting of current zone and objects), displaying text about objects, and playback of audio or video about an object. The PDA used to run the client was an iPaq with Windows CE and the wireless LAN IEEE 802,11b network integrated. This made it possible to get the position of the device using WLAN-positioning. The location-awareness was presented in the client application in two ways. The first way was to show a map with the position of the user, position of possible friends, highlighting of current zone and nearby objects. The second way was optional for the user, and made the application able to initiate display of objects (text, audio or video) when the user entered the hotspot area of an object. To the left in Figure 17.7, a PDA client is shown.

Figure 17.7: The system running on iPaq and SonyEricsson P900

17.3.2 The Mobile Phone Client

A mobile phone client could either be implemented in J2ME or using HTML and the web-browser installed in the phone. We decided to do the latter by implementing an additional servlet component that communicates with the runtime server and produces HTML for the mobile phone. We used WAP push to send the appropriate web pages when the user was within a specific area to enable the client to react on the user location. This made it possible to, e.g., get the phone to initiate playback of a video when the user was close to a specific alter. We used a SonyEricsson P900 in the test because of its support for WAP push and the built-in multimedia player. To the right in Figure 17.7, the mobile phone client is running on a P900. The most common way to position mobile phones is to use GSM positioning. This method is too course-grained to be used for positioning inside a cathedral. To solve this problem, we came up with the idea of letting the user wear WLAN tags that could be positioned using WLAN positioning. WLAN tags are produced by RadioNor Communications, and are very small WLAN radios that transmit a MAC-ID. Such devices are small and do not bother the user more than the ordinary tags they need anyway to enter the cathedral to show that they have paid the entry fee. A picture of a WLAN-tag is shown in figure 17.8. The size of the tag is about 4cm wide and 3cm high. For the mobile phone client, it was necessary to register the MAC-ID of the WLAN tag and the mobile phone number to link the phone to the tag.

Figure 17.8: A photo of a WLAN tag

17.3.3 The Position Technology Used

We used the Cordis RadioEye [Com04] WLAN positioning produced by RadioNor Communications to position the client devices used in our location-aware application. The RadioEye is a small box with advanced antenna technology and a Linux-server that can determine the physical coordinates of every WLAN-terminal that are active within its coverage area. The sensors decode the MAC addresses of the network devices and determine their geographical position with a typical accuracy of 1-2 meters. A RadioEye covers a radios about 60 degrees from the centre and can e.g. be placed in the ceiling of a building (see Figure 17.9).

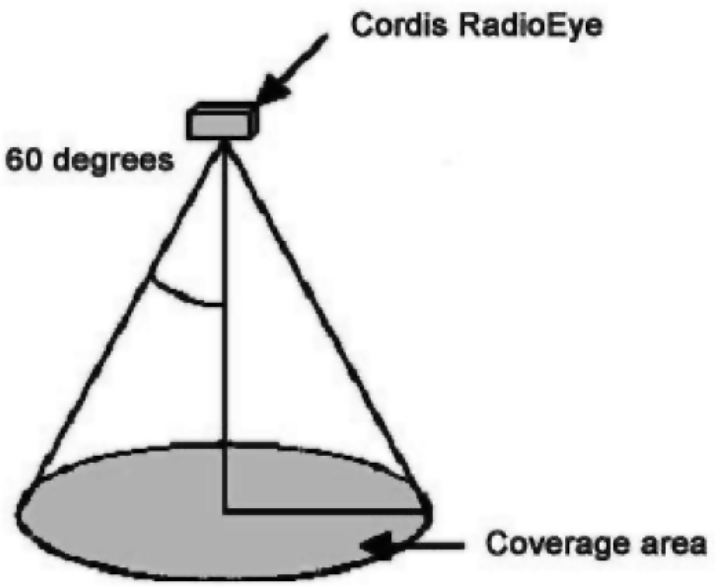

Figure 17.9: Coverage area of a RadioEye

17.3.4 Setting up and Running the Location-aware Application

The demonstration of the location-aware tour guide for the Nidaros Cathedral was performed May 14th 2004. Before we could start to run the demonstration, we had to install the infrastructure needed for running the system. Four RadioEyes were installed at the gallery of the cathedral to cover four different zones of the building. After the RadioEyes were in place, the coordinates from the RadioEye server had to be aligned with the coordinates used by the system. This was done by taking measurements from different locations in the cathedral. These measurements were made in a pre-test before the real demonstration to make sure that the system would work as expected.

Approximately 20 people were present at the demonstration representing various technology-oriented companies as well as the media (television, magazines and newspapers). The demonstration of the system was received very well and especially the mobile phone application attracted much attention. As far as we know, there are no similar location-aware applications running on mobile phones. A representative from the cathedral was very exited with the project, and could imagine that a tour guide on mobile phones would be a great substitute for the PDA tour guide used today. The demonstration was shown in two of the major television channels in Norway (NRK and TV-Norge), the largest national newspaper VG and several magazines [5]. Figure 17.10 shows a picture from the demonstration.

[5]Media coverage of the event is available on http://media.mowahs.com/ including video clips from the demo.

Figure 17.10: Demonstration of a location-aware tour guide

17.4 Experiences

This section describes experiences we gained from developing a location-aware application using the Nidaros framework. We have found the framework very useful for several reasons. Firstly, the server side of the system can be used directly without any modifications. The only thing missing is the information to be used in the location-aware application. This information can easily be added by using the creator tool. In addition, the creator tool can be used to make changes to the location information before and during the operation phase.

By using the available client template, the time to implement a client is rather short. Currently, client templates are only available for HTML and Flash. It would have been useful to provide templates for Java 2 Micro Edition and .Net Compact Framework. It does not require much work to write a client from scratch using the defined XML-interface. XML parsing might cause a challenge for a client implemented in J2ME because of the memory limitation in J2ME. However, the XML messages used in the Nidaros framework are relatively small and simple, and do not contain several levels. A possible extension of the Nidaros framework could be to make a client generator to ease the implementation of new clients for all client technologies.

The simulated movement of users was found to be a very useful feature of the runtime system and was invaluable for testing client applications. It takes several hours to set up a real environment with real sensors, and it would be time consuming to engage real users to debug the application.

The database used in the framework was found general enough for the tour guide application. However, there are limitations on what type of information that can be stored.

This means that the database scheme might have to be extended to fit any location-aware application.

We introduced a file server that could feed the clients with multimedia contents because the limited storage available on client devices. In an ideal system, all the media files should be stored locally on the device for quick and responsive presentations. This was impossible for the tour guide for the Nidaros Cathedral if more than one language should be supported on the same device. Most of the multimedia files were audio files, but it was not storage space enough for more than one language. By introducing more videos, this would be a bigger problem. We found from the demonstration in the cathedral that the user had to wait from 5 to 7 seconds before the audio or video was played. We stored the most used media files on the device to avoid such long waiting times. An extension of the Nidaros framework could be to have smarter media file communication management. This means, e.g., that the mobile client can start to pre-fetch files that are likely to be played in the near future based on the location and movement of the user. Such an extension would include a more advanced file server as well as media communication management on the mobile client. A problem with this approach could be a high demand on the wireless network for transporting multimedia files that might not be used. Another way this could be optimized is to use the statistical tool to analyse which media files are played most and in what locations.

Our framework uses a client-server solution over a wireless network. This means that if the wireless network is down or have some problems, the clients will not function properly. The clients should therefore have off-line mode that enables the user to use the system independent of the availability of the wireless network.

The mobile phone client got tremendous attention when we demonstrated the location-aware tour guide in the Nidaros Cathedral. The main reason was to be demonstrated such advanced applications running on devices that many own. The current solution involves a WLAN-tag to make the system work. For a commercial tour guide for mobile phones, the WLAN-tag could be available in the entrance of a sight by paying the entrance fee. Sending an SMS that includes the ID of the tag to the tour guide server could initialize the setup of the mobile location-aware tour guide. The combination of using a WLAN-tag together with a mobile phone gives other new exiting opportunities for location-aware applications that can be used both indoors and outdoors. The WLAN-positioning technologies like the RadioEye from RadioNor Communications can be installed in all public building like airports, hospitals, shopping centres etc.

The use of a location server makes it easy to adapt to new positioning technologies when they are available. The only required change of the system is to implement a new interface for the new position technology in the location server.

The choice of using XML for data exchange between client and server has many benefits. The main benefit is the extensibility of the interface and provision of an open interface

to other systems. The main disadvantage of using XML is the overhead sending messages between client and server. For a system with many users, this can cause scalability problems because of the limited bandwidth in wireless networks. A problem we experienced running the location-aware tour guide, was the high demand on CPU and memory on the mobile clients for parsing the XML-data. Generally, the clients spent more time on parsing XML data than they used to send requests and receive responses. A high demand on the CPU will also make the battery run out faster. We foresee that this problem will not be so dominant for future mobile devices with improved performance and battery technology.

17.5 Related Work

This section describes work on similar frameworks from location-aware and context-aware applications.

The NEXUS Project [VS00] has developed a generic infrastructure that can be used to implement all kinds of context-aware applications both for indoor and outdoor services. The NEXUS clients access the server via a standardized user interface running on the mobile device carried by the user. The interface has to be adjusted to the different kinds of clients, especially concerning the different level of computing power, different amounts of memory and different size of displays. A NEXUS station can manage sensor systems that measure both global indicators (like temperature) and object related information (like location). The NEXUS framework uses separate components for sensor management and communication, and use spatial models with multiple representations to model the physical world stored in distributed databases.

The NEXUS project has isolated the sensors in a separate module. This idea was also used when creating the architecture of the Nidaros framework. Another idea that was used from the NEXUS project was to provide standardized interfaces between the client and server. The NEXUS project uses a distributed database to store the necessary information. In the Nidaros framework, we use a centralized database for storing location related information. However, the file server might be replicated for performance purposes.

The Framework for Location-Aware ModElling (FLAME) is a configurable, generic software framework for development of location-aware applications [CNS04]. FLAME provides support for multiple sensor technologies, provides a simple spatial model for the representation of locatable entities. In addition, FLAME provides a simple event architecture for the presentation of location information to applications, and a queryable location database. The framework and its applications are largely event-driven in order to accommodate the real-time nature of the location information that they handle. The database holds the initial states (like static regions), and it also holds a synopsis of the real-time location information. A region manager stores and retrieves regions from the

database. A spatial relation manager generates application-related events to satisfy currently active subscriptions. The event adapters generate events, e.g., when a person has moved a "Person Movement Event" is generated. Events allow transfer of location information from FLAME to the application. In addition, non event-based applications can receive information from FLAME. Several tools to facilitate the creation and management of location-aware applications are included in FLAME. A graphical World Builder tool allows the user to quickly construct a model of his world without the need for programming. The SRMConfigurator tool is an application that can read a XML description of how FLAME should be configured, and it automatically creates and configures FLAME according to this description. FLAME also provides interface modules for several location technologies. A template is also provided which allows the user to support other location technologies.

FLAME uses a Region Manager to retrieve and store regions from the database. This idea was also used in the Nidaros framework. We were also inspired by the World Builder tool in FLAME when we created the creator tool. Our framework could also benefit from a more extensive graphical world builder tool. A fundamental difference between the Nidaros and the FLAME framework is that FLAME is event-driven. When an event appears, new information is sent to the applications. In the Nidaros framework, the client applications will send a request when a service is needed from the server. The advantage of our approach is that we avoid overloading the mobile devices with incoming events. Also less traffic is demanded over the wireless network in our approach, since communication is happening only when needed. The disadvantage with our approach is that it does not support immediate actions when some specific events occur. For location-aware systems where safety is important, an event-based system should be used.

Dey and Abowd [Dey00, DA01] presents requirements and a conceptual framework for handling context information. The requirements to be fulfilled by the framework are *Separation of concerns*, *Context interpretation*, *Transparent, distributed communications*, *Constant availability of context acquisition*, *Context storage and history*, and *Resource discovery*. The conceptual framework for handling context presented by [DA01] suggests the use of **context widgets** to provide access for applications to context information from their operating environment. The context widget is regarded as a mediator between applications and the operating environment, insulating applications from context acquisition concerns. Context-specific operations are addressed in the framework by four categories of components: interpreters, aggregators, services and discoverers. The framework defined by Dey and Abowd focuses more on the management of various context sources and to represent these context sources in an application. The Nidaros framework uses the location server to deal with the context sources. Since our framework only focus on location, a rigorous framework for representation of various context sources is not needed. However, a possible extension of the Nidaros framework could be to support other context sources than location.

17.6 Conclusion

Although there exist different types of location-aware applications, there are still many location-aware services that have not been explored. Many of the existing location-aware applications are tailored just for one purpose. In this paper, we have presented the Nidaros framework to be used to implement location-aware applications. Our framework provides support for the development phase, the operation phase and the analysis phase. In the development phase the creator tool is used to add needed information into the database to be used by the final application. Further, the client templates can used for faster development of mobile clients with some basic functionality. In the operation phase, the runtime system manages all interaction between clients and the server including communication of multimedia files and determination of clients' positions using a location server. For the analysis phase a statistic tool can be used to analyse the usage of the location-aware application, detect possible bottlenecks of the system, and see what objects that are most popular.

Acknowledgement

Several institutions have been involved in the creation and demonstration of the Nidaros framework. The core work of developing the framework was done by the members of the MOWAHS-project[6] (including students and researchers) at Norwegian University of Science and Technology (NTNU), Telenor R & D and Klipp og Lim Media. To be able to test the framework in a real environment we also involved people from the Nidaros Cathedral and people from RadioNor Communications that configured and provided us RadioEyes. The development of the framework was also made possible by using the PATS laboratory[7]. The PATS laboratory is related to the project ARTS (Arena for Research on advanced Telecom Services), sponsored by the Norwegian Research Council, where NTNU, Telenor R & D, Abelia, Ericsson and Hewlett-Packard are main partners.

[6] More information about the MOWAHS project is available at http://www.mowahs.com
[7] More information about the PATS laboratory is available at http://www.pats.no

Part V

Appendix

APPENDIX A

Other Publications

This appendix contains abstracts and description of papers peer-reviewed and accepted at international conferences. The papers are either not directly related to the topics presented in this thesis, or is partly covered by other papers included in the thesis.

AP1: COTS Products Characterization [TJSW02]

A way to learn about Commercial Off-The-Shelf (COTS) products is to define a set of characteristics or attributes and then to collect information about these attributes. In an industrial context, the attributes used to select COTS clearly depend on project specific goals. In our educational context we made an attempt to define general COTS characterization attributes. The resulting framework provides a structure, which facilitate the learning process. Our proposed attributes have several similarities with the generic evaluation attributes defined by ISO 9126. A comparison with such a standard provides a deeper insight into the problem of characterizing COTS products.

Author(s): Marco Torchiano, Letizia Jaccheri, Carl-Fredrik Sørensen, and Alf Inge Wang.
Published where: The 14th International Conference on Software Engineering and Knowledge Engineering (SEKE'02), July 15-19, 2002, Ischia, Italy.

AP2: A Comparison of Two Different Java Technologies to Implement a Mobile Agent System [WS03]

This paper describes an evaluation of the two technologies Aglets from IBM Japan and JavaSpaces from Sun Microsystems used to implement the same mobile agent system. The Aglets framework was initially designed to support mobile agents, while the JavaSpaces framework was not. In this evaluation we have looked at issues like adaptability, configurability, extensibility, location transparency, and abstraction level. We have also measured the effort that was used (in hours) and the lines of code necessary to implement the same mobile agent system using the two different technologies. We found that there are several advantages by using JavaSpaces instead of Aglets for implementing mobile agent systems even if it was not intentionally designed to support this. Our evaluation does not look at the difference in performance of the two technologies.

Author(s): Alf Inge Wang and Carl-Fredrik Sørensen.
Published where: IASTED International Conference on Applied Informatics (AI'2003), Innsbruck, Austria, 2003.

AP3: A Mobile Agent Architecture for Heterogeneous Devices [WSI03]

This paper describes an approach for running mobile agents on various devices from mobile phones and Personal Digital Assistants (PDAs) to powerful PCs. Most mobile devices suffer from limited computation resources (memory and processors), limited network connection and bandwidth, and limited battery life. Mobile agents are a promising technology for minimizing the problems described above. However, most mobile agent systems today are very resource demanding both for the client and the server. In this paper we propose a simple mobile agent architecture that makes it possible to access a mobile agent system on various devices. This architecture proposes that clients will state their capabilities. Based on these capabilities, the client will either run the full mobile agent on the device or only run a light-weight version of the agent on the device. In our new approach, the mobile agents are basically the same on all clients, but for small mobile devices the code of the mobile agent is removed. This means that for mobile devices with minimal resources, only the data of the agent can be changed. The code of this agent is stored at the server. When the agent returns to the server, the two parts are joined and the agent is ready to be executed. The joined mobile agent can migrate to other agent servers and clients.

Author(s): Alf Inge Wang, Carl-Fredrik Sørensen, and Eva Indal.
Published where: IASTED International Conference on Wireless and Optical Communications (WOC'2003), Banff, Canada, July 14-16, 2003, pages 584–590.

AP4: An Evaluation of a Mobile Task Reporting System for Different Mobile Devices [SWØHB03]

This paper describes an evaluation of using three different types of mobile devices as clients in a mobile task reporting system. The heterogeneity of hardware and software introduced by various mobile devices has made it harder for software developers to make systems that can support this broad spectrum of devices. We have made a prototype of a system to handle management of tasks for mobile workers. Mobile workers are offered to bring parts of the system on mobile devices, making it possible to update task status and write small reports on the spot when finishing a task. We have in this paper evaluated how well the system has worked for the different devices based on several evaluation criteria like screen size, input device, physical device size, battery life, connectivity etc. In this evaluation we have used a portable PC, a PDA, and a mobile phone. We found that the system should provide different support dependent on the device. The mobile phone was found to be useful for checking task states and for limited information browsing. The PDA was very useful for writing end-reports when finishing tasks (on the actual location), and the portable PC was best for managing tasks (getting overview of the different tasks and states), and for task delegating tasks. We also discuss the problem of keeping data consistent when several off-line mobile devices can update the same data, and we indicate some solutions to this problem.

Author(s): Carl-Fredrik Sørensen, Alf Inge Wang, Øystein Hoftun, and Kenneth Aron Bjørkhaugen.
Published where: IASTED International Conference on Software Engineering and Applications 2003 (SEA'2003), Marina Del Rey, California, USA, November 3-5, 2003, pages 751–756.

AP5: Case Study: Use of the Mobile Tool HandyMan for Mobile Work [WSFS03]

Mobile electronic equipment such as laptops, PDAs and mobile phones have the recent years become very important tools for managing mobile work more efficient. In this paper we present a case study of such a tool called HandyMan. We have looked at how five Norwegian companies use this tool, how they synchronise their mobile device with other systems, how they deal with ad-hoc tasks, and what the major benefits and problems are. The motivation for this case study is to identify typical issues that must be taken into account when designing a system for supporting mobile work. In the case study we found that the non-functional requirements are very important in the design of a mobile system. Also requirements to the mobile device itself are very important to a system that should be used in a hostile environment. There are also organisational issues that must be considered before introducing a mobile system in a company. Lastly, we identified a need for better off-line support to help the users to know when to synchronise and charge, and to handle ad-hoc tasks.

Author(s): Alf Inge Wang, Carl-Fredrik Sørensen, Thale Christina Fritzner, and Eldrid Schei.
Published where: IASTED International Conference on Software Engineering and Applications 2003 (SEA'2003), Marina Del Rey, California, USA, November 3-5, 2003, pages 757–762.

AP6: Cooperating Sentient Vehicles for Next Generation Automobiles [SBF+04]

It is becoming clear that location-aware intelligent transportation systems will be one of the most promising upcoming applications for next generation vehicles. The driving force behind this is the introduction of pervasive high performance wireless networks and location sensing technologies, such as GPS and roadside detection systems. Intelligent transportation systems utilise intervehicle cooperation without human assistance to provide autonomous vehicle navigation from a given source to a predetermined destination. The resultant sentient vehicles are 'context-aware' autonomous cars that form cooperative 'flotillas' of peers using mobile ad hoc network environments (MANETs). In this paper we report on our experiences of building a component framework based middleware architecture designed to meet the challenges of such environments. We show how such a framework can be used to engineer a proof of concept cooperating sentient vehicle application, and highlight the research challenges raised.
Author(s): Thirunavukkarasu Sivaharan, Gordon Blair, Adrian Friday, Maomao Wu, Hector Duran-Limon, Paul Okanda, and Carl-Fredrik Sørensen.
Published where: 1st ACM International Workshop on Applications of Mobile Embedded Systems (WAMES 2004), Boston, MA, USA, 6 June 2004.

AP7: Context-Aware Middleware for Pervasive and Mobile Ad Hoc Environments

Recent advances in the area of mobile ad hoc computing and pervasive computing have driven the emergence of new challenges. For example, the "Intelligent Environment" or "Smart Environment" has become one of the key research areas in the pervasive computing arena. Mobile ad hoc scenarios also include time-critical applications such as an autonomous vehicle system in which vehicles are able to operate independently and cooperate with each other to avoid collisions. In this paper, we present a context-aware middleware architecture for the support of both pervasive and mobile ad hoc environments.

Author(s): Hector A. Duran-Limon, Gordon S. Blair, Adrian Friday, Thirunavukkarasu Sivaharan, Maomao Wu, Paul Okanda, and Carl-Fredrik Sørensen.
Published where: Mobile Computing Workshop (TCM 2004), Colima, Colima, Mexico - September 21, 2004.

AP8: A Three Level Framework for Process Support: The MOWAHS Approach [WSC04]

A common assumption for many process-centred support environments is that they provide the same process support at different levels of the organization. We believe that the required process support from one level to another of an organization varies. In this paper we propose a framework that divides process support into three levels: Individual, group and team level. Further, we characterize each level and describe the required process support and type of process modelling language for each level. We also look at what interfaces are needed between the levels. Finally we use the framework to characterize an eXtreme Programming (XP) development process. Our framework focuses only on the project level of software development and does not consider management processes above this level.

Author(s): Alf Inge Wang, Carl-Fredrik Sørensen and Reidar Conradi.
Published where: 2004 International Conference on Software Engineering Research and Practice – Session on Team-based Software Engineering (TBSE'04). Las Vegas, Nevada, USA, June 21-24, 2004, pages 10–16.

Part VI

Glossary

A

ATM – Asynchronous Transfer Mode

B

BlueTooth – Short-range wireless technology.
BPM – Business Process Management.

C

CAGIS – Cooperative Agents in Global Information Space.
CGI – Common Gateway Interface.
COM – Component Object Model.
CORBA – Common Object Request Broker Architecture.
CSCW – Computer-Supported Collaborative Work.
CSE – Cooperative Software Engineering.

D

DCE – Distributed Computing Environment.
DCOM – Distributed Common Object Model.
DIAS – Distributed Intelligent Agent System.
Distributed System [CDK01] – A system in which hardware or software components located at networked computers communicate and co-ordinate their actions only by passing messages. Computing devices may be connected to a wide range of networks as for instance the Internet, mobile phone networks, corporate networks, home networks or to combinations of these.
DSDM – Dynamic Systems Development Method.

G

GPRS – General Packet Radio Service. Addition to GSM that supports data packages in addition to voice traffic.
GSM – Global System for Mobile telephony. Second generation (2G) mobile telecommunication system.

H

HTML – HyperText Markup Language.
HSCSD – High Speed Circuit Switched Data.
HTTP – HyperText Transmission Protocol.

I

IDL – Interface Definition Language.
IP – Internet Protocol.
IR – Infrared.
IrDA – Infrared Data Association.
ITU – International Telecommunications Union.

L

LAN – Local Area Network.

M

MANET – Mobile Ad hoc Network.
MDCA – Microsoft Distributed Component Architecture.
Middleware – Software that connects two otherwise separate applications [ISP00].
MOWAHS – Mobile Work Across Heterogeneous Systems.
MTS – Microsoft Transaction Service.

O

OMG – Object Management Group.

P

P2P – Peer-to-Peer.
PDA – Personal Digital Assistant.
PML – Process Modelling Language.
Process definition – The representation of a business process in a form which supports automated manipulation, such as modelling, or enactment by a workflow management

system. The process definition consists of a network of activities and their relationships, criteria to indicate the start and termination of the process, and information about the individual activities, such as participants, associated IT applications and data, etc. [Coa99].
PSEE – Process-centred Software Engineering Environment.

Q

QA – Quality Assurance.
QoS – Quality of Service.

R

RF – Radio Frequency.
RFID – Radio Frequency Identification.
RMI – Remote Method Invocation.
RPC – Remote Procedure Protocol.

S

SMS – Short Message Service.
Software engineering – An engineering discipline which is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use [Som01].
SOAP – Simple Object Access Protocol.
SPT – Software Process Technology.

T

TCP/IP – Transmission Carrier Protocol/Internet Protocol.

U

UMTS – Universal Mobile Telecommunications System. The European variant of the third generation (3G) mobile telecommunication system.
URL – Universal Resource Locator.
USB – Universal Serial Bus.

W

WAN – Wide-Area Network.
WAP – The Wireless Application Protocol.
WirelessMAN – Wireless Metropolitan Area Network.
WLAN – Wireless LAN, based on IEEE 802.11, also called IP-zone or hotspot.
WML – Wireless Markup Language.
Workflow – The **automation** of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules [Coa99].
Workflow management system (WFMS) – A system that defines, creates and manages the execution of workflows through the use of software, running on one or more workflow engines, which is able to interpret the process definition, interact with workflow participants and, where required, invoke the use of IT tools and applications [All01].
WWW – World-Wide Web.

X

XML – Extended Markup Language.
XP – Extreme Programming.
XSLT – Extensible Stylesheet Language for Transformation.

Part VII

Bibliography

Bibliography

[AAH+97] Gregory D. Abowd, Christopher G. Atkeson, Jason Hong, Sue Long, Rob Kooper, and Mike Pinkerton. Cyberguide: A Mobile Context-Aware Tour Guide. *Wireless Networks*, 3(5):421–433, 1997.

[AbW01] Adnan Al-bar and Ian Wakeman. A Survey of Adaptive Applications in Mobile Computing. In *21th International Conference on Distributed Computing Systems Workshops (ICDCSW'01)*, pages 246–251, Mesa, Arizona, United States, 2001. IEEE.

[AEH+02] Gregory D. Abowd, Maria Ebling, Guerney Hunt, Hui Lei, and Hans-Werner Gellersen. Guest Editors' Introduction: Context-Aware Computing. *IEEE Pervasive Computing*, 1(3):22–23, July–September 2002.

[Agh02] Gul A. Agha. Introduction – Adaptive Middleware. *Communications of the ACM*, 45(6):30–32, June 2002.

[AIM05] Gregory D. Abowd, Liviu Iftode, and Helena Mitchell. Guest Editors' Introduction: The Smart Phone–A First Platform for Pervasive Computing. *IEEE Pervasive Computing*, 4(2):18–19, April–June 2005.

[Alg04] Offshore Algorithms. Rockyroad/jrra. Web: http://www.jrra.org, 2004.

[All01] Rob Allen, editor. *Workflow: An Introduction*, pages 1–24. WfMC, 2001.

[Amb01] Vincenzo Ambriola, editor. *Software Process Technology – 8th European Workshop (EWSPT'2001)*, Witten, Germany, June 2001. Springer-Verlag. LNCS 2077.

[AP05] C. W. Akhil Arora and K. S. Pabla. jxme: JXTA Platform Project. http://www.jxme.org, February 2005.

[AS94] Kenneth R. Abbott and Sunil K. Sarin. Experiences with Workflow Management: Issues for the Next Generation. In *Conference on Computer Supported Cooperative Work*, pages 113–120, Chapel Hill, North Carolina, United States, 1994. ACM Press.

[Ash98] Steven Ashley. Smart Cars and Automated Highways. *Mechanical Engineering Magzine*, May 1998. http://www.memagazine.org/backissues/may98.

[Ava03] AvantGo. Web browsing and caching for wireless devices. http://avantgo.com, Accessed February 18th 2003.

[Bar97] Jakob E. Bardram. Plans as Situated Action: An Activity Theory Approach to Workflow Systems. In Tom Rodden, John Hughes, and Kjeld Schmidt, editors, *5th European Conference on Computer Supported Cooperative Work*, pages 17–32, Lancaster University, UK, 7-11 September 1997. Kluwer Academic Publishers.

[Bar05] Jakob E. Bardram. Activity-based computing: support for mobility and collaboration in ubiquitous computing. *Personal Ubiquitous Comput.*, 9(5):312–322, 2005.

[Bas92] Victor R. Basili. The Experimental Paradigm in Software Engineering. In H. Dieter Rombach, Victor R. Basili, and Richard W. Selby, editors, *Experimental Software Engineering Issues: Critical Assessment and Future Directions*, pages 3–12, Dagstuhl Castle, Germany, September 14-18 1992. Springer Verlag. LNCS 706.

[BC04] Gregory Biegel and Vinny Cahill. A Framework for Developing Mobile, Context-aware Applications. In *Second IEEE International Conference on Pervasive Computing and Communications (PerCom'04)*, pages 361–365, Orlando, Florida, March 14-17 2004. IEEE Computer Society Press.

[BCA+01] Gordon S. Blair, Geoff Coulson, Anders Andersen, Lynne Blair, Michael Clarke, Fabio Costa, Hector Duran-Limon, Tom Fitzpatrick, Lee Johnston, Rui Moreira, Nikos Parlavantzas, and Katia Saikoski. The Design and Implementation of Open ORB version 2. *IEEE Distributed Systems Online*, 2(6), 2001. http://dsonline.computer.org/0106/features/bla0106_print.htm.

[BCD+97] Gordon Blair, Geoff Coulson, Nigel Davies, Philippe Robin, and Tom Fitzpatrick. Adaptive Middleware for Mobile Multimedia Applications. In *8th International Workshop on Network and Operating System Support for*

Digital Audio and Video (NOSSDAV '97), pages 259–273, St. Louis, Missouri, USA, 19-21 May 1997.

[BCN+96] C. Basile, S. Calanna, E. Nitto, A. Fuggetta, and M. Gemo. Mechanisms and Policies for Federated PSEEs: Basic Concepts and Open Issues. In Carlo Montangero, editor, *5th European Workshop on Software Process Technology (EWSPT'1996)*, pages 86–91, Nancy, France, October 1996. Springer Verlag. LNCS 1149.

[BDK99] Abraham Bernstein, Chrysanthos Dellarocas, and Mark Klein. Towards Adaptive Workflow Systems: CSCW-98 Workshop Report. *ACM SIGGROUP Bulletin*, 20(2):54–56, 1999.

[BE01] Victoria Bellotti and Keith Edwards. Intelligibility and Accountability: Human Considerations in Context-Aware Systems. *Human-Computer Interaction (HCI) Journal. Special Issue: Context-Aware Computing*, 16(2–4):193–212, 2001.

[Bec99] Kent Beck. *eXtreme Programming Explained: Embrace Change*. The XP series. Addison-Wesley, Reading, Mass., US, 1999.

[BFJ+01] George Buchanan, Sarah Farrant, Matt Jones, Harold Thimbleby, Gary Marsden, and Michael Pazzani. Improving Mobile Internet Usability. In *Tenth International Conference on World Wide Web*, pages 673–680, Hong Kong, Hong Kong, 2001. ACM Press.

[BFS+02] Rajesh Krishna Balan, J. Flinn, Mahadev Satyanarayanan, S. Sinnamohideen, and H. Yang. The Case for Cyber Foraging. In *10th ACM SIGOPS European Workshop*, pages 87–92, Saint-Emilion, France, September 2002. ACM Press, New York, NY, USA.

[BGH02] Barry Brown, Nicola Green, and Richard Harper, editors. *Wireless World: Social and Interactional Aspects of the Mobile Age*. Computer Supported Cooperative Work. Springer, 2002.

[BGMPW00] Orkut Buyukkokten, Hector Garcia-Molina, Andreas Paepcke, and Terry Winograd. Power Browser: Efficient Web Browsing for PDAs. In *Conference on Human Factors in Computing Systems (CHI'00)*, pages 430–437, The Hague, The Netherlands, 2000. ACM Press.

[BHR+99] Staffan Björk, Lars Erik Holmquist, Johan Redström, Ivan Bretan, Rolf Danielsson, Jussi Karlgren, and Kristofer Franzén. WEST: A Web Browser for Small Terminals. In *12th Annual ACM Symposium on User Interface Software and Technology*, pages 187–196, Asheville, North Carolina, United States, 1999. ACM Press.

[BK95] Douglas P. Bogia and Simon M. Kaplan. Flexibility and Control for Dynamic Workflows in the WORLDS Environment. In *Conference on Organizational Computing Systems*, pages 148–159, Milpitas, California, United States, 1995. ACM Press.

[BKS05] Frank Bomarius and Seija Komi-Sirviö, editors. *Product Focused Software Process Improvement: 6th International Conference, PROFES 2005*, Oulu, Finland, June 13-18 2005. Springer Verlag. LNCS 3547.

[BN01] Grete Bach and Torbjørn Nystadnes. IT-behov innen pleie- og omsorgstjenesten. Technical Report 6/01, KITH – Kompetansesenter for IT i helsevesenet, March 23rd, 2001.

[Bol00] Gregory Alan Bolcer. Magi: An Architecture for Mobile and Disconnected Workflow. *IEEE Internet Computing*, 4(3):46–54, May–June 2000.

[Bor05] Gaetano Borriello. RFID: Tagging the World – Introduction. *Communications of the ACM*, 48(9):34–37, 2005.

[Bos00] Jan Bosch. *Design & Use of Software Architectures*. Addison-Wesley Longman Publishing Co., Inc., 2000.

[BPS02] Federico Bergenti, Agostino Poggi, and Matteo Somacher. A Collaborative Platform for Fixed and Mobile Networks. *Communications of the ACM*, 45(11):39–44, November 2002.

[BSK95] Israel Ben-Shaul and Gail E. Kaiser. *A Paradigm for Decentralized Process Modeling*. Kluwer Academic Publishers, Boston/Dordrecht/London, 1st edition, 1995.

[BSK98] Israel Z. Ben-Shaul and Gail Kaiser. Federating Process-Centered Environments: The Oz Experience. *[DF98]*, 5(1):97–132, 1998.

[Bus95] Uwe Busbach. Distributed Work Management: An Application Area for Mobile Computing. In *28th Annual Hawaii International Conference on System Sciences (HICSS'95)*, pages 34–41, Hawaii, United States, 1995. IEEE Press.

[BW02] J. Brendon and J. Wilson. *JXTA*. New Riders Publishing, 2002.

[CBCP01] Michael Clarke, Gordon S. Blair, Geoff Coulson, and Nikolaos Parlavantzas. An Efficient Component Model for the Construction of Adaptive Middleware. In *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware'2001)*, pages 160–178, Heidelberg, Germany, November 2001. Springer-Verlag, London, UK. LNCS 2218.

[CBCP02] Geoff Coulson, Gordon S. Blair, Michael Clark, and Nikolaos Parlavantzas. The Design of a Highly Configurable and Reconfigurable Middleware Platform. *ACM Distributed Computing Journal*, 15(2):109–126, April 2002.

[CBDF99] Keith Cheverst, Gordon S. Blair, Nigel Davies, and Adrian Friday. The Support of Mobile-Awareness in Collaborative Groupware. *Personal and Ubiquitous Computing*, 3(1 & 2):33–42, 1999.

[CC03] Alvin T.S. Chan and Siu-Nam Chuang. MobiPADS: A Reflective Middleware for Context-Aware Mobile Computing. *IEEE Transactions on Software Engineering*, 29(12):1072–1085, 2003.

[CDE+00] Pierre-Yves Cunin, Sami Dami, Jacky Estublier, Gianpaolo Cugola, Alfonso Fuggetta, H. Verjus, F. Pacull, and M. Rivière. Support for Software Federations: The PIE Platform. In *[Con00]*, pages 38–52, Kaprun, Austria, February 21-25 2000. Springer Verlag. LNCS 1780.

[CDK01] George Couloris, Jean Dollimore, and Tim Kindberg. *Distributed Systems – Concepts and Design*. The International Computer Science Series. Addison-Wesley, third edition, 2001.

[CDMF00] Keith Cheverst, Nigel Davies, Keith Mitchell, and Adrian Friday. Experiences of Developing and Deploying a Context-Aware Tourist Guide: The GUIDE Project. In *Sixth Annual International Conference on Mobile Computing and Networking*, pages 20–31, Boston, Massachusetts, United States, 2000. ACM Press.

[CEM01] Licia Capra, Wolfgang Emmerich, and Cecilia Mascolo. Exploiting Reflection and Metadata to build Mobile Computing Middleware. In *Workshop on Middleware for Mobile Computing. Co-located with Middleware 2001*, Heidelberg, Germany, November 2001.

[CEM02] Licia Capra, Wolfgang Emmerich, and Cecilia Mascolo. A Micro-Economic Approach to Conflict Resolution in Mobile Computing. *ACM SIGSOFT Software Engineering Notes*, 27(6):31–40, 2002.

[CL97] Imrich Chlamtac and Yi-Bing Lin. Guest Editors' Introduction–Mobile Computing: When Mobility Meets Computation. *IEEE Transactions on Computers*, 46(3):257–259, March 1997.

[CN00] Reidar Conradi and Mads Nygård. MObile Work Across Heterogeneous Systems (MOWAHS); A research proposal for 2001-2004. Technical report, Department of Computer and Information Science, NTNU, Trondheim, Norway, October 2000.

[CNS04] George Coulouris, Hani Naguib, and Kam Samugalingam. Flame: An Open Framework for Location-Aware Systems. http://wwwlce.eng.cam.ac.uk/qosdream/ Publications/flame.pdf, November 2004. Submitted for publication.

[Coa99] The Workflow Management Coalition. Workflow Management Coalition - Terminology & Glossary. Technical report, The Workflow Management Coalition (WfMC), February 1999. Document Number WFMC-TC-1011.

[Com04] RadioNor Communications. Cordis RadioEye. http://www.radionor.no/, March 2004.

[Con00] Reidar Conradi, editor. *Software Process Technology – 7th European Workshop (EWSPT'2000)*, Kaprun, Austria, February 21–25 2000. Springer-Verlag. LNCS 1780.

[COR03] CORTEX. Final definition of CORTEX programming model. Technical Report CORTEX Deliverable D6 version 1.0, CORTEX, April, 2003.

[CTC+97] Henry Chang, Carl Tait, Norman Cohen, Moshe Shapiro, Steve Mastrianni, Rick Floyd, Barron Housel, and David Lindquist. Web Browsing in a Wireless Environment: Disconnected and Asynchronous Operation in ARTour Web Express. In *Third Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pages 260–269, Budapest, Hungary, 1997. ACM Press.

[CV01] António Casimiro and Paulo Veríssimo. Using the Timely Computing Base for Dependable QoS Adaptation. In *20th IEEE Symposium on Reliable Distributed Systems*, pages 208–217, New Orleans, USA, October 2001. IEEE Computer Society Press.

[CW03] Reidar Conradi and Alf Inge Wang, editors. *Empirical Methods and Studies in Software Engineering*. Springer, Heidelberg, Germany, 2003. LNCS 2765.

[DA00] Anind K. Dey and Gregory D. Abowd. Towards a Better Understanding of Context and Context-Awareness. In *Workshop on The What, Who, Where, When, and How of Context-Awareness, part of Conference on Human Factors in Computing Systems (CHI 2000)*, The Hague, The Netherlands, April 2000. ACM Press.

[DA01] Anind K. Dey and Gregory D. Abowd. A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. *Human-Computer Interaction (HCI) Journal. Special Issue: Context-Aware Computing*, 16(2–4):97–166, 2001.

[DBCF94] Nigel Davies, Gordon Blair, Keith Cheverst, and Adrian Friday. Supporting Adaptive Services in a Heterogeneous Mobile Environment. In *IEEE Workshop on Mobile Computing Systems and Applications*, pages 153–157, Santa Cruz, CA, USA, December 1994. IEEE Computer Society Press.

[DEA98] S. Dami, J. Estublier, and M. Amiour. APEL: A Graphical Yet Executable Formalism for Process Modeling. In E. Nitto and Alfonso Fuggetta, editors, *Process Technology*, pages 61–96, Politecnico di Milano and CEFRIEL, 1998. Kluwer Academic Publishers.

[Dey00] Anind K. Dey. *Providing Architectural Support for Building Context-Aware Applications*. PhD thesis, College of Computing, Georgia Institute of Technology, Atlanta, GA, USA, December 2000.

[Dey01] Anind K. Dey. Understanding and Using Context. *Personal and Ubiquitous Computing*, 5(1):4–7, 2001.

[DF98] Elisabetta Di Nitto and Alfonso Fuggetta, editors. *Process Technology: Special Issue*, volume 5 of *Journal on Automated Software Engineering*. Kluwer Academic Publisher, January 1998.

[DFWB98] N. Davies, A. Friday, S. P. Wade, and G. S. Blair. L^2imbo: A Distributed Systems Platform for Mobile Computing. *Mobile Networks and Applications – Special Issue on Protocols and Software Paradigms of Mobile Networks*, 3(2):143–156, August 1998. Internal report number MPG-97-03.

[DLS02] Per Dahlberg, Fredrik Ljungberg, and Johan Sanneblad. Proxy Lady: Mobile Support for Opportunistic Interaction. *Scandinavian Journal of Information Systems*, 14(1):3–17, 2002.

[DMPD99] Henrique Domingos, J. Legatheaux Martins, N. M. Preguiça, and Sérgio M. Duarte. A Workflow Architecture to Manage Mobile Collaborative Work. In *Proceedings of Portuguese Workshop on Mobile Computing*, pages 49–61, Tomar, Portugal, November 1999.

[dPL00] Julio Cesar Sampaio do Prado Leite. Is there a Gap between RE Research and RE Practice? In *Fourth International Conference on Requirement Engineering (ICRE 2000)*, pages 73–74. IEEE Computer Society, 2000.

[DRD+00] Alan Dix, Tom Rodden, Nigel Davies, Jonathan Trevor, Adrian Friday, and Kevin Palfreyman. Exploiting Space and Location as a Design Framework for Interactive Mobile Systems. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 7(3):285–321, 2000.

[DS00] Per Dahlberg and Johan Sanneblad. The use of Bluetooth enabled PDAs. In *Proceedings of IRIS-23*, Uddevalla, Sweden, August 12-15 2000.

[DSAF99] Anind K. Dey, Daniel Salber, Gregory D. Abowd, and Masayasu Futakawa. The Conference Assistant: Combining Context-Awareness with Wearable Computing. In *3rd International Symposium on Wearable Computers*, pages 21–28, San Francisco, California, USA, October 1999. IEEE.

[DT93] Flavio DePaoli and Francesco Tisato. Language Constructs for Cooperative Systems Design. In *[SP93]*, pages 329–343, 1993.

[Dul02] Ken Dulaney. The Wireless and Mobile Market Starts to Mature. In Monica Basso, editor, *PREDICTS 2003, Gartner's Predictions for the year ahead - Wireless & Mobile*. Gartner, Inc., 22 November 2002.

[ea94] Reidar Conradi et al. *EPOS: Object-Oriented and Cooperative Process Modelling*, pages 33–70. Research Studies Press/John Wiley & Sons, 1994.

[EGR91] Clarence A. Ellis, S. J. Gibbs, and G. L. Rein. Groupware – Some Issues and Experiences. *Communications of the ACM*, 34(1):39–58, January 1991.

[EJÖ02] Mattias Esbjörnsson, Oskar Juhlin, and Mattias Östergren. The Hocman Prototype - Fast Motor Bikers and Ad Hoc Networking. In *1st International Conference on Mobile and Ubiquitous Multimedia (MUM 2002)*, Oulu, Finland, December 11-13 2002.

[Ell00] Arne Munch Ellingsen. Mobil Profil, trådløs elektronisk pasientjournal for hjemmetjenesten, 2000. (In Norwegian).

[ELMB92] Ahmed K. Elmagarmid, Yungho Leu, James G. Mullen, and Omran Bukhres. Introduction to Advanced Transaction Models. In Ahmed K. Elmagarmid, editor, *Database Transacation Models for Advanced Applications*, pages 35–52. Morgan Kaufmann, 1992.

[Emm02] Wolfgang Emmerich. Distributed Component Technologies and their Software Engineering Implications. In *24th International Conference on Software Engineering (ICSE'2002)*, pages 537–546, Orlando, Florida, 2002. ACM Press.

[EMP99] Yrjö Engeström, Reijo Miettinen, and Raija-Leena Punamäki, editors. *Perspectives on Activity Theory*. Learning in doing: Social, Cognitive, and Computational Perspectives. Cambridge University Press, 1999.

[FBCC02] Adrian Fitzpatrick, Gregory Biegel, Siobhán Clarke, and Vinny Cahill. Towards a Sentient Object Model. In *Workshop on Engineering Context-Aware Object Oriented Systems and Environments (ECOOSE)*, Seattle, WA, USA, November 2002.

[FDBC99] Adrian Friday, Nigel Davies, Gordon S. Blair, and Keith Cheverst. Developing Adaptive Applications: The MOST Experience. *Journal of Integrated Computer-Aided Engineering*, 6(2):143–157, 1999.

[FFK+02] Margaret Fleck, Marcos Frid, Tim Kindberg, Eamonn O'Brien-Strain, Rakhi Rajani, and Mirjana Spasojevic. From Informing to Remembering: Ubiquitous Systems in Interactive Museums. *IEEE Pervasive Computing*, 1(2):13–21, April–June 2002.

[Fil04] Roberto Silveira Silva Filho. Awareness and Privacy in Mobile Wearable Computers. IPADS: Interpersonal Awareness Devices. Final report for ICS234A - Virtual Colocation, Web: http://www1.ics.uci.edu/~rsilvafi/papers/VirtualColocationFinalPaper.pdf, 2004.

[Fin97] Kathleen E. Finn. *Video-Mediated Communication*. Lawrence Erbaum Associates, Inc., April 1 1997.

[Fin00] Anthony Finkelstein, editor. *The Future of Software Engineering*, 22nd International Conference on Software Engineering (ICSE'2000), Limerick, Ireland, 2000. ACM Press.

[For82] Charles Lanny Forgy. RETE: A Fast Algorithm for the Many Patterns/-Many Objects Pattern Match Problem. *Artificial Intelligence*, 19(1):17–37, September 1982.

[FRF02] Alfred Fent, Herbert Reiter, and Burkhard Freitag. Design for Change: Evolving Workflow Specifications in ULTRAflow. In A. Banks Pidduck, J. Mylopoulos, C.C. Woo, and M. Tamer Ozsu, editors, *14th International Conference on Advanced Information Systems Engineering (CAiSE '02)*, pages 516–534, Toronto, Canada, May 27-31 2002. Springer-Verlag. LNCS 2348.

[Fri96] Adrian Friday. *Infrastructure Support for Adaptive Mobile Applications*. PhD thesis, Lancaster University, Lancaster, England, 1996.

[Fug00] Alfonso Fuggetta. Software Process: A Roadmap. In A. Finkelstein, editor, *[Fin00]*, pages 27–34, Limerick, Ireland, June 2000. ACM Press.

[FZ94] George H. Forman and John Zahorjan. The Challenges of Mobile Computing. *Computer*, 27(4):38–47, April 1994.

[GA99] Avelino J. Gonzalez and Robert Ahlers. Context-Based Representation of Intelligent Behaviour in Training Simulations. *Transactions of the Society for Computer Simulation International*, 15(4):153–166, 1999.

[Gar00] David Garlan. Software Architecture: A Roadmap. In A. Finkelstein, editor, *[Fin00]*, pages 91–101, Limerick, Ireland, June 2000. ACM Press.

[GCP03] Franca Garzotto, Tullio Salmon Cinotti, and Massimaliano Pigozzi. Designing Multi-Channel Web Frameworks for Cultural Tourism Applications: the MUSE Case Study. In *Museums and the Web 2003*, Charlotte, North Carolina, USA, March 19–22 2003.

[GVR02] R.L. Glass, I. Vessey, and V. Ramesh. Research in software engineering: an analysis of the literature. *Information and Software Technology*, 44(8):491–506, 2002.

[GWH02] John Grundy, Xing Wang, and John Hosking. Building Multi-device, Component-based, Thin-client Groupware: Issues and Experiences. In *Third Australasian Conference on User Interfaces*, pages 71–80, Melbourne, Victoria, Australia, 2002. Australian Computer Society, Inc.

[HB01] Øystein Hoftun and Kenneth Aron Bjørkhaugen. Task Reporting System. Technical report, Dep. of Computer and Information Science, NTNU, Norway, November 2001.

[HCL00] Martin Hope, Tom Chrisp, and Nigel Linge. Improving Co-operative Working in the Utility Industry through Mobile Context Aware Geographic Information Systems. In *8th ACM International Symposium on Advances in Geographic Information Systems (GIS'00)*, pages 135–140, Washington, D.C., USA, 2000. ACM Press, New York, NY, USA.

[HFW99] L. Holmquist, J. Falk, and J. Wigström. Supporting Group Collaboration with Inter-Personal Awareness Devices. *Journal of Personal Technologies*, 3(1–2):13–21, 1999.

[HMR01] Abdelsalam (Sumi) Helal, Steven Edwin Moore, and Balaji Ramachandran. Drishti: An Integrated Navigation System for Visually Impaired and Disabled. In *5th IEEE International Symposium on Wearable Computers (ISWC '01)*, pages 149–156, Washington, D.C., USA, 2001. IEEE Computer Society.

[Hof02] Øystein Hoftun. MObile Nagging Geek Organizer. Technical report, Dep. of Computer and Information Science, NTNU, Norway, June 2002. Diploma thesis (311 p.).

[Hol95] D. Hollingsworth. The Workflow Management Coalition – The Workflow Reference Model. Technical report, Workflow Management Coalition, Lighthouse Point, Fla., January 1995. Technical Report WFMC-TC00-1003, version 1.1.

[HRH01] Jonathan Hammond, Rosamund Rawlings, and Anthony Hall. Will it work? In *Fifth IEEE International Symposium on Requirements Engineering (RE'01)*, pages 102–109, Toronto, Canada, 2001.

[HWF98] Lars Erik Holmquist, Joakim Wigstrom, and Jennica Falk. The Hummingbird: Mobile Support for Group Awareness. In *Demonstration at ACM Conference on Computer Supported Cooperative Work*. ACM Press, 1998.

[Hyg03] J. Hygen. Informasjonsteknologi for en bedre helsetjeneste (Si@!) – Information Technology for a better health care., 2003. (In Norwegian).

[IEE00] IEEE. *IEEE Std 1471 Recommended Practice for Architectural Description*, 2000.

[ISP00] ISP. Middleware - isp glossary definition and links. http://isp.webopedia.com/TERM/m/middleware.html, 2000.

[IWR02a] Ellen Isaacs, Alan Walendowski, and Dipti Ranganathan. Mobile Instant Messaging Through Hubbub. *Communications of the ACM*, 45(9):68–72, September 2002.

[IWR02b] Ellen Isaacs, Alan Walendowski, and Dipti Ranganthan. Hubbub: A sound-enhanced mobile instant messenger that supports awareness and opportunistic interactions. In *SIGCHI Conference on Human Factors in Computing Systems*, pages 179–186, Minneapolis, Minnesota, USA, 2002. ACM Press.

[JB04] Jens Bæk Jørgensen and Claus Bossen. Executable Use Cases: Requirements for a Pervasive Health Care System. *IEEE Software*, 21(2):34–41, March–April 2004.

[JBR99] Ivar Jacobson, Grady Booch, and James Rumbaugh. *The Unified Software Development Process*. Addison-Wesley Longman, Inc, 1999.

[JC92] Ivar Jacobson and Magnus Christensen. *Object-Oriented Software Engineering*. Addison-Wesley Longman, Inc, 1992.

[Jen01] Finn V. Jensen. *Bayesian Networks and Decision Graphs*. Springer Verlag, 2001.

[JHE99] Jin Jing, Abdelsalam Sumi Helal, and Ahmed Elmagarmid. Client-Server Computing in Mobile Environments. *ACM Computing Surveys (CSUR)*, 31(2):117–157, June 1999.

[JHTL04] Xiaodong Jiang, Jason I. Hong, Leila A. Takayama, and James A. Landay. Ubiquitous Computing for Firefighters: Field Studies and Prototypes of Large Displays for Incident Command. In *SIGCHI Conference on Human Factors in Computing Systems (CHI'04)*, pages 679–686, Vienna, Austria, 2004. ACM Press, New York, NY, USA.

[JK97] Sushil Jajodia and Larry Kerschberg. *Advanced Transaction Models and Architectures*. Kluwer Academic Publisher, 1997.

[Jon02] Willie D. Jones. Building Safer Cars. http://www.spectrum.ieee.org, 2002.

[Kam00] Peter J. Kammer. Supporting Dynamic Distributed Work Processes with a Component and Event Based Approach. In *22nd International Conference on Software Engineering (ICSE'00)*, pages 710–712, Limerick, Ireland, 2000. ACM Press.

[KBT+00] Peter J. Kammer, Gregory Alan Bolcer, Richard N. Taylor, Arthur S. Hitomi, and Mark Bergman. Techniques for Supporting Dynamic and Adaptive Workflow. *Computer Supported Cooperative Work*, 9(3/4):269–292, 2000.

[KIT01] KITH. IT-behov innen pleie- og omsorgstjenesten, 2001. (In Norwegian).

[Kle95] Leonard Kleinrock. Nomadic computingan opportunity. *SIGCOMM Comput. Commun. Rev.*, 25(1):36–40, 1995.

[Kle96] Leonard Kleinrock. Nomadicity: anytime, anywhere in a disconnected world. *Mobile Networks and Applications*, 1(4):351–357, 1996.

[KML04] Andrew L. Kun, W. Thomas Miller, and William H. Lenharth. Computers in Police Cruisers. *IEEE Pervasive Computing*, 3(4):34–41, October–December 2004.

[Kor02] Gerd Kortuem. Proem: A Middleware Platform for Mobile Peer-to-Peer Computing. *ACM SIGMOBILE Mobile Computing and Communications Review*, 6(4):62–64, 2002.

[Kru95] Philippe Kruchten. Architectural Blueprints – The '4+1' View Model of Software Architecture. *IEEE Software*, 12(6):42–50, November 1995.

[KTBB92] Simon M. Kaplan, William J. Tolone, Douglas P. Bogia, and Celsina Bignoli. Flexible, Active Support for Collaborative Work with ConversationBuilder. In *ACM Conference on Computer-Supported Cooperative Work (CSCW '92)*, pages 378–385, Toronto, Ontario, Canada, 1992. ACM Press, New York, NY, USA.

[La 02] Thomas F. La Porta. Introduction to the IEEE Transactions on Mobile Computing. *IEEE Transactions on Mobile Computing*, 1(1):2–9, January-March 2002.

[LAAA96] Sue Long, Dietmar Aust, Gregory Abowd, and Chris Atkeson. Cyberguide: Prototyping Context-Aware Mobile Applications. In *Conference Companion on Human Factors in Computing Systems*, pages 293–294, Vancouver, British Columbia, Canada, 1996. ACM Press.

[LC98] Chunnian Liu and Reidar Conradi. Process View of CSCW. In *Int'l Symposium of Future Software Technologies (ISFST'98)*, pages 46–51, Hangzhow, P.R. China, October 28-30th 1998.

[LDS00] Fredrik Ljungberg, Per Dahlberg, and Johan Sanneblad. Supporting opportunistic communication in mobile setting. In *ACM Conference on Human Factors in Computing Systems (CHI'2000)*, pages 111–112, The Hague, The Netherlands, April 1–6 2000.

[LEF+00] Mik Lamming, Marge Eldridge, Mike Flynn, Chris Jones, and David Pendlebury. Satchel: Providing Access to Any Document, Any Time, Anywhere. *ACM Transactions on Computer-Human Interaction*, 7(3):322–352, September 2000.

[LSF04] Matthias Lampe, Martin Strassner, and Elgar Fleisch. A ubiquitous computing environment for aircraft maintenance. In *ACM Symposium on Applied Computing (SAC '04)*, pages 1586–1592, Nicosia, Cyprus, 2004. ACM Press, New York, NY, USA.

[LSG96] Thomas F. La Porta, Krishan K. Sabnani, and Richard D. Gitlin. Challenges for Nomadic Computing: Mobility Management and Wireless Communications. *Mobile Networks and Applications*, 1(1):3–16, 1996.

[LY02] Kalle Lyytinen and Youngjin Yoo. Issues and Challenges in Ubiquitous Computing. *Communications of the ACM*, 45(12):62–65, December 2002.

[Mac99] Alessandro Maccari. The Challenges of Requirements Engineering in Mobile Telephones Industry. In *10th International Conference and Workshop on Database and Expert Systems Applications (DEXA '99)*. Springer Verlag, 1999. LNCS 1677.

[MC02] René Meier and Vinny Cahill. STEAM: Event-based Middleware for Wireless Ad Hoc Networks. In *International Workshop on Distributed Event-Based Systems (ICDSC/DEBS'02)*, pages 639–644, Vienna, Austria, 2002.

[MC03] René Meier and Vinny Cahill. Exploiting Proximity in Event-Based Middleware for Collaborative Mobile Applications. In *4th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS'03)*, pages 285–296, Paris, France, 2003. Springer Verlag, Heidelberg, Germany. LNCS 2893.

[MCE02] Cecilia Mascolo, Licia Capra, and Wolfgang Emmerich. Middleware for Mobile Computing (A Survey). In E. Gregori, G. Anastasi, and S. Basagni, editors, *Networking 2002 Tutorial Papers*, LNCS 2497, Los Angeles, California, United States, 2002. Springer Verlag.

[MCZE02] Cecilia Mascolo, Licia Capra, Stefanos Zachariadis, and Wolfgang Emmerich. XMIDDLE: A Data-Sharing Middleware for Mobile Computing. *Personal and Wireless Communications Journal*, 21(1), April 2002.

[MDW99] Dejan Milojicic, Frederick Douglis, and Richard Wheeler, editors. *Mobility; Processes, Computers and Agents*. ACM Press/Addison-Wesley, 1999.

[Mea00] Nancy R. Mead. Why is it so Difficult to Introduce Requirements Engineering Research Results into Mainstream Requirements Engineering Practice? In *Fourth International Conference on Requirement Engineering (ICRE 2000)*, pages 75–76. IEEE Computer Society, 2000.

[Moh94] C. Mohan. Tutorial: Advanced Transaction Models - Survey and Critique. In *ACM International Conference on Management of Data (SIGMOD'94)*, page 521, May 1994.

[MOW04] MOWAHS. MObile Work Across Heterogeneous Systems. Web: http://www.mowahs.com, 2004.

[Myn00] Elizabeth D. Mynatt. Co-opting Everyday Objects. In *Designing Augmented Reality Environments (DARE '00)*, pages 145–146, New York, NY, USA, 2000. ACM Press.

[MZ04] Marco Mamei and Franco Zambonelli. Programming Pervasive and Mobile Computing Applications with the TOTA Middleware. In *Second IEEE International Conference on Pervasive Computing and Communications (PerCom'04)*, pages 263–273, Orlando, Florida, March 14-17 2004. IEEE Computer Society Press.

[Nar96a] Bonnie A. Nardi, editor. *Context and Consciousness*. MIT Press, MIT, USA, 1996.

[Nar96b] Bonnie A. Nardi. *Studying Context: A Comparison of Activity Theory, Situated Action Models, and Distributed Cognition*, chapter 4, pages 69–102. MIT Press, 1996.

[NE00] Bashar Nuseibeh and Steve Easterbrook. Requirements Engineering: A Roadmap. In A. Finkelstein, editor, *[Fin00]*, pages 35–46, Limerick, Ireland, June 2000. ACM Press.

[New03] Automotive Intelligence News. Talking Cars For Less Congestion – The Future Of Telematics . http://www.autointell-news.com/News-2003/October-2003/October-2003-1/October-01-03-p6.htm, 2003.

[NFR02] Christian Navarro, Jesus Favela, and Marcela Rodriguez. Extending instant messaging to support spontaneous interactions in ad-hoc networks. In *Workshop on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments*, New Orleans, Louisiana, USA, November 16–20 2002.

[NPS03] Erica Newcomb, Toni Pashley, and John Stasko. Mobile Computing in the Retail Arena. In *SIGCHI Conference on Human Factors in Computing Systems (CHI'03)*, pages 337–344, Ft. Lauderdale, Florida, USA, 2003. ACM Press, New York, USA.

[Orl92] W.J. Orlikowski. Learning from Notes: Organizational Issues in Groupware Implementation. In *Conference on Computer-Supported Cooperative Work, CSCW'92*, pages 362–369, Toronto, Canada, 1992. ACM Press.

[Per02] Perl Mongers - The Perl advocacy people. Perl. http://www.perl.org, 2002.

[PMR00] Gian Pietro Picco, Amy L. Murphy, and Gruia-Catalin Roman. Developing Mobile Computing Applications with LIME. In *22nd International Conference on Software Engineering (ICSE'2000)*, pages 766–769, Limerick, Ireland, 2000. ACM Press.

[Pre94] J. Preece, editor. *Human-Computer Interaction*. Addison & Wesley, New York, 1994.

[pro00] CAGIS project. Cooperative agents in global information space webpage. http://www.idi.ntnu.no/∼cagis, 2000.

[PTA94] Colin Potts, Kenji Takahashi, and Annie I. Antón. Inquiry-Based Requirements Analysis. *IEEE Software*, 11(2):21–32, 1994.

[Rak99] Andry Rakotonirainy. Trends and Future of Mobile Computing. In *10th IEEE International Workshop on Database and Expert Systems Applications*, pages 136–, Florence, Italy, 1–3 September 1999. IEEE CS.

[RCRM02] Anand Ranganathan, Roy T. Campbell, Arathi Ravi, and Anupama Mahajan. ConChat: A Context-Aware Chat Program. *IEEE Pervasive Computing*, 1(3):51–57, July–September 2002.

[RHC+02] Manuel Román, Christoher Hess, Renato Cerqueira, Anand Ranganathan, Roy T. Campbell, and Klara Nahrstedt. A Middleware Infrastructure for Active Spaces. *IEEE Pervasive Computing*, 1(4):74–82, October–December 2002.

[Ril03] Gary Riley. CLIPS – A Tool for Building Expert Systems. http://www.ghg.net/clips/CLIPS.html, 2003.

[RN02] Heri Ramampiaro and Mads Nygård. CAGISTrans: Providing Adaptable Transactional Support for Cooperative Work – An Extended Treatment. *International Journal of Information Technology and Management*, 2002.

[RN04] Heri Ramampiaro and Mads Nygård. CAGISTrans: Providing adaptable transactional support for cooperative work - An Extended Treatment. *Information Technology & Management (ITM) Journal*, 5(1–2):23–64, 2004.

[Rob05] Robert Bosch GmbH. CAN Homepage of Robert Bosch GmbH. http://www.can.bosch.com, 2005.

[RPM00] Gruia-Catalin Roman, Gian Pietro Picco, and Amy L. Murphy. Software Engineering for Mobility: A Roadmap. In A. Finkelstein, editor, *[Fin00]*, pages 243–258, Limerick, Ireland, 2000. ACM Press.

[Rue96a] Craig Ruefenacht. The RUST Mail System. In *The 10th annual LISA Conference*, Chicago, Illinois, USA, September 29 – October 5 1996.

[Rue96b] Craig Ruefenacht. The RUST Mail System. http://www.eng.utah.edu/rust/docs/, 1996.

[RWS+03] Heri Ramampiaro, Alf Inge Wang, Carl-Fredrik Sørensen, Hien Nam Le, and Mads Nygård. Requirement Indicators Derived from a Mobile Characterisation Framework. In *IASTED International Conference on Applied Informatics (AI'2003)*, Innsbruck, Austria, 10-13 February 2003. Electronic Proceedings 8 pp.

[Sat93] Mahadev Satyanarayanan. Hot Topics: Mobile Computing. *Computer*, 26(9):81–82, September 1993.

[Sat96] M. Satyanarayanan. Fundamental Challenges in Mobile Computing. In *Fifteenth Annual ACM Symposium on Principles of Distributed Computing*, pages 1–7, Philadelphia, Pennsylvania, United States, May 23-26 1996. ACM Press.

[Sat02a] M. Satyanarayanan. A Catalyst for Mobile and Ubiquitous Computing. *IEEE Pervasive Computing*, 1(1):2–4, January–March 2002.

[Sat02b] M. Satyanarayanan. Challenges in Implementing a Context-Aware System. *IEEE Pervasive Computing*, 1(3):2–2, July–September 2002.

[Sat03] M. Satyanarayanan. Of Smart Dust and Brilliant Rocks. *IEEE Pervasive Computing*, 2(4):2–4, October–December 2003.

[SAW94] Bill N. Schilit, N. Adams, and R. Want. Context-Aware Computing Applications. In *IEEE Workshop on Mobile Computing Systems and Applications*, pages 85–90, Santa Cruz, CA, USA, December 1994.

[SBF+04] Thirunavukkarasu Sivaharan, Gordon Blair, Adrian Friday, Maomao Wu, Hector Duran-Limon, Paul Okanda, and Carl-Fredrik Sørensen. Cooperating Sentient Vehicles for Next Generation Automobiles. In *1st ACM International Workshop on Applications of Mobile Embedded Systems (WAMES 2004)*, Boston, MA, USA, June 6th 2004. ACM.

[Sch95] Bill N. Schilit. *A System Architecture for Context-Aware Mobile Computing*. PhD thesis, Colombia University, New York, NY, US, 1995.

[SD97] Carla Simone and Monica Divitini. Ariadne: Supporting Coordination through a Flexible Use of the Knowledge on Work Process. *Journal UCS (Electronic Journal)*, 3(8), 1997. Also as NTNU/SU-report 23/97 and at http://www.iicm.edu/jucs_3_8.

[Sea99] Carolyn B. Seaman. Qualitative Methods in Empirical Studies of Software Engineering. *IEEE Transactions on Software Engineering*, 25(4):557–572, July/August 1999.

[SGKK04] Martin Strohbach, Hans Gellersen, Gerd Kortuem, and Christian Kray. Cooperative Artefacts: Assessing Real World Situations with Embedded Technology. In Nigel Davies, Elizabeth Mynatt, and Itiro Siio, editors, *6th International Conference on Ubiquitous Computing (UbiComp'04)*, pages 250–267, Nottingham, England, September 7-10 2004. Springer-Verlag. LNCS 3205.

[Siv02] Thirunavukkarasu Sivaharan. Publish Subscribe Component-based Middleware for PDAs in Wireless Ad-hoc Network. Technical report, Lancaster University, 2002. MSc thesis.

[SK01] Henrik Stormer and Konstantin Knorr. PDA- and Agent-based Execution of Workflow Tasks. In *Agents in E-Business (AgEB'01) Workshop of the Informatik 2001*, pages 968–972, Vienna, Austria, September 2001.

[SNØS+03] Carl-Fredrik Sørensen, Bjørn Næss, Øyvind Strand, Alf Inge Wang, and Reidar Conradi. A Survey of Mobile Support Needs in the Home Nursing Care. *Journal of Korean Society of Medical Informatics (KOSMI)*, 9(2):390–395, October 20-22 2003. Proceedings of APAMI (Asia-Pacific Association of Medical Informatics) & CJKMI-KOSMI Conference - Ubiquitous Computing in Health Care: From wired to wireless.

[Sof05] Software Engineering Group at the Department of Computer and Information Science, NTNU. International Publications. http://www.idi.ntnu.no/grupper/su/index.php3?file=publ/INT-PUBL.php3, 2005.

[Som01] Ian Sommerville. *Software Engineering*. Addison-Wesley/Pearson Education Limited, 6th edition, 2001.

[SP93] Ian Sommerville and Manfred Paul, editors. *European Software Engineering Conference ESEC'93*, September 1993.

[ST94] B. Schilit and M. Theimer. Disseminating Active Map Information to Mobile Hosts. *IEEE Network*, 8(5):22–32, 1994.

[Sta02] Statistisk Sentralbyrå – Statistics Norway. KOSTRA – Municipality-State-Reporting. http://www.ssb.no/english/subjects/00/00/20/kostra_en/, 2002.

[Suc87] Lucy Suchman. *Plans and situated actions. The problem of human-machine communication*. Cambridge University Press, 1987.

[Sut98] Alistair Sutcliffe. Scenario-based requirements analysis. *Requirements Engineering*, 3(1):48–65, 1998.

[SVLZ01] A. Spriestersbach, H. Vogler, F. Lehmann, and T. Ziegert. Integrating Context Information into Enterprise Applications for the Mobile Workforce - A Case Study. In *First International Workshop on Mobile Commerce*, pages 55–59, Rome, Italy, 2001. ACM Press.

[SWC05] Carl-Fredrik Sørensen, Alf Inge Wang, and Reidar Conradi. Support of Smart Work Processes in Context Rich Environments. In John Krogstie, Karlheinz Kautz, and David Allen, editors, *IFIP TC8 Working Conference on Mobile Information Systems (MOBIS'2005)*, pages 15–30, Leeds, UK, December 6-7 2005. Springer, New York, USA. IFIP 191/2005.

[SWL+02] Carl-Fredrik Sørensen, Alf Inge Wang, Hien Nam Le, Heri Ramampiaro, Mads Nygård, and Reidar Conradi. The MOWAHS Characterisation Framework for Mobile Work. In *IASTED International Conference on Applied Informatics (AI'2002)*, pages 258–264, Innsbruck, Austria, February 18-22 2002.

[SWS+04] Carl-Fredrik Sørensen, Maomao Wu, Thirunavukkarasu Sivaharan, Gordon Blair, Paul Okanda, Adrian Friday, and Hector Duran Limon. A Context-Aware Middleware for Applications in Mobile Ad Hoc Environments. In *ACM/IFIP/USENIX Middleware'04 workshop: 2nd Workshop on Middleware for Pervasive and Ad-Hoc Computing (MPAC'04)*, pages 107–110, Toronto, Ontario, Canada, October 18-22 2004. ACM Press.

[SWØH03] Carl-Fredrik Sørensen, Alf Inge Wang, and Øystein Hoftun. Experience Paper: Migration of a Web-based System to a Mobile Work Environment. In *IASTED International Conference on Applied Informatics (AI'2003)*, pages 1033–1038, Innsbruck, Austria, February 10-13 2003.

[SWØHB03] Carl-Fredrik Sørensen, Alf Inge Wang, Øystein Hoftun, and Kenneth Aron Bjørkhaugen. An Evaluation of a Mobile Task Reporting System for Different Mobile Devices. In M.H. Hamza, editor, *IASTED International Conference on Software Engineering and Applications (SEA'2003)*, pages 751–756, Marina Del Rey, CA, USA, November 3-5 2003.

[Szy97] Clemens Szyperski. *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley, 1997.

[TH02] Aase Tveito and Per Hasvold. Requirements in the Medical Domain: Experiences and Prescriptions. *IEEE Software*, 19(6):66–69, November/December 2002.

[The02] The IEEE Working Group for WLAN Standards. IEEE 802.11 Wireless Local Area Networks. http://grouper.ieee.org/groups/802/11/, 2002.

[The05] The Workflow Management Coalition. Welcome to WFMC. http://www.wfmc.org, 2005.

[THSK01] Jonathan Trevor, David M. Hilbert, Bill N. Schilit, and Tzu Khiau Koh. From Desktop to Phonetop: A UI For Web Interaction On Very Small Devices. In *14th Annual ACM Symposium on User Interface Software and Technology*, pages 121–130, Orlando, Florida, 2001. ACM Press.

[Tia98] Pierre F. Tiako. Modelling the Federation of Process Sensitive Engineering Environments: Basic Concepts and Perspectives. In Volker Grühn, editor, *6th European Workshop on Software Process Technologies*, pages 132–136, Weybridge, UK, September 16-18 1998. Springer Verlag. LNCS 1487.

[TJSW02] Marco Torchiano, Letizia Jaccheri, Carl-Fredrik Sørensen, and Alf Inge Wang. COTS Products Characterization. In *14th International Conference on Software Engineering and Knowledge Engineering (SEKE'02)*, pages 335–338, Ischia, Italy, 2002. ACM Press.

[TKF95] William J. Tolone, Simon M. Kaplan, and Geraldine Fitzpatrick. Specifying Dynamic Support for Collaborative Work within WORLDS. In *Conference on Organizational Computing Systems (COCS '95)*, pages 55–65, Milpitas, California, United States, 1995. ACM Press, New York, NY, USA.

[Ubi03] UbiCore. Universally Interoperable Core. http://www.ubi-core.com, Last accessed August 5th 2003.

[Uni01] Universidade de Lisboa, Lancaster University, Trinity College Dublin and Universität Ulm. CORTEX project homepage. http://cortex.di.fc.ul.pt, 2001.

[van01] W. M. P. van der Aalst. How to handle dynamic change and capture management information? An approach based on generic workflow models. *Computer Systems Science & Engineering*, 16(5), September 2001.

[VR02] Thomas Vantroys and José Rouillard. Workflow and Mobile Devices in Open Distance Learning. In *IEEE International Conference on Advanced Learning Technology (ICALT 2002)*, pages 123–127, Kazan, Tatarstan, Russia, September 9-12 2002. IEEE.

[VS00] Steffen Volz and Monika Sester. Positioning and Data Management Concepts for Location Aware Applications. In *2nd International Symposium on Telegeoprocessing*, pages 171–184, Nice-Sophia-Antipolis, France, 2000.

[vVS+03] K. van Laerhoven, N. Villar, A. Schmidt, G. Kortuem, and H.-W. Gellersen. Using an Autonomous Cube for Basic Navigation and Input. In *ICMI/PUI 2003*, pages 203–211. ACM Press, 2003.

[Vyg78] L. S. Vygotskij. *Mind and Society*. Harvard University Press, Cambrigde, MA, USA, 1978.

[Wan01] Alf Inge Wang. *Using a Mobile, Agent-based Environment to support Cooperative Software Processes*. PhD thesis, Norwegian University of Science and Technology, Department of Computer and Information Science, NTNU, Trondheim, Norway, February 5th 2001.

[Wap02] WapForum. Wireless Application Protocol Architecture Specification. http://www.wapforum.org, 2002.

[Wei93] Mark Weiser. Hot Topics: Ubiquitous Computing. *Computer*, 26(3):71–73, March 1993.

[Wei02] Mark Weiser. The Computer for the 21st Century. *IEEE Pervasive Computing*, 1(1):18–25, January–March 2002. Reprinted from Scientific American, 1991.

[WFB+04] Maomao Wu, Adrian Friday, Gordon Blair, Thirunavukkarasu Sivaharan, Paul Okanda, Hector Duran Limon, Carl-Fredrik Sørensen, Gregory Biegel, and René Meier. Novel Component Middleware for Building Dependable Sentient Computing Applications. In *ECOOP'04 Workshop: Component-Oriented Approaches to Context-Aware Systems*, Oslo, Norway, June 14th 2004.

[Wir05a] Wireless World Initiative (WWRF). Wireless World Initiative (WWRF). http://www.wireless-world-initiative.org/, 2005.

[Wir05b] Wireless World Research Forum (WWRF). Wireless World Research Forum (WWRF). http://www.wireless-world-research.org/, 2005.

[WL01a] Alf Inge Wang and Chunnian Liu. Process Support for Mobile Work Across Heterogenous Systems. In *[Amb01]*, 2001.

[WL01b] Mikael Wiberg and Fredrik Ljungberg. *Exploring the Vision of "Anytime, Anywhere" in the Context of Mobile Work*, pages 153–165. Idea Group Publishing, Hershey, PA, USA, 2001.

[Wor02] World Wide Web Consortium – W3C. Extensible Markup Language (XML). http://www.w3.org/XML/, 2002.

[WR96] C. Donald Wilcox and Gruia-Catalin Roman. Reasoning About Places, Times, and Actions in the Presence of Mobility. *IEEE Transactions on Software Engineering*, 22(4):225–247, April 1996.

[WS03] Alf Inge Wang and Carl-Fredrik Sørensen. A Comparison of Two Different Java Technologies to Implement a Mobile Agent System. In *IASTED International Conference on Applied Informatics (AI'2003)*, pages 1039–1044, Innsbruck, Austria, February 10-13 2003.

[WS05] Carl-Henrik Wolf and Michael Sars. Peer2ME. Web: http://www.peer2me.org, 2005.

[WSB+05] Alf Inge Wang, Carl-Fredrik Sørensen, Steinar Brede, Hege Servold, and Sigurd Gimre. The Nidaros Framework for Development of Location-aware Applications. In John Krogstie, Karlheinz Kautz, and David Allen, editors, *IFIP TC8 Working Conference on Mobile Information Systems (MOBIS'2005)*, pages 171–185, Leeds, UK, December 6-7 2005. Springer, New York, USA. IFIP 191/2005.

[WSC04] Alf Inge Wang, Carl-Fredrik Sørensen, and Reidar Conradi. A Three Level Framework for Process Support: The MOWAHS Approach. In Hamid R. Arabnia and Hassan Reza, editors, *International Conference on Software*

Engineering Research and Practice – Session: Team-based Software Engineering (TBSE'04), volume 1, pages 10–16, Las Vegas, Nevada, USA, June 21-24 2004. CSREA Press.

[WSF05] Alf Inge Wang, Carl-Fredrik Sørensen, and Thomas Fossum. Mobile Peer-to-Peer Technology used to Promote Spontaneous Collaboration. In *2005 International Symposium on Collaborative Technologies and Systems (CTS 2005)*, pages 48–55, St. Louis, Missouri, USA, May 15-20 2005. IEEE Computer Science. In conjunction with ICSE'05.

[WSFS03] Alf Inge Wang, Carl-Fredrik Sørensen, Thale Christina Fritzner, and Eldrid Schei. Case Study: Use of the Mobile Tool HandyMan for Mobile Work. In M.H. Hamza, editor, *IASTED International Conference on Software Engineering and Applications (SEA'2003)*, pages 757–762, Marina Del Rey, CA, USA, November 3-5 2003.

[WSHB02] Alf Inge Wang, Carl-Fredrik Sørensen, Øystein Hoftun, and Kenneth Aron Bjørkhaugen. An Evaluation of a Mobile Task Reporting System for Different Mobile Devices. Technical report, Dept. of Computer and Information Science, NTNU, Norway, April 2002.

[WSI03] Alf Inge Wang, Carl-Fredrik Sørensen, and Eva Indal. A Mobile Agent Architecture for Heterogeneous Devices. In *3rd IASTED International Conference on Wireless and Optical Communications (WOC'2003)*, pages 584–590, Banff, Canada, July 14-16 2003.

[WSR+05] Alf Inge Wang, Carl-Fredrik Sørensen, Heri Ramampiaro, Hien Nam Le, Reidar Conradi, and Mads Nygård. Using the MOWAHS Characterisation Framework for Development of Mobile Work Applications. In *[BKS05]*, pages 128–142, Oulu, Finland, June 13-18 2005. Springer Verlag. LNCS 3547.

[WWSL02] Lihui Wang, Brian Wong, Weiming Shen, and Sherman Lang. A Java 3D-Enabled Cyber Workspace. *Communications of the ACM*, 45(11):45–49, November 2002.

[Zim99] James Bryan Zimmerman. Mobile Computing: Characteristics, Business Benefits, and the Mobile Framework. Technical Report INSS 690 CC, University of Maryland European Division, April 2nd 1999.

[ZJWF02] Matthew J. Zieniewicz, Douglas C. Johnson, Douglas C. Wong, and John D. Flatt. The Evolution of Army Wearable Computers. *IEEE Pervasive Computing*, 1(4):30–40, October–December 2002.

[ZW98] Marvin V. Zelkowitz and Dolores R. Wallace. Experimental Models for Validating Technology. *Computer*, 31(5):23–31, May 1998.