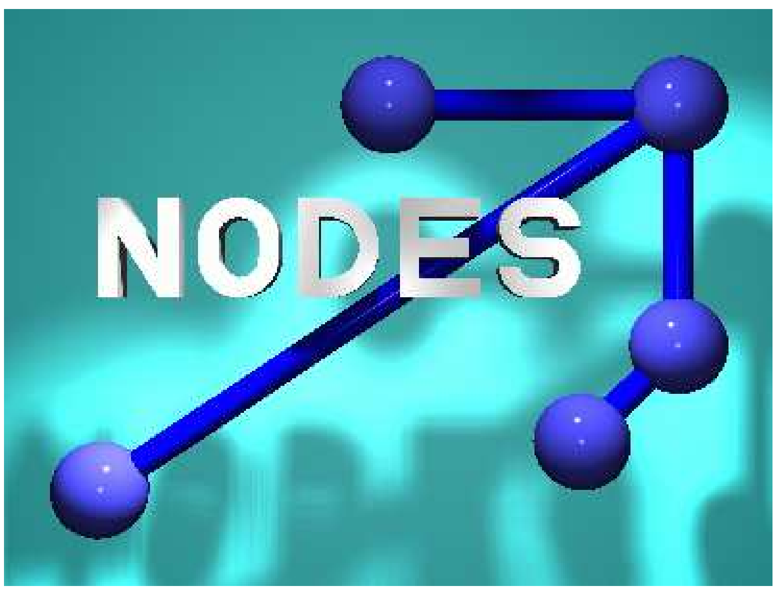
DEPARTMENT OF COMPUTER SCIENCE
SERIES OF PUBLICATIONS B
REPORT B-2005-1

The First Ten Years of the NODES Group

Kimmo Raatikainen, editor



UNIVERSITY OF HELSINKI
FINLAND

Contact information

Postal address:
Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2 b)
FIN-00014 University of Helsinki
Finland

Email address: postmaster@cs.Helsinki.FI (Internet)

URL: http://www.cs.Helsinki.FI/

Telephone: +358 9 1911

Telefax: +358 9 191 51120

ISSN 1458-4786
ISBN 952-10-2278-7 (paperback)
ISBN 952-10-2279-5 (PDF)
Computing Reviews (1998) Classification: C.2.1, C.2.4, C.2.5, C.3, D.2.4, D.4.7, H.2.4
Helsinki 2005
Helsinki University Printing House

The First Ten Years of the NODES Group

Kimmo Raatikainen, editor

Department of Computer Science
P.O. Box 68, FIN-00014 University of Helsinki, Finland
Kimmo.Raatikainen@cs.helsinki.fi
http://www.cs.helsinki.fi/kimmo.raatikainen/

Series of Publications B, Report B-2005-1
Helsinki, January 2005, 87 pages
ISSN 1458-4786
ISBN 952-10-2278-7 (paperback)
ISBN 952-10-2279-5 (PDF)

Abstract

The NODES Group was launched at the Helsinki University Computer Science Department on Friday, January 27, 1995. The objective was to stimulate and coordinate research on distributed and networked systems and applications. Now, ten years later, we can proudly claim success in excellent education and research. This jubilee book takes a retrospective look at our early research projects, their problem statenebts, achievements, and follow-ups. We also outline the research challenges for the next ten years.

Our background is in the KOPS, which was an abbreviation of *Käyttöjärjestelmät—OPerativ System*, that is, Operating Systems in Finnish and Swedish. We evolved from operating systems and performance analysis research to distributed and concurrent systems. Later we also got envolved with mobile computing and communication.

In the very beginning NODES research topics were: **distributed systems** and **telecommunication software**. Today we have five topics: **Wireless Internet** (*Communication over wireless (all protocol layers)* and *Mobile middleware*), **Open Distributed and Collaborative Systems** (*Co-operation of autonomous systems*), **Formal Methods** (*Protocol verification*), **Trust, Privacy, Security**, **Linux Developments** (*Timeliness and high availability in Linux* and *Linux–RTOS interworking*).

The research projects covered in this restrospective include AHTO, Mowgli, MoCo,

Dryad, and Darfin. *AHTO* was our first research project on distributed processing. The project ran years before the NODES Group, but it was our prolog to the whole research area of distributed systems and data communications. The *Mowgli* project was our entry to the mobile and wireless world. Here we concentrate on the background of Mowgli, since our mobile research is covered in the Mowgli Jubilee Book *"Wireless Internet: Challenges and Solutions"*, Report B-2004-3, Helsinki University Computer Science Department. *MoCo* covers our research on modeling concurrency. The *Dryad* project was an immediate follow-up of AHTO. It addressed trading and ISO's ODP reference model. The *Darfin* project was our entry to telecommunications software. Its original focus was on real-time database systems for telecommunications.

Computing Reviews (1998) Categories and Subject Descriptors:

C.2.1 Computer Systems Organization [Computer-Communication Networks]: Network Architecture and Design
C.2.4 Computer Systems Organization [Computer-Communication Networks]: Distributed Systems
C.2.5 Computer Systems Organization [Computer-Communication Networks]: Local and Wide-Area Networks
C.3 Computer Systems Organization: Special-Purpose and Application-Based Systems
D.2.4 Software [Software Engineering]: Software/Program Verification
D.4.7 Software [Operating Systems]: Organization and Design
H.2.4 Information Systems [Database Management]: Systems

General Terms:

Design, Experimentation, Measurement, Performance, Reliability, Standardization, Verification

Additional Key Words and Phrases:

Distributed systems, Mobile computing and communication, Internet protocols, Formal specification and verification, Real-time databases

Acknowledgements

This book presents the collective work of the *NODES Group* at the Helsinki University Computer Science Department. The current core team includes Timo Alanko, Auvo Häkkinen, Timo Karvi, Teemu Kerola, Markku Kojo, Lea Kutvonen, Päivi Kuuppelomäki, Matti Luukkainen, Jukka Manner, Liisa Marttinen, Tiina Niklander, and Professor Kimmo Raatikainen (leader). In the early days, the group was supervised by Professor Martti Tienari, now Professor Emeritus.

During the ten years we have had the priviledge to work with dozens of excellent graduate and post-graduate students. The list is too long for this section, but lists of Ph. D., Ph. Lic., and selected M. Sc. theses in Chapter A should be regarded as such. In addition, the authors want to acknowledge the current active Ph.D. and M.Sc. students in our projects: Davide Astuti, Suresh Chande, Laila Daniel, Jyrki Haajanen, Juha-Pekka Haataja, Lauri Hyttinen, Jin Jun, Ilpo Järvinen, Jaakko Kangasharju, Mika Karlstedt, Jouni Korhonen, Oskari Koskimies, Pradip Lamsal, Mikko Laukkanen, Simone Leggio, Heikki Lindholm, Janne Metso, Markus Miettinen, Aki Nyrhinen, Ilja Ponka, Petteri Pöyhönen, Oriana Riva, Sini Ruohomaa, Toni Ruokolainen, Marko Saaresto, Pasi Sarolahti, Sasu Tarkoma, Lea Viljanen, and Taneli Vähäkangas.

External funding has been crucial for our research. The National Technology Agency of Finland (TEKES) has funded us by millions of euros during the ten years. We hope that they feel that the money was well spent. We have also got public funding from European Commission, European Space Agency, Ministry of Education, and Academy of Finland. Nokia and TeliaSonera Finland (former Sonera and Telecom Finland) have been our most important industrial partners. Other industrial partners include Alcatel Space, Digital Equipment Corp. (DEC), Elisa (former Helsinki Telephone Company), Ericsson Finland, Finnet Group, the Finnish State Computer Centre (VTKK), ICL, Nixu, Solid, SSH, Stonesoft, SysOpen, Systems Software Partners, Technical Research Center of Finland (VTT), and Tellabs.

Contents

1 Introduction by Kimmo Raatikainen **1**

2 The Prolog: AHTO by Timo Alanko **7**
2.1 The network environment in the late 1980s 7
2.2 New challenges 8
2.3 AHTO Goals 8
2.4 Basic Concepts of Distributed Computing 9
2.5 The AHTO system 11
2.6 The project in a nutshell 11
2.7 Spring-offs 11

3 The Dawn of Mowgli by Timo Alanko, Markku Kojo and Heimo Laamanen **13**
3.1 The Mobile Environment in the Early 90s 13
3.2 How it started—Enterprise Viewpoint 14
3.3 Are there problems? 17
3.4 Specifying goals 18
3.5 The Mowgli approach 19
3.6 Proof of concept 22
3.7 Some afterthoughts 23
3.8 The years after 24
3.9 Mowgli publications 25

4 MoCo Research Field by Martti Tienari **29**

5 DRYAD and Beyond by Lea Kutvonen **33**
5.1 Development of open distributed computing environments and application interoperability 34
5.2 New challenges 37
5.3 The DRYAD trading service 38

5.4 The Pilarcos enhanced trading services and eCommunity establishment . . . 42
5.5 The web-Pilarcos B2B middleware . . . 43
5.6 Conclusion . . . 44
5.7 Publications of the group . . . 45

6 Databases in Telecommunications by Kimmo Raatikainen 51
6.1 Research Premises . . . 52
6.2 Darfin Objectives . . . 53
6.3 Rodain Objectives . . . 53
6.4 Darfin Results and Rodain Directions . . . 53
6.4.1 High-Level Information Model . . . 54
6.4.2 Real-Time Transactions . . . 55
6.4.3 Object Models . . . 56
6.4.4 Correctness Criteria for Real-Time Transactions . . . 57
6.5 Perspective . . . 58
6.6 Publications . . . 58

7 Our First Steps in European Co-operation by Kimmo Raatikainen 63

8 The Next Ten Years by Kimmo Raatikainen and Lea Kutvonen 67
8.1 Nodes 1995–2004 . . . 67
8.2 Reconsiderations . . . 69
8.3 New premises . . . 71
8.4 Environmental expectations . . . 72
8.5 Research challenges . . . 74
8.6 Postscript . . . 78

A NODES Theses 1995–2004 81
A.1 Ph.D. Theses . . . 81
A.2 Ph.Lic. Theses . . . 82
A.3 Selection of M.Sc. Theses . . . 82

About the Authors

Timo Alanko has the M.Sc. degree in mathematics (1964) and the Ph.D. degree in computer science (1983), both from the University of Helsinki. From 1964 to 1967 he worked at Nokia Electronics, Helsinki. From 1968 he has been employed by the University of Helsinki in the Department of Computer Science as a research and teaching assistant and as a lecturer; since 1983 acting in various professorships. The academic year 1987–1988 he spent as a visiting professor at the University of Tampere. At the department, his main responsibilities in research and education have been in the areas of operating systems, distributed systems and data communication, and performance analysis. He has been the project leader or a senior researcher in several national research projects in the area of distributed systems and mobile wireless computing. He has supervised three Ph.D. theses and about 200 M.Sc. thesis. From 1999 to 2001 he was the head of the Computer Science Department and 2001–2003 the leader of the Distributed Systems and Data Communications section at the department. In the years 1992–1997 he was a member of faculty council of the Faculty of Science. In 1987–1988 he was a member of the executive committee of the Finnish Information Processing Association, in 1997–1999 a member of board of the Finnish Union of University Lecturers, and in 2001–2002 a member of board of the International Association ”Top Universities e-Learning International Program” (Brussels). He is member of the Finnish Society of Computer Science, ACM, and IEEE/Computer Society. In 2002 he received the degree of Dr. hc. from the University of Petrozavodsk.

Markku Kojo has M.Sc. degree in Computer Science (University of Helsinki). He was a system programmer and system analyst at Finnish PTT, 1983–1986. From 1986 he has been employed by the University of Helsinki in the Department of Computer Science as a teaching assistant, as a system manager, as a researcher, and as a lecturer. In 1994–1995 he worked as a researcher in the national research project Mowgli being the principal designer of the Mowgli mobile computing architecture. In 1997 he worked for Sonera ltd. as a network system consultant. From 1999 he has been working as a project technical leader and a project manager in several national research projects on mobile computing and communications. In 2000–2002 he joined EC projects BRAIN (IST-1999-10050) and MIND

(IST-2000-28584) as a senior researcher. In MIND he worked as a Workpackage and Activity leader. From Fall 2001 to the early 2004 he worked as a project technical leader of a partner team in ESA project TRANSAT. His research interests include Mobile Computing, Wireless Networking, Distributed systems, and Computer Networks. He is the author of a number of scientific publications in the these areas. He has made several contributions to IETF. He is member of the Finnish Society of Computer Science, ACM, ISOC, and ISOC Finland.

Lea Kutvonen holds PhD degree in computer science from University of Helsinki. She was first employed by the University of Helsinki Department of Computer Science in 1985, and has since worked in various positions ranging from senior programmer to chief systems analysts, and from teaching assistant to acting professor. Her research interests with leading role in some national research projects include distributed computing platforms, open distributed processing, B2B collaboration, interoperability, and distributed systems architectures. She has been the advisor of around 50 MSc theses; currently she supervises 6 PhD students. Her current leasure-time activities include chairing of the Finnish Society of Computer Science.

Heimo Laamanen has M.Sc. (1982) and Ph.L. (2000) degrees in computer science (University of Helsinki). He had several different positions, e.g. Senior Support Engineer and Research Engineer, at Digital Equipment Corporation (1982–1996). Since 1996 he has been employed by TeliaSonera as a principal researcher. He has lead the research activities in the field of intelligent wireless Internet. He has several publications in these areas. He has also acted as a lecturer at the University of Helsinki teaching mobile computing and software agents technologies since 1996. He has supervised several M.Sc. thesis. He has also participated in a leading role European projects, e.g. CRUMPET (IST-1999-20147), WISE (IST-2000-30028), and CASCOM (IST-2004-511632). He has also participated national research projects (funded by the National Technology Agency of Finland—TEKES—and industry) on mobile computing, wireless communication, and middleware for mobile computing. Principal Researcher Heimo Laamanen has been active in standardization work, for example, by being a chairman of technical committees and a member of FIPA Board of Directors (2000–2002). His current research interests include wireless data communication, semantic web technology, and context-awareness.

Kimmo Raatikainen has M.Sc. (1983) and Ph.D. (1990) degrees in computer science (University of Helsinki). He was a system manager at the Finnish State Computer Centre, Division of University Support, 1981–1985. From 1986 he has been employed by the Helsinki University Computer Science Department as a research and teaching assistant, as Assistant, Associated and Full Professor (from 1998). He is the leader of Distributed Systems and Data Communications section at the department. From January 2000 Professor Raatikainen has been part-time Principal Scientist (and Research Fellow from April 2004) in Nokia Research Center and from January 2002 also part-time Principal Scientist (leading the Mobile Computing research area) in the Helsinki Institute for Information Technology (HIIT). From August 2004 he has been the head of Helsinki Graduate School in Computer Science and Engineering (HeCSE).

Professor Raatikainen has had a leading role in several European projects including DOLMEN (AC036), MONTAGE (AC325), PRIME (AC370), HPGIN (ESPRIT 29737), BRAIN (IST-1999-10050), CRUMPET (IST-1999-20147), MIND (IST-2000-28584). He has also led several national research projects (funded by the National Technology Agency of Finland—TEKES—and industry) on mobile computing, wireless communication, middleware for mobile computing and on telecommunications software architectures. His current research interests include wireless communication, middleware for mobile distributed systems, and operating systems. He has over 100 scientific publications in these areas

Professor Raatikainen has supervised 7 Ph.D. theses and c. 90 M.Sc theses. Currently 16 Ph.D. students are actively conducting their theses under his supervision.

Martti Tienari (born 1935) wrote his Ph.D thesis in Mathematics 1962, worked in the start-up team of Nokia Electronics (at that time a division of the Finnish Cable Work) 1960–1967, professor of computer science and the chairman of the computer science department at the University of Helsinki 1967–1998, retirement at the end of the year 1998 from active duty, working still as professor emeritus at the department. His working areas have changed during the years: At Nokia in 1960ies as problem analyst, programmer and computer center director, in 1967–1972 analysis of roundoff errors in numerical algorithms, 1973–1983 programming languages and compiler (especially theory-based metacompiling), 1983– communication protocols, distributed systems and modelling of concurrency. He has supervised 19 Ph.D theses in computer science, his CV lists 58 publications.

Chapter 1

Introduction

Kimmo Raatikainen

The NODES Group was formed at the Helsinki University Computer Science Department to stimulate and coordinate research on distributed and networked systems and applications. The official launching date was January 27, 1995, but the oldest document found is a memo dated November 22, 1994. Timo Alanko, Lea and Petri Kutvonen, Liisa Marttinen and Kimmo Raatikainen were planning the group. The second meeting that could be traced was on November 30, 1994. There was also a public gathering on December 13, the original call (in Finnish) is shown in Figure 1.1.

The group's scope of interests was stated to include concurrency, telecommunication software, and distributed systems. At that time the focus was on the design and development of infrastructure for mobile computing, on performance evaluation of distributed and networked systems, and on ODP (Open Distributed Processing) Reference Model.

One major problem in establishing a research group is the selection of a name. We had many long discussions before the name of *NODES*—Petri Kutvonen's proposal—was introduced to Professor Martti Tienari who finally accepted the name. During the discussions, at least the following names were proposed:

- TeleSystems,
- TeleData,
- DiNeX (DIstribution - NEtworks - eXperimention),
- DANES (Distributed And NEtworked Systems),
- FEDS (Formal and Experimental Distributed Systems),

```
         << xxxxx  xxxxxx >>
        <<    xx     xx    >>
       <<      xx   xx      >>
      <<        xx xx        >>
THE  <<          xxx          >>  GROUP
      <<         xxx         >>
       <<       xx xx       >>
        <<     xx   xx     >>
         <<  xxxxx xxxxx  >>


Olemme perustamassa laitoksellemme hajautettujen ja
verkotettujen järjestelmien ja sovellusten kokeelliseen
tutkimukseen erikoistunutta tutkimusryhmää. Toistaiseksi
ryhmä kulkee työnimellä 'The <X> Group'.

Open <X> on kaikille asiasta kiinnostuneille tarkoitettu
ideointi- ja keskutelutilaisuus tiistaina 13.12. klo 14:30->
salissa B450.
__________________________________________________________________________

Tässä tulee joitakin alustavia kaavailuja keskustelun pohjaksi:

Ryhmän tavoitteena on stimuloida ja koordinoida laitoksella
tehtävää em. alueella tapahtuvaa tutkimusta ja opetusta.
Ryhmän keskeisiä tutkimusaiheita ovat liikkuvuus, suorituskyky ja
arkkitehtuurit. Konkreettinen tutkimustyö tapahtuu projekteissa ja
erillisissä tutkimushankkeissa. Järjestäytymällä tutkimusryhmäksi pyritään
tutkimustoiminnan systematisointiin ja näkyvyyden lisääntymiseen.


Toiminta keväällä 1995:

Kiinteä tapaamisaika: joka toinen perjantai klo 13-15.

<X>-Forum: Ryhmään assosioituneiden tutkijoiden ja opiskelijoiden
keskustelutilaisuus
20.1.: Tarvittavan opetuksen ideointi
17.2.: Tutkimuksen suuntaviivat

Kollokvio: Ryhmään assosioituneiden projektien esittelyjä
 3.2.: Mobile National Host
 3.3.: Dryad
31.3.: Mowgli
28.4.: Darfin

Open <X>: Avajaiskokkarit 27.1.


<X> ryhmän puolesta

Timo Alanko ja Kimmo Raatikainen
```

Figure 1.1: Email call to the first NODES public gathering

- PaD-FEDS (Parallel and Distributed - Formal and Exprerimental Development of Systems),
- CoDE-F (Concurrency and Distribution Experimentally and Formally),
- DANCE (Distributed Applications in Networked Computing Environments),
- GLADE (Group ... appLications and Architectures for Distributed Environments),
- AIDE (Applications In Distributed Environments).

Actually it took five years before the NODES Group was "officially" recognized in the Annual Report 2000[1]. Before that the NODES Group was included in Computer Software Research as extracted from old reports of department activities.

1994-5[2]: Our computer software research can be subdivided into two main areas: distributed systems and telecommunication software (Raatikainen, Tienari) and programming languages and software engineering (Paakki).

1996-7[3]: Our computer software research can be subdivided into two main areas: distributed systems and telecommunication software (Raatikainen, Tienari) and programming languages and software engineering (Paakki). In some projects these two areas are intertwined.

The Annual Report 1998[4] recognized *Distributed Systems and Data Communication* as a research area but did not mention the NODES Group:

> *Distributed Systems and Data Communication (Professor Kimmo Raatikainen, Professor Timo Alanko, Professor Martti Tienari): mobile computing, formal specification and verification, distributed systems, computer networks, operating systems.*

Finally, in the Annual Report 2000 the NODES Group officially appears:

> *The NODES Group has been formed to stimulate and coordinate research on distributed and networked systems and applications. The group's scope of interests includes concurrency, telecommunication software, and distributed systems.*

[1]Report B-2001-2, Helsinki University Computer Science Department.
[2]Report A-1996-3, Helsinki University Computer Science Department.
[3]Report A-1998-1, Helsinki University Computer Science Department.
[4]Report A-1999-3, Helsinki University Computer Science Department.

> *The current focus is on the design and development of infrastructure for mobile computing, on performance evaluation of distributed and networked systems, and on open distributed software architectures and services.*

At that time we had four research topics, each with some subtopics:

Nomadic Computing: Communication over wireless and Adaptation to tempo-spatial changes,

Distributed Software Systems: Co-operation of autonomous system and CORBA performance,

Formal Methods: Protocol verification, and

Operating System Enhancements: High-speed network device drivers and Real-time database systems.

Today we have five topics:

Wireless Internet: Communication over wireless (all protocol layers) and Mobile middleware,

Open Distributed and Collaborative Systems: Co-operation of autonomous systems,

Formal Methods: Protocol verification,

Trust, Privacy, Security

Linux Developments: Timeliness and high availability in Linux and Linux-RTOS interworking.

The most notable change during the last five years has been the inclusion of the theme *Trust–Privacy–Security*. Inside the four "old" themes, however, there have been changes in the content as the following chapters will indicate.

Each research group should evolve over time. Therefore, our next step is to reconsider our mission. The current mission slide is shown in Figure 1.2 The slogan *"Any technology distinguishable from magic is insufficiently advanced"* is still valid. As long as computing and communication is not ubiquitous, that is invisible, we have someting to improve. The same is perhaps also true with the high-level research focus of *"how systems can be divided into independently working parallel parts and how these parts communicate with each other."* However, we need to elaborate the research topics of *functionality in the basic components*, *the protocols between the parts*, and *performance evaluation* so that our mission could guide our future research projects.

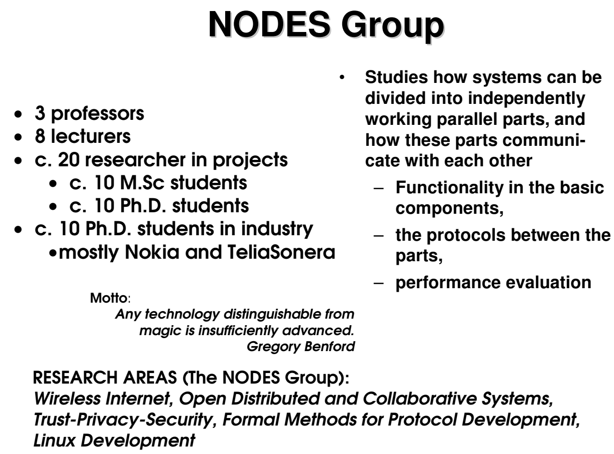


Figure 1.2: NODES Mission Slide

Nevertheless, in Spring 1995 we arranged a colloquium introducing the projects in the NODES Group:

Feb 3 Håkan Mitts (VTT): Mobile National Host

Feb 17 Timo Alanko: Mobile Computing (Mowgli)

Mar 3 Martti Tienari: Process equivalences in analysis of concurrent systems

Mar 17 Lea Kutvonen: Trading in Open Systems (Dryad)

Mar 31 Kimmo Raatikainen: Real-Time Databases in Telecommunications (Darfin)

Apr 21 Juha Puustjärvi: Advanced Transaction Models (Transcoop)

May 5 Jukka Paakki: Protocol Implementation Language KANNEL

In this book we take a retrospective look at the founding projects: their problem statements, outcomes, and follow-ups. The rest of the book contains the following arcticles:

- *The Prolog: AHTO* by Timo Alanko,
- *The Dawn of Mowgli* by Timo Alanko, Markku Kojo and Heimo Laamanen,
- *MOCO: Modeling of Concurrency as a Research Field* by Martti Tienari,

- *DRYAD and Beyond* by Lea Kutvonen,
- *DARFIN: Database Architecture for Intelligent Networks* by Kimmo Raatikainen,
- *Our First Steps in European Co-operation* by Kimmo Raatikainen, and
- *The Next Ten Years* by Kimmo Raatikainen and Lea Kutvonen.

Chapter 2

The Prolog: AHTO

Timo Alanko

The department started research in the area of open distributed processing at the end of the 1980s. In our first project (AHTO, *Avoimen Hajautetun Tietojenkäsittelyn Ohjelmistoteknologia*, Open Distributed Processing Technology; 1988-1990) we designed and implemented, as middleware, a distributed software platform, which offered a homogeneous interface to computing services available in a heterogeneous computer network.

2.1 The network environment in the late 1980s

In the late 80s the world was different. The Internet existed, but it was only used within the academic community. However, the availabe services were not too many, in practice they were limited to e-mail, file transfer, and remote login. As far as programming was concerned, intercomputer communication could be based on concepts such as pipes and message passing. At the system level, major publicity was acquired by Xerox PARC's Grapevine[1], a system which implemented a distributed name service (to support message passing) and which used remote procedure calls for communication.

Within the enterprise community peer-to-peer communication was based on plain old telephony, fax and mail.

[1]Schroeder, M. D.; Andrew D. Birrell, A. D. and Needham, R. M.: Experience with Grapevine: the growth of a distributed system. ACM Transactions on Computer Systems 2, 1 (Feb. 1984).

2.2 New challenges

At the time the information processing in a typical organization had already started to diversify: the traditional mainframe-based computer system was expanding to a collection of computers, offering different applications in different offices on different operating systems. This development led to new data-processing needs inside the organization: information was expected to flow seamlessly through a more and more colorful garden of computing units.

The information processing of an enterprise developed gradually to a distributed system, but this evolution did actually not take place through a distribution of the centralized processing, as the wording suggests, but through an integration of formerly independent applications. The challenge: how to make alien actors interoperate, how to combine isolated systems into integrated services in an enterprise-wide information processig system.

At the end of 1980s several research projects were active in the area. There also existed some few industry-oriented approaches, but these did not succeed in creating too much enthusiasm among the users.

System-level standardization was coming under consideration. Both ISO and CCITT (the later ITU) were starting work groups to develop a conceptual framework for open distributed processing. In industry, the Object Management Group was not yet further than in the starting phase of what was to become, in the early 1990s, the CORBA specifications.

Middleware, as a concept, had not yet emerged.

2.3 AHTO Goals

The general idea in AHTO was to develop both a conceptual framework and appropriate tools to help in specifying, implementing and using distributed applications. We had various groups of interested partners in mind: end users, system designers, and programmers.

The concrete goal was to devise something which was later to be called a middleware layer:

- It was to provide a homogeneous interface to all available services (for end users, for upper-layer applications).

- The local services, as legacy software, should be usable as they were, through the local operating system, with no changes required at all.

- It should contain a control interface for specifying computations which included distributed, concurrent tasks.

- Instances of AHTO, implemented on different operating systems, were to cooperate in a seamless way.

In practice AHTO was to offer communication operations for remote retrieval of information and for remote interprocess communication. In order to access remote services and remote information, a global naming or directory service was needed. As the computers were different, presentation transformations were needed. In addition, the user should not be harrassed by platform-dependent practical things such as the invocation of a local process to execute his or her task. In terms of the later ODP Reference Model[2], AHTO was to offer location, access and persistence transparency. On the other hand, mobility during an ongoing computation was not considered.

The implementation was to become a “proof-of concept”, no more, and the operating systems chosen were Unix, VAX/VMS (Digital Equipment Corp.), and OS/2 (IBM).

2.4 Basic Concepts of Distributed Computing

One of the basic ideas in AHTO was to conceptually separate the computation and its execution. A *computation* is a logically connected collection of activities invoked by an end user. Computations are independent and they do not communicate directly with each other. A computation can consist of one or more *information processing tasks*. The information processing tasks work together and they reflect the logical structure of the application at hand. The individual information processing tasks are autonomous actors, each consisting of one thread of control, they execute serially or in parallel, and they communicate when needed.

The single-threaded activity of an Information Processing Task can span machine boundaries, which are transparent to the end user and essentially transparent to the control thread. An Information Processing Task can also make *remote function calls* to any remote nodes, and even these are location transparent.

The execution of an Information Processing Task, and of a computation as a whole, is the responsibility of *processes* of local operating systems: at any instant of time, for each Information Processing Task, there exists, on some node, an active process, which is executing it. The invocations of the local processes are automatic and hidden from Information Processing Tasks (this also applies for the remote function calls).

Hence, the AHTO system allows specification of parallel computations but hides the distribution of the physical execution.

[2]Raymond, K. A.: Reference Model of Open Distributed Processing: A Tutorial. In Proceedings of the International Conference on Open Distributed Processing (Berlin, Germany: Sep. 13-16, 1993). ITU-T Recommendations in X.900 series.

The AHTO system supports several distributed architectures. A *sequential computation* consists of a series of Information Processing Tasks where the tasks are executed serially, one at a time. In a *pipeline computation* the subsequent tasks may execute concurrently, passing intermediate results along the pipeline. A *parallel computation* consists of a group of Information Processing Tasks, which execute concurrently and are free to communicate in any way, all the time.

The normal communication is connection-oriented, and the type of communication is specified through the attributes of the connection. In additon to the traditional *synchronous* and *asynchronous* communication AHTO provides *non-concurrent* communication, where the receiver is allowed to start first after the sender has terminated the connection. This type of connection is persistent: the receiver can wake up at an unspecified time (but not before the sender has closed its part of the connection).

Servers can be either *shareable* or *non-shareable*. The control is with the connection management: as long as a non-shareable server has established connections to an Information Processing Task, it remains “reserved” for the others. Obviously, this does not apply for non-concurrent use: here the server is always “free”.

The *naming* system in AHTO supports a globally unique namespace and provides location transparency for the users. It is conceptually based on the X.500 directory model, but it was also influenced by Lampson’s Global Name Service[3]. An AHTO directory covers all entities needing to be named: individual objects such as files, services, devices and human beings, and organizational entities such as project teams, mailing lists, just to name some examples. The directory can contain several overlapping organizational structures; out of these exactly one determines the administrational (”spanning”) organization of the directory.

The objects have specifying attributes that allow descriptive naming. The attributes typically specify individual objects, but they may also specify characteristics of a larger context. An example: the “environment” attribute “default printer” may have been specified for a user, for a user group, or for the department; a query for the value proceeds through the current context, upwards in the directory hierarchy, until an element with the required value specification is found. This was one of the AHTO naming features which were devised with user mobility in mind, but today one might wonder whether a user would have been very happy with this type of automatic help from the system side.

Based on the descriptive naming a basic *trading service* was designed and implemented. The trader collected static information from the directory (information retrieval) and dynamic information from relevant operating systems (remote

[3]Lampson, B. W.: Designing a global name service. Proceedings of the fifth annual ACM symposium on Principles of distributed computing (Nov. 1986).

function call), and made decisions based on user-defined rules.

2.5 The AHTO system

A proof-of-the-concept prototype of the AHTO system was implemented. The prototype included a location-transparent user control system to specify and execute AHTO computations with Information Processing Tasks on different computers. The prototype also included the essential program-level operations needed in distributed AHTO applications and remote-service invocations.

As examples of infrastructure services AHTO had the name service, a trader service, and a service for remote file accesses. As a distributed application we designed and implemented a distributed message system.

The main operating system, on which the AHTO development took place, was Unix. In addition, even the implementations on VAX/VMS and OS/2 became operational.

It is worth mentioning that a remarkable amount of implementation work was done in student projects.

2.6 The project in a nutshell

Members of the project included Timo Alanko, Petri Kutvonen, Matt Mutka, and Markku Kojo. The leader of the project was Professor Martti Tienari. The volume of the project was about 10 person years. The project was supported by Nokia Data Systems, Telecom Finland, Finnish State Computer Center (VTKK), and, as part of the FINSOFT Software Technology Program, by TEKES (National Technology Agency of Finland). It generated more than 10 M.Sc. theses, some popular articles, but only a few publications[4]. At that time TEKES was more interested in software than in scientific publications. Therefore, the main results of the project were a novel middleware prototype, and the gained experience which could later be exploited in subsequent research projects.

2.7 Spring-offs

One immediate effect of the AHTO project was a growing interest in open distributed systems in general. It had effects on the education and research at the

[4]Timo Alanko and Petri Kutvonen: The AHTO directory: a distributed directory designed for distributed usage. IEEE Distributed Processing Technical Committee Newsletter 14, 1 (1992), 13–20. Timo Alanko, Jukka Keskinen, Petri Kutvonen, Matt Mutka and Martti Tienari: The AHTO project: software technology for open distributed processing. Report A-1989-4, Helsinki University Computer Science Department.

department, and it started new departmental activities within the IT society.

The longest lasting direct effect of the project was the still ongoing participation of the department in the development work on the Reference Model for Open Distributed Processing, under the auspices of ISO and CCITT (ITU-T).

An immediate follow-up research project was the DRYAD project (1992-1996), which concentrated on heterogeneous environments where autonomously administered systems federate with each other. The federation solutions were built on meta-information services, such as the trading service and the type repository service. The DRYAD project developed a prototype software package for some of the middleware services, especially for the trading function. This line of research has led to the current Pilarcos projects.

AHTO must also be considered as the base on which the Mowgli project was developed.

Chapter 3

The Dawn of Mowgli

Timo Alanko, Markku Kojo and Heimo Laamanen

3.1 The Mobile Environment in the Early 90s

The new mobile world

The concepts *mobile office* populated by *nomadic users* emerged in the early 90s. People started to have home computers with network access (through modems, often with low bandwidth—only 300bps to 2400 bps, and a typical LAN could sustain 5 Mbps), and enterprises could afford to buy portable computers for employees (about 3 kg for $20,000$ FIM). On the wireless side, NMT telephones were in fairly common use (several hundreds of thousands, anyway) and the new digital GSM had just become available. The creative new idea was to connect the portable computer to a mobile telephone, and there it was: the untethered user who had access to his (and later hers) information sources "anytime, anywhere". For this, no changes in the software platforms were required—it was just a replacement of some copper with airborne transportation. In practice, however, some problems were anticipated. True, the data communication algorithms were still correct, but their behavior was tuned for fast and reliable networks. On the other hand, data transmission over wireless connections was not fast, and cellular telephones were expected to offer a rather variable quality of service. Especially crossing cell borders generated unexpected delays, and even a breakdown of the connection was not unusual. Facing these facts of life was the lonely end user, in the middle of nowhere, looking at his silent laptop computer, not knowing what it was doing—if anything.

Research and development

Much of today's common technology was far from ripe then—to say the least. GSM networks had been in use for some time, but the penetration level was less than 10% even in Finland. At the time the Mowgli project started, the data transmission over GSM was under field testing, and the first experimental work had to be done using the analog NMT network.

In academia wireless data communication was a hot research topic; however, the goal was to develop protocols for a wireless local area network. Computer mobility in the global world was tackled by a new IETF working group for mobile IP. However, mobile IP was never needed during the project, as the cellular network took care of the mobility allowing the user to to make a data call home. For the Mowgli project, this created unexpected problems: referees considered papers not handling cell-level handovers unripe.

Some attention had also been paid to the performance problems of TCP when wireless links were involved. The Mowgli project was one of the very first with a proposal to split the transport layer control into two, at least to some extent autonomous, parts. Efficient usage of data transmission capacity became an issue, and ETSI, for example, started the GPRS standardization work in 1994.

3.2 How it started—Enterprise Viewpoint

For anything to grow and to flourish well, there usually need to be several good roots. This was also the case with the Mowgli project. One of the roots was always going on demand to enhance the productivity of employees in the industry. The productivity of field service engineers at Digital Equipment Corporation (Digital) had been a topic of improvement at the beginning of 90s. One of the main challenges was the time that a field service engineer had to spend at the office or on the road; thus not at a customer site and not being productive enough. For example, in many cases to acquire the information needed to solve a problem, the field service engineer had either to travel back to the office or call a supporting engineer at the office to access the problem-and-solution database. The first case usually required unproductive driving time to the office and back to the customer site again, and the second case required additional support engineers at the office all the time. Thus the challenge was to minimize both these factors.

Based on the innovative work of research director Petteri Heng—and also based on the fact that Finland was considered an excellent research laboratory—Digital had established a research group in Finland to explore new ideas. One of the ideas was to use wireless data communications to enable field service engineers to access from customer sites the required support systems so far accessible only at the office; thus to reduce unproductive driving time and the number of sup-

Figure 3.1: Mowgli's First Equipment Set.

port persons at the office. The work started with a study of feasible wireless data networks—NMT (analogue system) and Mobitext (digital system)—and their features. GSM data services were at that time just promises of future improvements.

The brief study comprised overall feasibility issues and performance measurements with equipment shown in Figure 3.1. The performance measurements started with tuning modems to find out what are the best possible parameters (e.g. modulation, timeouts, and retransmit counters). The actual measurements were carried out both at the Digital office in Niittykumpu (stationary) and driving around Kirkkonummi and Savitaipale (mobile). The feasibility test included for example traveling with a field service engineer and trying to access support systems. The results indicated that wireless data communications was a feasible option, but there were several problems to be solved before it could be utilised in everyday production systems. The main problems were low and changing throughput and unreliability. But more detailed and in-depth research on the topic was required to understand the real nature of wireless data communications, and to develop solutions.

Technical feasibility alone was not an adequate motivation—especially in this new domain—to start the research project, but an economical motivation was also required by Digital's management. One of the economical motivations was based on the GSM Data Justification Model illustrated in Figure 3.2.

Every topic in this model had to had a sound justification. For example, "Reduce cost of doing business" was justified by the following factors:

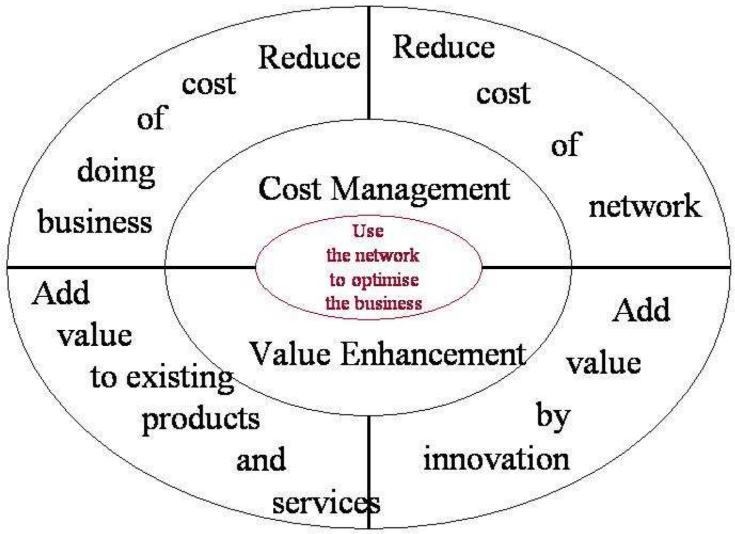


Figure 3.2: GSM Data Justification Model.

1. Less manpower, as supporting field service engineers were not needed at the office.
2. Less traveling, as necessary information could be accessed from the customer site.
3. More efficient (just-in-time) access to information.
4. Less office room, as less supporting field service engineers needed at the office and field service engineers spend more time at the customer sites.

"Add value by innovation" was justified by explaining that wireless data communications enabled totally new ways of working, as information was accessible whenever and wherever needed.

Technical feasibility and economical motivations convinced Digital's research management that it was beneficial to start a research project. However, Digital saw that it would not be wise to carry out the research alone but to do it in cooperation with a university or a research institute.

The Helsinki University Computer Science Department had a reputation as a good research organization to carry out the actual research work, and Heimo Laamanen contacted Professor Timo Alanko to discuss whether the department was interested in a research project to study and try to solve the problems discussed above. Professor Timo Alanko expressed a preliminary interest at the department. But for a well-working research project in the domain of wireless data communications, other partners were needed to obtain support and knowledge of both wireless networks and mobile terminals. After several negotiations Telecom Finland (Jorma Haaranen and Jouko Rosenberg), Nokia Mobile Phones (Mikko Terho), and Nokia Networks (Petri Pöyhönen) had seen the potential of a joint research project, and thus the research consortium (University of Helsinki, Telecom Finland, Nokia Mobile Phones, Nokia Networks, and Digital) was established. Professor Martti Tienari became the leader of the project.

The launch of the Mowgli project was also the beginning of long-lasting negotiations for the research agreement. One of the main challenges was caused by the different basis of legal systems between Finnish companies and US companies. Finally, after several rounds and months of negotiation with lawyers an agreement was achieved, and the partners signed the contract. However, the actual research work had started long before that. So in order to be successful, a research project has to resolve other problems as well as plain research ones.

Post Script

By the year 1995 the economical climate had changed so much because of the recression that in November Digital saw it to be far too early to launch any mobile and wireless data transfer product into the market. Therefore, Digital canceled all the activities related to wireless data transfer. The Mowgli research work continued with the support of the other companies and the National Technology Agency of Finland (TEKES).

3.3 Are there problems?

It is true that problems were anticipated. However, before starting to devise solutions a wise man tries to learn something about the nature of the problem.

The project started with quite extensive (at that time) field measurements. The protocol was an unmodified TCP/IP suite, and the telephone systems under test were first NMT and, later, GSM. Some of the findings were expected, some were surprises. As far as TCP control is considered, noteworthy were the long round-trip times (1–2 seconds) and especially their variability: the longest 1% were a magnitude longer than the median. The retransmission timer of the TCP adapts easily to a certain level of delays, but starts to misbehave when encountering delays that are longer than expected.

On the other hand, transferring a file proceeded at the pace of some 700–900 bytes per second, a value quite acceptable for remote users of that time, but for a desktop worker, accustomed to retrieve a megabyte file within a few seconds, a 20 minute wait might come as a surprise.

From the user's point of view response times for straightforward requests stay at an acceptable level (i.e., a couple of seconds)—on the average. However, long periods of silence would probably become stressing: is it just a long response time or are there problems with the server, or is the link broken? "What is going on—if anything?"

Even on another level long response times are a nuisance. At that time web applications were designed with a fast Internet connection time in mind. This resulted in rather chatty protocols, for example. What the user could see of it was a long, long silence while the protocols exchanged greetings at a tranquil pace.

3.4 Specifying goals

What were we up to? Two main areas were to be considered:

1. How to control and even tune the TCP/IP-based data communication?
2. How to improve the end user's quality of life?

The issue in the TCP control was TCP's ability to react, in a totally distributed way, to network congestion. The idea is simple, elegant, and effective: the sender starts carefully and increases the sending rate as long as the network seems to have sufficient capacity (the acknowledgements keep on arriving and clocking out new data segments). A long sudden delay in the round-trip time may result in premature timer expiration interpreted as a sign of congestion; the sender's reaction is to unnecessarily retransmit the current window of data and dramatically drop the sending rate.

However, in a wireless environment an unexpected long round-trip time more probably has a totally different reason: the air link had intermittent problems which resulted in an unusually long transmission time. In this case, the error recovery should be more or less the opposite when compared to congestion recovery: do not retransmit nor drop the rate but continue at full speed as soon as possible.

For an application, problems with the air link could lead to the interpretation that the link had been broken and, hence, the connection should be terminated. The termination of the connection typically resulted, at that time, in a termination of the application. A nightmare vision: a big file transfer over a poor-quality link resulting in the retrieval of several first halves of the file without a single latter half.

The first stated goals of Mowgli were:

- to improve the performance of TCP,
- to make applications fault tolerant with respect to the wireless link, and

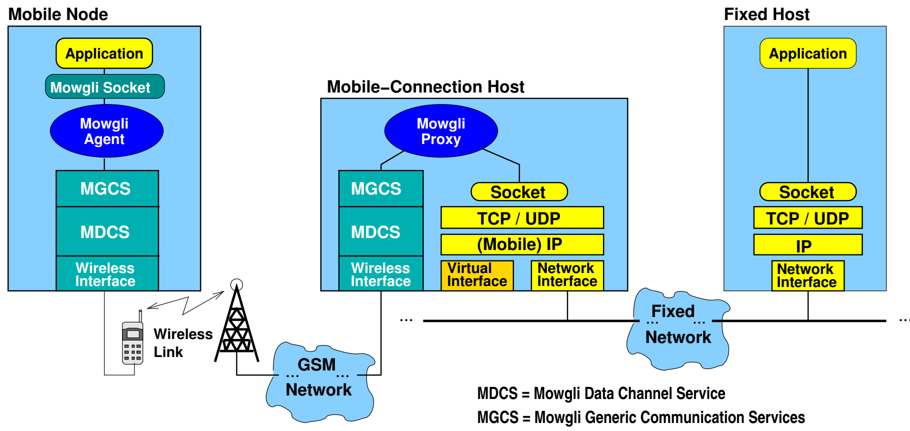


Figure 3.3: Mowgli Architecture.

- to improve the quality of life of the end user.

The restricting facts of life were obvious:

- On foreign servers: the TCP/IP protocol suite must not be touched.
- Tthe applications belonging to legacy software must not be changed, that is the socket interface must remain unmodified.
- A later inclusion of mobile IP should not be made impossible.

3.5 The Mowgli approach

Separate the different worlds

The control principles in the wireline and wireless worlds are quite different, and it is difficult to see how any compromise could benefit either side. Hence, a separation of these two behaviorally different worlds became the key idea in the Mowgli[1] approach. The major components of the Mowgli architecture are shown in Figure 3.3. The essential component for doing this is a pair of actors, functionally behaving as a distributed interceptor.

We consider an environment where the application is running in a *Mobile Node*, and this application needs service offered somewhere in the wireline network. The Mobile Node is connected to the fixed network through a GSM link, and the node which provides the Mobile Node with a connection point to the wireline Internet is called a *Mobile-Connection Host* (MCH).

[1]Mowgli is the acronym of the project name Mobile Office Workstations using GSM Links.

The client communicates with the server over a TCP connection. Now, this connection is split between the client API and the TCP/IP service, and control over the leg in between is given to the Mowgli communication system.

Here, the MCH plays a major role. The MCH contains a *proxy*, a representative of the client inside the wireline network. For the server the proxy plays the role of the (mobile) client: it is the endpoint of the TCP connection. On this connection, between the proxy and the server, the normal TCP control behaves exactly as expected.

The wireless connection, in its turn, is controlled by the proxy and by its colleague the *agent*, which resides in the Mobile Node. The agent intercepts the messages the client sends and delivers them to the proxy for further forwarding to the server. In some cases, for example when operating in disconnected mode, the agent may represent the server to the client.

To sum up:

- The client communicates with the agent.
- The agent and the proxy take care of the data transfer over the wireless link.
- The proxy communicates with the server.

The agent proxy pair operates at the application layer (or, in OSI terms, implements the session layer and can contain functionalities on the presentation and application layers). Two basic kinds of pairs are provided:

- *Generic agent and proxy* that is independent of application semantics, being capable of serving any user application.
- *Application-specific agent and proxy* that is tailored for a specific application protocol and is able to take advantage of its knowledge about the application protocol and semantics in enhancing the data communication.

The control of the data transfer is implemented as the *Mowgli Generic Communication Services* (MGCS). The *Mowgli Data Channel Service* (MDCS) covers the data-link, network and transport layers specifically designed to communication over a (slow) wireless link. Thus, the communication system can be fully tuned with respect to the special characteristics of the wireless link. It is worthwhile to notice that this approach is by no means restricted to wireless links: it can be used to control any other type of network segment with extravagant characteristics.

Eventually the Mowgli system was to include a collection of actors and services: a data channel service to control the wireless link, a data transfer service to control the tasks a user had specified, a generic agent-proxy team, which was able to serve any legacy application, and application-specific agent-proxy teams for WWW browsing, e-mail and file transfer.

Mowgli Data Channel Service

The main design objectives of the Mowgli Data Channel Service (MDCS) were to improve the fault tolerance and performance of data communication over low-performing, vulnerable links.

The MDCS was a special transport service that transparently replaced standard TCP/IP protocols on the wireless link. The basic challenge for the MDCS was to cope with the modest performance and irregular behavior of a wireless link. Hence, the traffic had to be minimized to the “bare necessity”, and the flow control had to be tailored to the wireless link’s characteristics.

The goal was to offer communication services that are similar to the TCP (and UDP) transport services. In addition, several new features were considered important. To start with, the role of the mobile node as an “end station” makes some header information purposeless—hence, it could be deleted. On the other hand, it might be wise to support several types of underlying communication media with different reliability characteristics—hence, two modes of operation were specified: *default mode* and *error-monitoring* mode. In addition, to cope with lossy wireless links, a *window-recovery* mode was considered useful and later added on.

The MDCS could independently control all connections over the wireless link. To facilitate transmission of data that differ in size and urgency, the MDCS provided priority-based multiplexing of data, a feature that turned out to give unexpected benefits in user-visible behavior of the system. This was a kind of forerunner for the QoS support widely considered in the system models of today.

Mowgli Generic Communication Service

The Mowgli Generic Communication Service (MGCS) was gradually constructed around the original generic agent-proxy team. As the basic idea turned out to work and even in a quite satisfactory way, it was considered worthwhile to try out extensions of its functionality.

The first extension was a customized agent-proxy. The idea was to adapt the agent-proxy communication service to the characteristics of the application. Not too surprisingly, the applicaton chosen was WWW browsing.

Hence, a Mowgli WWW agent and proxy pair were implemented, and these communicated with each other using the highly optimised Mowgli HTTP protocol. Obviously, for the WWW client and WWW server this pair was transparent.

When the user starts a browsing session they establish a communication session, which is maintained regardless of the current state of the wireless link; for example, during periods of disconnection the proxy can perform operations in the fixed network on behalf of the Mobile Node and the WWW agent serves the

WWW client locally.

Major performance optimizations were gained through different techniques to reduce unnecessary round trips and transfer volumes over the wireless link. Two interesting techniques were data compression and intelligent filtering. The type-specific compression turned out to be effective: lossy compression of pictures, with only a fraction of the pixels transferred, scarcely affected their quality in a noticeable way. Intelligent filtering allowed images to be left out—if the user did not want them. Besides, based on the "end-station" role of the Mobile Node the headers could be reduced to one fifth of the original size.

A performance-conscious user could use of background transfers: before starting to read a document (s)he could instruct the Mowgli WWW Agent to download further interesting-looking documents as a background task. In addition, the system itself was able to automatically do some predictive downloading.

The user control and selected data transfer modules of WWW browsing were later extended and implemented as an independent module in MGCS: the *Mowgli Data Transfer Service* (MDTS). Hereby, other applications such as e-mail and file transfer could leverage on the same useful services.

The idea in MDTS was to separate the workload specification (*what* the user wants to get done) from the actual execution control. According to user specifications, expressed by attributes of data units to be transported, the MDTS can make its own decisions about invoking operations when conditions are favorable, about postponing transfers when conditions deteriorate, about trying to recover from failures, and about canceling operations. To provide cost efficiency and fault tolerance, the MDTS was allowed to establish, release, and reestablish connections using the MDCS services, always according to the parameters specified by the higher-level communication software or by the end user. Obviously, the user must have the final control; hence, at this phase Mowgli also had a versatile user interface, which allowed the user to get information about the system status and to explicitely control it.

3.6 Proof of concept

The Mowgli architecture was in essence a new architectural approach to network-based computing. Obviously the validity of the thinking had to be shown through implementing a prototype.

The essential design phase took about a year, with several people involved but with only one hired researcher, Markku Kojo. In September 1994 the prototype implementation started as a student project, and the number of hired staff was doubled to two. The prototype became operational during the Spring 1995. The mainline development was on the Linux platform, but in the same spring a Windows version was also started, again as a student project. In summer 1995, the Mowgli WWW system was developped. The last bigger efforts took place during 1996, resulting in a revised version of the main system, with a new graphical control interface, an almost working mobile e-mail system, and a detailed security architecture. As an example of wrong decisions, the system also had a separate file-transfer service: an FTP implementation, considered important in those days, but very soon out of market, replaced by the corresponding WWW functionality. The main system reached a "beta-test" level both on the Linux and on the Windows platforms; the WWW agent and the mobile e-mail became available only on the Linux platform.

Performance testing was started early in the project—as a matter of fact, the project itself was started with performance measurements. Eventually, the main testing platform became the basic transport service and mobile WWW agent-proxy, and the quite impressive results were published in IEEE Journal on Selected Areas in Communications, September 1997 [14] and in the IEEE Global Internet 1996 Conference [9]. As a side remark it is worth noticing that the mobile WWW agent-proxy gained good PR as a demonstration tool, being demostrated in the ITU Telecom Interactive 1997 Forum in Geneva, for example. A competition between the Mowgli browser and "another browser" gave spectacular results: long after the Mowgli browser had completed and the audience had lost all interest in the competition "the other browser" was still making deliberate progress.

3.7 Some afterthoughts

The Mowgli project was a good example of how research and education can be combined. During the whole project there were typically only two hired people working in it. Markku Kojo, the chief engineer of Mowgli was always there—first hired by the project and later on a part-time basis. Then Mika Liljeberg, then Jani Kiiskinen and Heikki Helin, and later many others, about ten people altogether.

Most of the programming was done in student projects, under the guidance of the Mowgli staff. Tens of CS students carried out their software engineering project implementing Mowgli software, including Linus Torvalds who participated in the first student project.

Typically the software produced in the student groups still needed a little bit of this and a little bit of that to become fully operational. Hence, we often hired "the best of the group" to finish the software.

Senior research staff were involved only on a part-time basis. Most active in the developing, evaluation, and reporting of Mowgli were Timo Alanko, Kimmo Raatikainen, and, on the industrial side, Heimo Laamanen.

The main goal of implementation was a "proof of validation". This attitude resulted later, as the approach had shown its merits, in certain problems. In a proof of validation it is important that the main line functionality works, and this must be shown with a rather tight schedule. With this in focus, handling of exceptional situations is of minor importance. Therefore, the first versions of the data communication protocols were not based on any complete automata models, but later versions were quite complete implementations of the protocols.

Now, as the approach turned out to "work", new functionalities started to pile up on the first platform. At this stage, as the sofware started to grow in volume and complexity, a minor bugs started to announce themselves. As the development of Mowgli software was terminated, the system had hundreds of thousands of lines of source code, some of which were probably pretty close to the verge of a "don't touch me" state.

The Mowgli project published from a rather early phase (1994) to the year 2000, mostly conference papers, altogether 17 papers (see Section 3.9). In and around the project close to 15 M.SC. theses were prepared (see Section A.3).

With the early papers one unexpected problem was the challenge of making the reviewers understand that the paper was not about wireless LANs. Each time the preliminary story about cellular telephones became longer and longer. Yes, it was new then.

It is probably not a surprise that one paper (the only one with "official language check") was lost by the editor. It turned up years later, at a time when it did not contain new results any more.

3.8 The years after

The work continued. The most direct follow-up of Mowgli was the work with the TCP/IP protocol performance analysis and enhancements on different networks with different link characteristics. This work still continues with a series of projects under the *IIP*[2] acronym. Already during the Mowgli time Markku Kojo had started participation in the IETF standardization work, being a co-founder of the Performance Implications of Link Characteristics working group that was one of the first groups addressing issues with Internet protocols on wireless links and starting the era of wireless impact on the Internet protocols in the IETF. These activities still continue today, in 2004, with a number of active participants from the NODES group. The general idea in this involvement is to base standardization on research results.

A spin-off, starting from modeling desires related with GPRS, was the work on a data communication performance modeling, which resulted in the Seawind network emulator. During recent years, Seawind has been used as an important

[2] IIP stands for Improved Internet Protocols.

tool for the performance measurements studies on the IIP projects and has gained certain popularity in the academic institutes throughout the world.

Different paths were based on the agent-proxy approach. In the NODES group the agents were made mobile, and this was the area where Monads became active. In a couple of years it produced an interesting software system and some 10 publications.

Outside the university, Heimo Laamanen et al were for several years involved in the agent standardization work, organized by FIPA.

To wind up the saga of Mowgli, the academic credits earned by the staff that worked in the Mowgli offsprings: Ph.Lic. degree for Heimo Laamanen, and Ph.D. degrees for Heikki Helin, Stefano Campadello, Jukka Manner, and Andrei Gurtov.

Interested readers can learn more about our mobile research from the Mowgli Jubilee Book[3].

3.9 Mowgli publications

1. Kojo, M; Raatikainen, K and Alanko, T: Connecting Mobile Workstations to the Internet over a Digital Cellular Telephone Network. Proceedings of MOBIDATA'94 Workshop (Nov. 1994). Revised version published in Mobile Computing, Kluwer Academic Publishers, 1996, pp. 253–270.

2. Alanko, T; Kojo, M; Laamanen, H; Liljeberg, M; Moilanen, M and Raatikainen, K: Measured Performance of Data Transmission Over Cellular Telephone Networks. Computer Communication Review 23, 5 (October 1994): 24–44.

3. Liljeberg, M; Alanko, T; Kojo, M; Laamanen, H and Raatikainen, K: Optimizing World-Wide Web for Weakly Connected Mobile Workstations: An Indirect Approach. Proceedings Second International Workshop on Services in Distributed and Networked Environments (SDNE '95). (June 5-6, 1995; Whistler, Canada): 132–139.

4. Alanko, T; Kiiskinen, J; Kojo, M; Laamanen, H; Liljeberg, M; Raatikainen, K: TCP/IP Over GSM: — The Mowgli Case Study. In Proc. of 4th Summer School on Telecommunications. August 1995; Lappeenranta, Finland.

5. Kiiskinen, J; Kojo, M; Liljeberg, M and Raatikainen, K: Data Channel Service for Wireless Telephone Links. Proc. of 2nd International Mobile Computing Conference. (Mar. 25–28, 1996; Hsinchu, Taiwan) pp. 60–69. Also

[3]Kimmo Raatikainen: *Wireless Internet: Challenges and Solutions—First Ten Years of Research in Mobile Computing at Helsinki University Computer Science Department and Challenges for the Next Ones.* Report B-2004-3, Helsinki University Computer Science Department.

in IEEE-CS Bulletin of TC on Operating Systems and Applications Vol. 8, 1 (Spring 1996).

6. Liljeberg, M; Helin, H; Kojo, H and Raatikainen, K: Enhanced services for World-Wide Web in mobile WAN environment. In Proc. 3rd International Conference on Communicating by Image and Multimedia, IMAGE'COM 96, 1996, pp. 119–124.

7. Alanko, T; Kojo, M; Laamanen, H; Raatikainen, K and Tienari, K: Mobile computing based on GSM: The Mowgli approach. In Proc. IFIP World Conference on Mobile Communications: Technology, Tools, Applications, Authentication and Security (eds. J.L. Encarnacao and J.M. Rabaey), Chapman & Hall, 1996, pp. 151–158.

8. Alanko, T; Kiiskinen, J; Kojo, M; Laamanen, H; Liljeberg, M; Raatikainen, K: MOWGLI: Efficient Use of GSM Data Service. In Proc. of 5th Summer School on Telecommunications. August 1996; Lappeenranta, Finland.

9. Liljeberg, M; Helin, H; Kojo, M and Raatikainen, K: Mowgli WWW software: Improved usability of WWW in mobile WAN environments. In Proc. IEEE Global Internet 1996 Conference, 1996, pp. 33–37.

10. Laamanen, H; Alanko, T; Kojo, M and Raatikainen, K: The Mowgli project for pan-European GSM data. In Mobile Internet, IBC Technical Services, 1996.

11. Alanko, T; Hippeläinen, L; Kojo, M; Laamanen, H; Liljeberg, M and Raatikainen, K: Mowgli: Enhanced communication services for mobile computing using GSM. In Proc. ITU Telecom Interactive 97 Forum, 1997.

12. Alanko, T; Kari, H H; Kojo, M; Laamanen, H; Liljeberg, M; Raatikainen, K and Tienari, M: Communication services for mobile office in wireless WAN environments. In Global Communications: Interactive '97 (ed. R. Struzak), Hanson Cooke Ltd, 1997, pp. 219–225.

13. Alanko, T; Kojo, M; Liljeberg, M and Raatikainen, K: Mowgli: Improvements for Internet applications using slow wireless links. In Proc. 8th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 1997, pp. 1038–1042.

14. Kojo, M; Raatikainen, K; Liljeberg, M; Kiiskinen, J and Alanko, T: An efficient transport service for slow wireless telephone links. IEEE Journal on Selected Areas in Communications 15, 7 (1997), pp. 1337–1348.

15. Raatikainen, K: Mobile access to Internet: a mediator based solution. Proc. International Network Conference ’98 (Plymouth, UK, 6-9 July 1998), pp. 3–8.

16. Alanko, T; Kojo, M and Raatikainen, K: Overview of the Mowgli Project. Proc. Euromicro Summer School on Mobile Computing ’98 (Oulu, Finland, August 20-21, 1998), pp. 170–179.

17. Alanko, T; Kojo, M; Liljeberg, M and Raatikainen, K: Mobile access to the Internet: A mediator-based solution. Internet Research: Electronic Networking Applications and Policy, 9, 1 (1999), pp. 58–65.

Chapter 4

MoCo: Modeling of Concurrency as a Research Field

Martti Tienari

Formally specifying computer communication protocols was the starting point where our research in modeling of concurrency was begun in early 1980s. Gradually we took a more general view of the field and formally analyzed not only protocols but parallel processes in general. In 80s we initially constructed an analyzer named PROTAN for communication protocols specified in the ESTELLE specification language. Later on, when a more general analysis tool ARA, developed by Dr. Antti Valmari in VTT Electronics, Oulu, was available we were motivated to transfer to the LOTOS specification language in our research. Input to ARA was to be done in Basic LOTOS.

In our general study of modeling of concurrency we based our treatment in processes describable as finite labeled transition systems. Processes described in LOTOS or other input languages must first be converted into a transition system form. Special emphasis has been laid on minimizing or reducing the size of the transition systems with a goal that visual verification of process properties could be done from concise pictures, where all unessential transition labels are hidden away, drawn for the process workings. Another important goal has been to reduce the size of the global system graph by using compositional verification and reduction (minimization) of the system component graphs. Minimization has to take place based on some process equivalences preserving properties which are of interest in verification. In our research and case studies we have especially been using "decorated trace equivalences" which preserve the traces and deadlocks of the systems under study.

The first dissertation in the MoCo project was Jaana Eloranta's PhD thesis *Minimal Transition Systems with Respect to Divergence Preserving Behavioural*

Equivalences[1]. She introduced some novel divergence preserving process equivalences and demonstrated their use in some case studies. An important contribution was a theory of transition minimization, which had not been treated in the literature before her work. In the earlier literature only minimization with respect to the number of states was discussed. Her theoretical results and the transition minimization algorithms were published first in BIT as the article *Minimizing the Number of Transitions with Respect to Observation Equivalence*[2], and later as a more general theory (with M. Tienari and A. Valmari) in the article *Essential Transitions to Bisimulation Equivalences*[3].

An important line of research in MoCo has been looking for behavioral equivalences which are as weak as possible. This has been based on ideas introduced by Professor Antti Valmari, who currently works at the Technical University of Tampere and is also a Docent at the University of Helsinki. A decorated trace equivalence, called CFFD-equivalence, has had a central position in this line of research. It has been shown to be the weakest equivalence preserving traces, deadlocks and divergences in system graphs, CFFD-equivalence, was introduced by Antti Valmari and Martti Tienari in the article *An Improved Failures Equivalence for Finite-State Systems with a Reduction Algorithm*[4]. The background theory of CFFD-equivalence is treated more thoroughly in a later journal article by the same authors titled *Compositional Failure-Based Semantic Models for Basic LOTOS*[5].

A line of research handling the weakest decorated trace equivalences is due to Roope Kaivola who in 1996 defended his dissertation *Equivalences, Preorders and Compositional Verification for Linear Time Temporal Logic and Concurrent Systems*[6]. His most important result was to prove that the NDFD-equivalence, which is closely related to CFFD-equivalence but slightly weaker, is the weakest compositional (with respect to most common structural connectors) equivalence which preserves all next-timeless temporal logic properties of a system. The result was published already in 1992 at the CONCUR'92 conference (with Antti Valmari) as the paper *The Weakest Compositional Semantic Equivalence Preserving Nexttime-less Linear Temporal Logic*[7]. Later on, he published some applications of his theories as a conference paper *Using compositional preorders in the verification of Sliding Window protocol*[8]. His thesis project also resulted in two other

[1] Report-A-1994-1, Helsinki University Computer Science Department.

[2] BIT 31 (1991), pp. 576–590.

[3] Theoretical Computer Science, 179 (1997), pp. 397–419.

[4] Protocol Specification, Testing, and Verification, IX, B. Johnsson, J. Parrow and B. Pehrson (eds), Elsevier Science Publishers, Amsterdam, 1991, pp. 3–18.

[5] Formal Aspects of Computing, 7 (1995), pp. 440–468.

[6] Report-A-1996-1, Helsinki University Computer Science Department.

[7] Proceedings of CONCUR '92 (Concurrency Theory), Lecture Notes in Computer Science 630, Springer-Verlag 1992, pp. 207-221.

[8] Proceedings of the 9th International Conference on Computer Aided Verification (CAV'97),

international conference papers in 1991–93.

Roope Kaivola has an unusual career because he is a Ph.D. from both the University of Helsinki (in 1996) and University of Edinburgh (in 1997). This happened because Roope got a possibility for doctoral studies in Edinburgh at a time when his Ph.D. thesis in Helsinki was still uncompleted. The two thesis subjects are unrelated which can be seen from the title of his Edinburgh thesis: *Using Automata to Characterise Fixed Point Temporal Logics*[9]. The research in Edinburgh has been published internationally in 3 conference papers and 2 journal articles in years 1995–1998. He is currently a Docent of our department but since 1999 has worked at Intel company in USA as a hardware verification specialist. He still publishes at Intel; the title of a recent publication reveals more of his current work: *Proof engineering in the large: formal verification of Pentium-4 floating-point divider*[10].

A subject somewhat related to compositional verification is studied by Timo Karvi in his dissertation *Partially Defined Lotos Specifications and their Refinement Relations*[11]. In this thesis he studied how large Lotos specifications can be developed stepwise. Central in his work are novel refinement relations based on decorated trace semantics of systems. The properties of these relations are studied and algorithms are given to compute these. Central parts of this thesis will be published soon as a journal article accepted in the journal *Formal Methods in System Design* (together with Martti Tienari and Roope Kaivola) with the title *Stepwise Development of Process-Algebraic Specifications in Decorated Trace Semantics*.

A research line which has not so far reached the Ph.D. or Ph.Lic. status is worth mentioning here. In the report *Verification of progress properties in a shared-memory concurrent system represented as a labeled transition system*[12] by Martti Tienari and Päivi Kuuppelomäki, the problem of verifying a "leads-to" type of progress properties from a global state graph is studied. The goal was to develop a method to check progress properties from a corresponding behaviorally equivalent minimized system. A novel divergence preserving bisimulation equivalence was developed to make this possible. Unfortunately the method turned out not to be easily generalizable for systems consisting of more than two communicating processes. Thus it could not really be claimed to be a viable alternative to model-checking based on temporal logic which is the established verification method for this kind of problems.

In the latest Ph.D. thesis in the MoCo project *A Process Algebraic Reduction*

LNCS Vol. 1254, Springer, 1997, pp. 48-59

[9] PhD thesis, Report CST-135-97 (ECS-LFCS-97-356), University of Edinburgh, Department of Computer Science, 1997, 192+10 p.

[10] International Journal on Software Tools for Technology Transfer (STTT) 4(3): 323-334 (2003).

[11] Report-A-2000-5, Helsinki University Computer Science Department.

[12] Report-C-2000-3, Helsinki University Computer Science Department.

Strategy for Automata Theoretic Verification of Untimed and Timed Concurrent Systems[13] Matti Luukkainen expanded the scope of process graph minimization method to timed specification of systems. In specifying he uses a system timed automata which in addition to control states and integer variables also contain variables to record the progress of time. He shows how process algebraic compositional verification and reduction methods can be employed in the analysis of systems composed of timed automata by discretizing state spaces. In his work both the system to be verified and the correctness criteria are modeled as Buchi-automata, the theory of which needs some extension for this purpose.

The pensioned author of this article has supervised the work of the MoCo project in his position of Professor Emeritus until the year 2004 when it was agreed that the younger active researchers in the field Timo Karvi and Matti Luukkainen take the responsibility to lead the project.

[13] Report-A-2003-7, Helsinki University Computer Science Department.

Chapter 5

DRYAD and Beyond

Lea Kutvonen

At the department, the AHTO project established the ideas of a middleware layer that provides homogeneous access to networked operating system services. After AHTO was completed, the challenge was posed to establish a new topic and a new group. As a response to that challenge, further research on higher-level middleware services was initiated to support.

A row of projects was introduced over the following years:

- DRYAD (Directory adventure: traders in open networks) in 1992–1996;
- Pilarcos (Production and integration of large component systems) in 2000–2002;
- web-Pilarcos (Production and integration of systems in web services environments) from 2003 onwards; and
- TuBE (Trust based on evidence) started in 2004.

The common theme for these projects is the direction towards a service-oriented environment, where each service is autonomously administered, and used in a loosely-coupled community of peers. Such an environment is necessary and beneficial for electronic business networks. The existing middleware solutions are still not sufficient to address these needs: further concepts for managing services and collaborations are needed. Interoperability issues are still under major development and research internationally.

The DRYAD group concentrated on trading service that provides matchmaking of service requests and service offers based on service types and other properties, to support dynamic binding. The Pilarcos project enhanced the set of middleware services for analysing the interoperability requirements between services in more detail. As a further step, the web-Pilarcos project produced prototypes for B2B middleware services for establishing interoperable eCommunities, monitoring the behaviour of the eCommunities, and facilities for managing changes and breaches of the eCommunity contract during the eCommunity lifetime. The TUBE project focuses on the essential trust concepts and trust management facilities required in such an environment.

5.1 Development of open distributed computing environments and application interoperability

When the DRYAD project was initiated, the worlds of distributed computing platforms and application integration solutions were fairly different from the current ones. Corresponding to the platform and application worlds, the software engineering view of standalone, fully developed entities was still common. In the following, a few paragraphs to first comment on the platform environments, then the application integration views.

The term “open distributed processing” had in the 1990s—and still has—various interpretations, depending on the goals of the user of the term. We envisioned a three-step ladder of networked systems, traditional distributed systems, and federated systems. At each step of the ladder, more demanding requirements have been raised, and the solutions are each time built on the services realized by the previous step.

Networked systems plainly allow remote access from one computer to another. The difficulties of the heterogeneous environment caused and the lack of coordination in management activities necessitated the development of traditional distributed systems with integrated software in control. The traditional distributed systems trust a shared software layer that homogenizes the computing environment from the point of view of programmers and end-users. However, the assumptions of homogeneous environment and a shared control over systems owned by independent organizations are not realistic when world-wide information services are aspired to. Therefore, a federated system architecture is needed. In a federated environment, each service is autonomously administered and separate metainformation services are used at operational time to maintain interworking and interoperation.

The key concept of open distributed processing is interoperability. Interoperability means that services are able to exchange information in a meaningful way and to mutually use the exchanged information.

A characterizing concept of different forms of open distributed processing environments is the meaning of openness used. In networked systems, openness refers to the use of standard technologies and protocols. In traditional distributed systems, openness refers to use of technology standards and functional standards that specify interfaces, services and supporting formats to accomplish interoperability and portability of applications, data and people. In federated systems, openness refers to preparedness to negotiate about the interoperation technique in terms of available services, interfaces, protocols, remuneration and quality of service, and to make decisions on interoperation relationships to other (new) open systems.

With different concepts of openness, the semantics of interoperation is determined differently: for networked systems, interoperation is limited to standard protocols for the operating system level service like process, memory, and file management, and access control, supplemented with remote login, file transfer, and electronic mail protocols. For traditional distributed systems, the middleware services provide an enhanced set of services like distributed file systems, global name services, directory services, distributed time services, and transaction management. The goal is to provide a distribution-transparent, homogeneous environment. For federated systems, the agreements are not built into the selected software and required standards. Instead, repositories for dynamically introducing new representation formats, protocols, locations and services are used by metalevel protocols. The essential middleware services are those that take care of establishing cooperation, managing collaborations, testing interoperability preconditions, and managing the collaboration lifecycle.

To place these on a timeline, we can recall that the first versions of CORBA were introduced in 1990, CORBA 2 around 1994, and correspondingly TINA was born in 1992 and reorganized in 1998. The RM-ODP work was active in 1989–1996.

Looking at business application systems, we can consider the development of enterprise systems as a first step in collaboration architecture challenges. New ICT supported economics have generated the need for organizations to adopt IT-based solutions for their business processes. Two classes of systems have distinguished themselves: workflow management systems (WfMSs) and Enterprise Resource Planning Systems (ERPs). Both solutions automate the business processes, data transfer and information sharing across the enterprise. While the WfMS focus on process control flow, the ERP systems are strong in information-centric solutions and are more flexible in adapting new service components.

A business process became defined as multistep activity that supports an organization's mission such as manufacturing a product and processing insurance claims. Interactions between partners' external business processes may be carried out based on a specific B2B standard, like EDI, RosettaNet or bilateral agreements. B2B standards define the format and semantics of messages, bindings to communication protocols, business process conversations (e.g., joint process), security mechanisms etc. A B2B framework may have to support several B2B standards and proprietary interaction protocols.

The new generation workflow systems (IEWS, Inter-Enterprise Workflow Systems) focus mainly on the interactions at the business process layer. Early projects focus on integration of known and small numbers of tightly coupled business processes while more recent projects focus on loosely coupled processes.

To place these on a timeline, we can recall that WfMC (workflow management coalition) was founded in 1993, RosettaNet in 1998, and ebXML in 1999.

Until recently, integration across organizations has been very costly, because of different proprietary interfaces and data formats used by each application and system. A change is visible via the emergence of Web Services technologies and related standards. Although the wish that the proprietary interfaces can be replaced by ubiquitously supported standards and data formats may be unrealistic, the orientation towards service-centric and process-aware architectures will solve the essential problems. Further work is needed to develop mechanisms for introducing new interface and process standards.

Achieving full interoperability among two business partners means that they are able to collaborate at all levels of their enterprise architecture. Interoperability does not only address the ability of software components to collaborate regardless of different languages, data formats, interfaces, execution platforms, communication protocols or message formats. A systematic approach to interoperability will also take into account interoperation issues at more abstract levels, such as business process interoperability. We might define business process interoperability as the ability of business activities of one party to interact with those of another party, whether these business activities belong to different units of the same business or to different businesses.

An integrated architecture approach considers that the first step when pursuing business process interoperability among collaborating partners is to ensure their process integration. Process integration involves modeling and visualizing of enterprise scales processes such that, via certain exchanges of “events”, they can “work” together in order to accomplish a number of business services or transactions.

Besides the integrated approach, a unified or federated architecture approach can be taken. While the integrated approach requires a shared process model across business partners, the unified model requires only that there is a shared metamodel on the shared processes. The actual processing, i.e. the vertical dimension is excluded. The federated approach requires tools to exist for the partners to coordinate the selection and refinement of a shared process.

The DRYAD and Pilarcos projects have focused on the federated architecture models, which still are not commonly used. Instead the main trend in management of inter-enterprise stem from the use of the unifying model that determines and ensures interoperation between partners. Federated solutions are still less common, in lack of dynamic interoperability tests, metalevel management functionality and ontologies for matching behavior and nonfunctional aspects.

Service orientation was not commonly discussed in the 1990s. The DRYAD project was closely involved with ODP work, and thus the expressions of service orientation are phrased in terms of large-scale objects with multiple interfaces, and the interface descriptions as metainformation with a central role in the interoperability requirements. (In Pilarcos projects, the ODP objects were mapped on CORBA component model and computing platform.) In the 2000s, SOA (service oriented architecture) and the Web Services type of architecture paradigms have gained ground.

5.2 New challenges

World-wide cooperation between computing systems and telecommunication facilities requires that the overall system architecture covers situations where services are supported by independent organizations in a multi-vendor computing environment. Such a system is continuously under evolution: new services are created, and existing services are offered by a varying set of service providers. Furthermore, the history of the systems involved is different, thus introducing differences, for example in information contents, expressions, interfaces and processing technology. The major design problems in the multi-organizational environment are related to the cooperation mechanisms between the services supported by the sovereign organizations and their computing systems.

Joining the architecture goals of interoperation among sovereign systems, support for service evolutions, and support for independent system administration lead to a situation where the public availability of interface specifications is not adequate. A set of infrastructure services is required for exchanging metainformation between sovereign cooperating systems about available services and their properties.

In order to support evolution and system independence, the interoperating systems need interface information at run-time. Publicity of interfaces is a basic requirement, but not sufficient. Both evolution and sovereignty of systems require that information about services can be updated with a consistent, run-time mechanism throughout the integrated network.

A temporary interoperation relationship between independently administered and thus sovereign software components or subsystems is called an eCommunity. The eCommunity comprises autonomous service providers (software constellations at an administrative domain) jointly performing the required business process contracted. The eCommunity contract includes agreements of the communication semantics; information representation; data exchange protocols; quality of service, such as timeliness, trustworthiness, and precision; security-related information; and failure semantics and recovery protocols. The contracts can be established on two system levels, between application level objects and between platform level objects.

The eCommunity establishment between application objects requires supporting services that mediate contract-related information, meta-information, within and across system boundaries. One of these services is the trading service. The trading function is a mechanism that can be used to exchange interface information. The trading mechanism allows advertising available services, service providers, and their interface properties, and supports retrieval of service-provider information. When cooperation contracts are explicitly expressed, the trading mechanism can be used for contract negotiation. Each potential partner in eCommunities can advertise its interface properties, location, and preconditions for the contract. When an initiative for establishing the eCommunity is done, the trading mechanism can pick suitable candidates. The properties and the preconditions of the selected candidates are then merged into the contract, and the candidates are promoted to partners. The partners are expected to obey the regulations agreed upon in the contract, but are nevertheless monitored during the operation of the eCommunity for potential breaches. The failure detection and recovery protocols denoted in the contract are there to be used at the operational time.

5.3 The DRYAD trading service

The goal of the DRYAD project was to do conceptual and experimental work on collaboration between autonomously administered services. Prototypes were developed for trading service and the supporting type repository.

Trading presents a global information repository. The global repository can be updated by independent information producers throughout the world-wide network, and the information users can create effective and large information searches. Trading is expected to be a global and generic service that is able to give an equal service on various platforms, and moreover, across various platforms. However, the users of trading services do not only have expectations on the semantical contents of the service but also expect certain technical characteristics. In a federated system environment, the global trading service may need to simultaneously support different interface characteristics.

Trading differs from repository services—-like name servers, directory servers, and databases—in the assumptions made of the typical load profile. Traders are assumed to manage frequent updates and frequent queries. The set of trader clients that are allowed to update the repository is large, unknown and constantly evolving. Also the structure of stored information evolves. The actions in which traders participate are not necessarily transactions and they are intended to be small with respect to required memory space and required processing time. However, keeping the agreed response time is critical, even more critical than being able to do a full search in the repository.

Realizations of the trading functionality have to solve various problems. Trading makes information available to a large but still controllable set of users. The qualities of the available information should fulfill the user expectations: information consistency, freshness, and accuracy may be required, access times may be essential, and high probability for information availability may be crucial. However, the selected techniques may not affect the autonomy of the information-producing systems nor the information user systems. Excess of network traffic must be avoided, as well as creating new security threats to the systems. Furthermore, the trading mechanism should be able to adopt to the evolution of the mediated information contents. Moreover, the topology of trader networks that cooperate for the global trading service is constantly changing. Finally, federated environments create a special area of problems. The federable systems have autonomous administrations and different technologies, and evolution schemes.

Trading activities are described through a trading community that represent the roles of ‘importer’, ‘exporter’ and ‘trader’, as illustrated in Figure 5.1. The object in the trader role supports a repository of ‘offers’. Each offer describes properties of an entity. The offers are produced by objects in exporter roles. When an exporter sends an offer for a trader to be stored, it is said to ‘export’. When an exporter requests an offer to be deleted, it is said to ‘withdraw’ an offer. The objects in importer roles make queries to the offer repository. The query and the response to the query form an interaction called ‘import’. The import requests have a basic form that is similar to database queries: the request specifies criteria for selecting the offers to be included to the response. The objects that use the traded information are not necessarily the importers or exporters themselves. The ‘clients’ and ‘servers’ that use the information are therefore drawn as separate roles in Figure 5.1.

The trader role can be distributed to a set of autonomous trader objects. Each trader object controls a private trading domain. This arrangement has two benefits: first of all, each federable system in the federated system environment has a private agent in their possession. Secondly, the potentially huge trading community is divided into smaller partitions and thus the load on each trader object becomes reasonable.

For the trading community, it is essential that the property set specified for the interesting service type is equally understood by all objects. Furthermore, it is

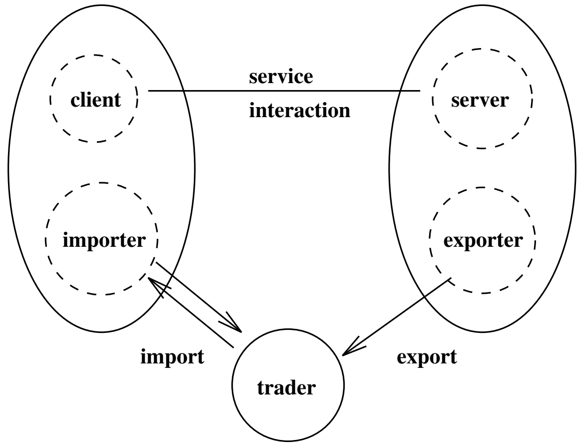


Figure 5.1: The trading community.

essential that the property names and property value sets have similar definitions and the semantical interpretation does not differ. A consistent set of service types and properties within service types can be standardized, which will enlarge the set of potential members in any trading community. However, standardization does not provide a solution that would be sufficient alone, because of the long evolution delays involved. An additional mechanism for dealing with the semantic consistency of service types and properties is a global type repository service that can provide mappings between types and property sets.

The traders are not themselves responsible for the information quality, but provide several management techniques for properties. These techniques support different qualities of information. For each property, the technique is separately selected by the exporter, based on assumptions of the change and request frequencies on the information.

The forms of property values are static, dynamic, and modifiable. These forms allow two aspects. First, the information user may be aware whether the value is prone to changes over a period of time. Second, the information producer may either push new values to the offer itself, or the trader requests a new value when needed.

The prototype software produced by the DRYAD project theme consist of

- an ODP-like Dryad trader with dynamic property values, simple type management, and interoperation between traders; the trader has a library interface and a browser (Artemis);

- management tools (Nereid) for adding service types dynamically into the system, and for defining acceptability of importers and exporters;

- abstract communication layer (Naiad) to support ODP interrogations and announcemneents; and
- an implicit trading facility (Daphne) that allows replacement of statically named and used services by dynamically traded ones.

When the DRYAD trader was developed, we actively participated in the standardization work of ODP (Open Distributed Processing), especially the trading service. That service was finally completed in 1996, and at that time the documents came out as joint ISO/ITU and OMG CORBA recommendations. The goal was not to create a standard implementation as was initially suggested, but to experiment and effect the final standard itself. Much of the DRYAD project results ended up in the standard.

Specific topics taken up with the DRYAD trader were

- need for search guidance for the trader; the import parameters covered the type of the service to be searched and criteria for considering an offer to be interesting, but in addition, rules for traversing the network of federated traders, and restrictions on the resource consumption of the search, duration of the search, and the size of the response to be received;
- the role of type definitions as a structuring rule for service offers and the collaboration contracts they become part of;
- the trader's role for mediating interface information instead of binding information to implementations;
- optimization of access to dynamic property values with prefetching policies;
- use of priorities to merge user, administrator, and trader implementation specific rules for import; this addresses the needs of trust and usability;

The project group was supervised by Professor Martti Tienari, and managed by Lea Kutvonen; Tero Venetjoki, Sami Oinonen, Max Hamberg, and Tommi Björklund were all on the payroll of the project for some period. Timo Alanko, Liisa Marttinen, Petri Kutvonen and Pekka Kähkipuro were more involved in the ODP standardisation side of the work; Liisa initially leading the ODP followup group in Finland.

Software was produced by the project group but also in some software engineering project groups: Artemis interface by Synnove Kekkonen, Jorma Korkiakoski, Ristomatti Partanen, Veikko Siivola and Heikki Vepsäläinen, Nereid interface by Juha Hatakka, Jaakko Hyvätti, Iiro Koskinen, Mika Liljeberg, Marko Moilanen, and Juha Ylitalo, Naiad package by Max Hamberg, Marko Jokinen, Timo Sivonen, and Hongying Yin. M.Sc. thesis work (see Section A.3) by Tero Venetjoki, Veli-Matti Yrjänäinen, Kirsi Vidman, Sami Oinonen, Max Hamberg, Eeva Vuorinen, Pasi Ahopelto, and Jani Boström took additional steps forward in the project.

5.4 The Pilarcos enhanced trading services and eCommunity establishment

While the focus of the DRYAD project was on implementing a feasible trading service, the focus of the Pilarcos project was on the use of trading and dynamic establishment of inter-organizational applications. The challenge of interorganisational application management stems from the sovereignty of organizations: they are free to choose supported services, develop the services independently, use differing operational policies, and use different platform technology, too.

Again, work was done on two fronts:

- architecture development for a federated environment where applications are constructed based on community descriptions populated by traded components; and
- prototyping (mainly in a CORBA environment) some essential middleware services required for negotiation of federations, instantiating communities in a distributed fashion, and creating a communication channel over a heterogeneous networking environment.

We suggested a new category of interoperability management services to take care of the needs of multidomain applications. The experimentation was done with the CORBA CCM environment, and partially with EJB support, but is applicable to other component-based or object-based platforms too. Requirements for interoperability support are minimal: shared means to express services and locations. There is no embedded requirements for example to unify component models, because such decisions are left internal to each technology domain. The Pilarcos architecture uses platform-independent contracts for expressing what kind of elements should be present and where, and lets the load management services map that need to local technology solutions.

The enhanced Pilarcos trading system populated business networks based on architecture descriptions that defined roles of participants in terms of service interfaces required and information flows between peer roles. The Pilarcos trading service uses the standard CORBA Trading service to store service offers. For population of the various roles in the architecture model, service offers are first selected to match the service type and policy requirements of the corresponding role. In addition, interoperability interdependencies between offers to fulfill adjacent roles need to be tested. The communication assumptions, information representation, and assumed overall protocol between roles are only explicated in the service offers, and therefore cannot be tested on the architecture model level. As the main experimental environment was based on CORBA, and an enhancing ODP framework, we used IDL as a service description language, supplemented with appropriate extra properties to denote non-IDL features.

The Pilarcos project provided a breeding environment for application level service collaborations. The prototype middleware allowed dynamic formation of new collaborations at the application level, without explicit concerns in the application code itself. The preparation is done based on meta-level information that is taken as configuration information when establishing the actual distributed collaboration between running service implementations. Conceptually, further needs for operational time management was studied.

The Pilarcos approach is somewhat related to the OMG MDA approach, as both of them uses a model level to ensure interoperability between services. However, the Pilarcos approach is strongly oriented towards managing constellations of existing services, while MDA is more related to constructing distributed, composed services.

The project group was lead by Lea Kutvonen, and during maternity leave by Timo Alanko. On the payroll of the project were Markku Vähäaho, Egil Silfver, Juha-Pekka Haataja, Timo Suoranta, and Janne Metso for various periods. Software was produced by the project group but also in a software engineering project group of Janne Metso, Timo Suoranta, Petri Kärki and Antti Vähäkangas. M.Sc. thesis work (see Section A.3) by Meri Hyytinen, Juha Riihimäki, Teemu Head, Mika Tolvanen, Marko Asplund, Toni Kräkin, and Roni Riska took additional steps forward in the project.

5.5 The web-Pilarcos B2B middleware

The web-Pilarcos project enhanced the B2B middleware services to the operational time environment. The experimentation environment was changed to Web Services and Java-based technologies.

The challenges of web-Pilarcos were phrased to cover two intertwined features: eCommunity lifecycle management and dynamic maintenance and monitoring of interoperability. The eCommunity lifecycle covers establishing, modifying, monitoring, and terminating eCommunities, or looking from the application service point of view, operations for joining and leaving an eCommunity, either voluntarily or by community decision. Interoperability involves the technical level of messaging, the semantic level of information presentation and the pragmatic level of exchange processes and decision making.

In the breeding environment, new development has focused on verification of interoperability preconditions in terms of behaviour descriptions available. However, static verification of models and interoperability cannot be complete. In the B2B middleware provided by the web-Pilarcos project, we find it necessary to develop control environments for monitoring and reflectively restructuring the operational eCommunities, besides a breeding environment.

In the operational time environment, monitors follow the behaviour of services. The service behaviour is monitored against the service choreography (external business process) found in the contract. In the monitoring criteria, it is possible to use rules that consider the business network status as well, for catching behaviour rules such as “payment must be received by the bank before the warehouse can ship the delivery”. The monitoring information also defines criteria for notifying the local business process management agent of breaches. In case of a serious breach, the eCommunity enters an intermediate state during which decisions are taken (potentially negotiated) on the corrective actions. As a consequence, the faulty participant can be replaced by a new member.

The project has been a joint venture with VTT, and as such funded by TEKES. The VTT group has been supervised by Seppo Linnainmaa, and managed by Jyrki Haajanen. Jyrki works on his Ph.Lic. thesis on improvement of business process automation and effect of that to business strategies. Renne Tergujeff and Jenni Kanniainen have worked in that group. In addition, Elisa and SysOpen have invested research on related topics in collaboration with the web-Pilarcos project. Long relationship in research projects have been formed with Pekka Kähkipuro (SysOpen), Kari Lehtinen and Aimo Maanavilja (Elisa). Tellabs has been a follow-up partner. In the research group at the University of Helsinki the personnel has consisted of Toni Ruokolainen, Janne Metso, Juha-Pekka Haataja and Ilja Ponka. A software engineering project group has also been involved with Riina Glinskihh, Ahti Kare, Marko Lähde, Antti Juhani Mäki, Mika Stenberg, and Teemu Virtanen as members. M.Sc. thesis work (see Section A.3) by Markku Vähäaho, Toni Ruokolainen, Markku Karppinen, Topi Laamanen, Juhani Haavisto, Ari Keränen, Jenni Löytynoja, Kai Lindstöm, Lassi Tapola, and Jouni Ojanen took additional steps forward in the project.

5.6 Conclusion

The group working on open distributed and collaborative environments has had a consistent goal of supporting the needs of electronic business networks by high-level middleware services. Current research and development trends elsewhere show that the area of research is still topical and includes major challenges. For example, the group has become involved with the European network of excellence, INTEROP. INTEROP aims to create the conditions of innovative and competitive research in the domain of Interoperability for Enterprise Applications and Software. The network brings together over 50 partners from the areas of enterprise modeling, ontologies, and architectures. Our participation in the network have concentrated on Architectures and computing platforms. During the first year, the network has produced a number of state-of-the-art surveys, and is now forming a second phase work plan based on those results.

Further work is needed on creating a consistent stack of tools from the strategical business management, through business process modeling, to middleware support for managing eCommunities. Areas of specific interest include management of non-functional features in the models and contracts, static and dynamic verification of business network models, and trust relationship concepts and management in eCommunities.

The work is gaining more stable ground, as a number of fresh Ph.D. students have committed themselves to these research goals, and the coursework and seminars leading to this area have found their position in the teaching curricula.

5.7 Publications of the group

1. Kutvonen, L., Kutvonen, P., Broadening the User Environment with Implicit Trading, Proceedings of the IFIP TC6/WG6.1 International Conference on Open Distributed Processing, Berlin, Germany, 13-16 September, 1993, Publ. North-Holland, pp. 157–168.

2. Kutvonen, L., Comparison of the DRYAD trading system to ODP Trading function draft. Technical report. Department of Computer Science, University of Helsinki. October, 1994. C-1994-51.

3. Kutvonen, L., Federation transparency in ODP trading function. Techical Report. Department of Computer Science, University of Helsinki. May, 1994. C-1994-32.

4. Venetjoki, T., Meklareiden yhteistoiminta ja sen toteutus DRYAD-projektissa (Cooperating traders and implementation in DRYAD project). MSc Thesis. Department of Computer Science, University of Helsinki. April 1994. C-1994-25.

5. Yrjänäinen, V.-M., Portieri Dryad-meklarin toteutuksessa (About fault tolerance and traders). MSc Thesis. Department of Computer Science, University of Helsinki. April 1994. C-1994-22.

6. Vidman, K., Kuorman hajauttamisen liittäminen meklariin ja meklarin tietopalvelija hajautetussa ympäristössä (Utilizing a trader in load sharing).

MSc Thesis. Department of Computer Science, University of Helsinki. July 1995. C-1995-55.

7. Kutvonen, L., Achieving interoperability through ODP Trading function. The Second International Symposium on Autonomous Decentralized systems (ISADS'95). Arizona, April 1995, Publ. IEEE Computer Society, pp. 63–69.

8. Kutvonen, L., Supporting transition to open, heterogeneous computing environment. TINA'95: Integrating Telecommunications and Distributed Computing - from Concept to Reality. Melbourne, February 1995, pp. 55–66.

9. Kutvonen, P., Kutvonen, L., Measured performance of a special-purpose database engine for distributed system software. Proceeding of the first international workshop on High Speed networks and Open Distributed Platforms. St. Petersburg, June 1995.

10. Kutvonen, P., Kutvonen, L., Measured performance of a special-purpose database engine for distributed system software. Computer Communications Journal, January 1996, pp. 22–29.

11. Kutvonen, L., Overview of the DRYAD trading system implementation. The IFIP/IEEE International Conference on Distributed Platforms, February 1996, Publ. Chapman and Hall, pp. 314–326.

12. Kähkipuro, P., Marttinen, L., Kutvonen, L., Reaching interoperability through ODP type repository framework. Extended abstract. TINA'96: The convergence of telecommunications and distributed computing technologies. August 1996, VDE Verlag, pp. 283–284.

13. Kähkipuro, P., Marttinen, L., Kutvonen, L., Reaching interoperability through ODP type repository framework. Department of Computer Science, University of Helsinki. June 1996. Technical report, C-1996-96.

14. Kutvonen, L., The role of trading function in open distributed processing infrastructure. Ph. Lic. Thesis, 1996. Department of Computer Science, University of Helsinki. Report C-1996-84.

15. Oinonen, S., Co-operation of Independent Trader Implementations. MSc Thesis. Department of Computer Science, University of Helsinki. September 1996. C-1996-73.

16. Kutvonen, L., Why CORBA Systems Cannot Federate? OMA/ODP Workshop. Cambride, UK. November 1997. Extended abstract.

17. Kahkipuro, P., Kutvonen, L., Marttinen, L., Federated naming in an ODP environment. Joint International Conference on Open Distributed Processing (ICODP) and Distributed Platforms (ICDP), May 1997. Publ. Chapman and Hall, pp. 314–326.

18. Kutvonen, L., Management of application federations. International IFIP Workshop on Distributed Applications and Interoperable Systems (DAIS97), October 1997. Publ. Chapman and Hall, pp. 33–46.

19. Kutvonen, L., Supporting Global Electronic Commerce with ODP ToolsTrends in Electronic Commerce (TREC'98). II Volume. Hamburg, Germany, June 1998. dpunkt/Morgan Kauffman Publishing.

20. Kutvonen, L., Architectures for Distributed Systems: Open Distributed Processing Reference Model. HeCSE Workshop on Emerging Technologies in Distributed Systems. Lammi, Finland, January 1998. Helsinki University of Technology, Digital Systems Laboratory, Series A, Number 50. March 1998.

21. Kutvonen, L., Trading services in open distributed environments. PhD Dissertation, 1998. Department of Computer Science, University of Helsinki. Report A-1998-2.

22. Kutvonen, L., König, H., Tienari M., Distributed applications and interoperable systems II: IFIP TC6 WG6.1 Second International Working Conference on Distributed Applications and Interoperable Systems (DAIS '99), June 28 - July 1, 1999, Helsinki, Finland.

23. Kutvonen, L., Sovereign systems and dynamic federations. Second International IFIP Working Conference on Distributed Applications and Interoperable Systems (DAIS99), June 1999. Publ. Kluwer Academic Publishers, pp. 77–90.

24. Kutvonen, L., Haataja, J-P., Silfver, E., Vähäaho, M., Pilarcos architecture. Technical report. Department of Computer Science, University of Helsinki. C-2001-10.

25. Kutvonen, L., Haataja, J-P., Silfver, E., Vähäaho, M., Evaluation and exploitation of OMG CORBA technologies. Technical report. Department of Computer Science, University of Helsinki. C-2001-11.

26. Haataja, J-P., Silfver, E., Vähäaho, M., Kutvonen, L., CORBA Component Model—status and experiences. Technical report. Department of Computer Science, University of Helsinki. C-2001-65.

27. Vähäaho, M., Silfver, E., Haataja, J-P., Kutvonen, L., Alanko, T., Pilarcos demonstration prototype—design and performance. Technical report. Department of Computer Science, University of Helsinki. C-2001-64.

28. Kutvonen, L., Automated management of inter-organisational applications. The 6th IEEE International Enterprise Distributed Object Computing Conference (EDOC2002). Lausanne, Switzerland.

29. Vähäaho, M., Haataja, J-P., Metso, J., Suoranta, T., Silfver, E., Kutvonen, L. Pilarcos prototype II. Technical report. Department of Computer Science, University of Helsinki. C-2003-12.

30. Haataja, J-P., Metso, J., Suoranta, T., Vähäaho M., Silfver, E., Kutvonen, L. Platform experiences. Technical report. Department of Computer Science, University of Helsinki. C-2003-13.

31. Kutvonen, L., Costs and benefits of Pilarcos federation middleware. Technical report. Department of Computer Science, University of Helsinki. C-2003-14.

32. Vähäaho, M., Kutvonen, L., Enhanced trading service in middleware for inter-organisational applications. Manuscript. Department of Computer Science, University of Helsinki. 2003

33. Haataja, J., Tergujeff, R., Using BPEL4WS for Supply-Chain Integration—Experiences and Evaluation. Technical Report C-2003-74, Department of Computer Science, University of Helsinki and VTT, 2004.

34. Ponka, I., Web services security: an overview. Technical Report. Department of Computer Science, University of Helsinki, 2003. C-2003-73.

35. Metso, J., Suitability of UDDI Registry for web-Pilarcos Architecture—Performance Measurements. Technical report. Department of Computer Science, University of Helsinki, 2003. C-2003-72.

36. Karppinen, M., Hajautetun järjestelmän hallinta politiikkojen avulla. Pro gradu -tutkielma. C-2003-29

37. Kutvonen, L., B2B middleware for managing process-aware ecommunities. Technical report. Department of Computer Science, University of Helsinki. C-2004-63.

38. Kutvonen, L., Managing collaboration and interoperability of dynamic business networks. Technical report. Department of Computer Science, University of Helsinki. C-2004-62.

39. Kutvonen, L., Using business network models in web-Pilarcos. In Janis Grundpenkis and Marite Kirikova, editors, CAISE'04 Workshops, pp. 284–287. Riga Technical University, June 2004. Open INTEROP Workshop on Enterprise Modelling and Ontologies for Interoperability (EMOI-INTEROP 2004).

40. Kutvonen, L., Relating MDA and inter-enterprise collaboration management. In Second European Workshop on Model Driven Architecture (MDA), EWMDA-2, pp. 84–88, September 2004. Published as technical report No. 17-04 in University of Kent.

41. Kutvonen, L., Controlling dynamic eCommunities: Developing federated interoperability infrastructure. In Workshop on Interoperability of Enterprise Systems (INTEREST 2004), 2004.

42. Kutvonen, L., Challenges for ODP-based infrastructure for managing dynamic B2B networks. In A. Vallecillo, P. Linington, and B. Wood, editors, Workshop on ODP for Enterprise Computing (WODPEC 2004), pp. 57–64, 2004.

43. Berre, A-J., Hahn, A., Akehurst, D., Bezivin, K., Tsalgatidou, A., Vermaut, F., Kutvonen, L. and Linington, P. F.: State-of-the art for interoperability architecture approaches – Model driven and dynamic, federated enterprise interoperability architectures and interoperability for non-functional aspects. Deliverable D9.1 of INTEROP NoE. December 2004.

Chapter 6

Darfin: Database Architecture for Intelligent Networks

Kimmo Raatikainen

Darfin

Database ARchitecture For Intelligent Networks

University of Helsinki • Department of Computer Science

P.O. Box 26 (Teollisuuskatu 23) • FIN-00014 University of Helsinki • Finland

email: Kimmo.Raatikainen@cs.Helsinki.FI • fax: +358 0 708 4441

The research in the area of telecommunications software started 1993 at the department in the Darfin-project. It examined database architectures that can fulfil the requirements of Intelligent Networks (IN) and Telecommunication Management Networks (TMN). Until 1996 the project was funded by Telecom Finland. Later the research was carried out in TEKES-projects RODAIN and RTD-Pilot.

In 1995 we had long discussions with Telecom Finland to start co-operation with Telenor, that is Telecom Norway. The plan was that Juha Taina would move to Norway, and also Kimmo Raatikainen would spend a lot of time there. However, the price that Telenor asked was too high for Telecom Finland and the project remained in Finland.

At the beginning the researchers were Kimmo Raatikainen and Juha Taina. Tiina Niklander joined the team in 1995. Later Petri Elovaara, Jukka Kiviharju, Jan Lindström, and Pasi Porkka entered the research group. In 2000 Lea Kutvonen took the leadership of the RTD-Pilot project, and Ilpo Lyytinen and Vilho Raatikka were recruited into the project.

6.1 Research Premises

Databases will already in the near future have a central role in telecommunications networks. The information needed in operations and management of the nets will be collected into a logically uniform database. The world-wide nature of telecommunications prescribes that the only possibility to obtain the logical uniformity is the co-operation of autonomous databases.

During the next ten years great changes are expected in the ways of doing business on the telecommunications markets. The relative importance of traditional transmission and switching services will decrease. This is due both to liberalisation of telecommunications and to developments in the technology. We anticipate that in the future transmission and switching are done using generic technology based on international standards.

The ability of a telco operator to survive in the international competition will be based either on the huge volume of the few bulk services or on enhanced services. The ability to introduce enhanced services will primarily be based on the competence of responding in time to the needs on the markets. Future telecommunication architectures like IN and TINA are developed to meet the goal of rapid and efficient creation, deployment, and management of new services. An efficient and reliable database system is one of the key factors in meeting those goals.

Most of the recommendations issued in the 90s by ITU-T and other standardisation bodies are object oriented. Therefore, object orientation is a natural starting point for the telecommunications database system, too. However, object-oriented databases are still emerging. They are primarily designed for situations where objects have complex structures and transactions are long-living. In telecommunications the situations are different. Most objects have quite simple structures and transactions are short. In addition, we anticipate that real-time properties, guaranteed response time in particular, are soon needed.

Due to the world-wide nature of telecommunications international standards—both official and de facto—are mandatory. We have identified the following standards as our prerequisites:

- IN recommendations by ITU-T and ETSI
- TINA architecture by Tina Consortium
- TMN by ITU-T and ETSI
- RM-ODP by ISO and ITU-T
- CORBA by Object Management Group (OMG)
- ODMG-93 by Object Database Management Group (ODMG)

6.2 Darfin Objectives

The research project Darfin focused on object-oriented real-time databases. In particular, it identified its research topics as:

- real-time transaction processing,
- main memory databases,
- object models,
- distributed object directory,
- performance evaluation methodologies for real-time databases, and
- OSI Directory and Management protocols (X.500 and X.700).

6.3 Rodain Objectives

In the research project Rodain we continue the work done in the Darfin project. The ultimate objective of the Rodain project is to design and to specify a real-time object-oriented database architecture for Intelligent Networks and to implement a prototype based on that architecture. In particular, the following research activities were planned to be carried out:

1. Evaluation of Concurrency Control Algorithms for Real-Time Databases
2. Specification of Data Models for Real-Time Databases in Telecommunications
3. Evaluation of Platforms for Real-Time-Databases
4. Specification of RODAIN Database Architecture and Interfaces

6.4 Darfin Results and Rodain Directions

The results of the Darfin project are published in several public reports and international publications (see Section 6.6). Below we briefly summarise the key results and their implications to the research that was to be done in the Rodain project.

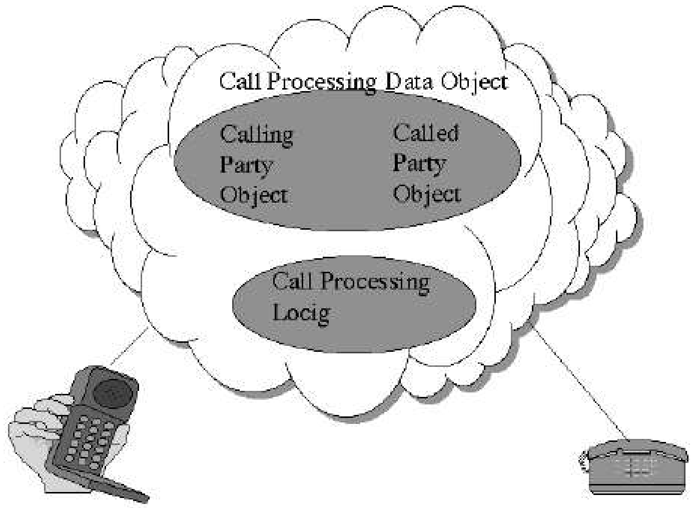


Figure 6.1: A High-Level Information Model of Phone Call

6.4.1 High-Level Information Model

On an abstract level we can interpret that a telephone call is controlled by call processing logic and data. The data can be termed *Call Processing Data Object*. It is a dynamic object that is created during the call establishment. During the call the object may change as a result of the so-called mid-call service control actions. Logically the data object can be divided into two parts: *Calling Party Object* and *Called Party Object*. Moreover, the data object may have attributes that trigger actions to modify and/or to create persistent data objects when the call is terminated.

Figure 6.1 summarises our abstract information model of a telephone call. It should be noted that the call is not necessarily between two telephones. The called party may also be an information processing system. User management of services is a typical case in which the called party is an information processing system.

The *Call Processing Data Object* is typically created in two phases. The first phase consists of creating the *Calling Party Object*. When the switch detects that the hook is lifted, a query is triggered. The query builds up the object by retrieving the service profile for the originating line from the persistent subscriber data object.

The *Called Party Object* is created in the second phase. This object depends on the access number dialled. Typically, the dialled digits are given as the key in the query that creates the *Called Party Object*. The query builds up the object by retrieving the service profile for the destination line from the persistent subscriber data object.

6.4.2 Real-Time Transactions

Based on the IN Recommendations of ITU-T (Q.1204 and Q.1211) we have concluded that real-time transaction processing capabilities will be needed. Real-Time Database System (RT-DBS) is a transaction processing system that tries to satisfy the explicit timing constraints (deadlines) associated with each incoming transaction. The ultimate goal of any RT-DBS is to maximize the fraction of transactions that meet their deadline. This is very different from the goal of traditional database management systems (DBMSs) that try to minimize the average response time.

When a database system for telecommunication services is to be designed, the list of most important issues includes real-time. We anticipate that both firm and soft deadlines will be needed. When the deadline is firm (the outcome of a late transaction is useless or harmful), then the late transaction (or transaction that can no more meet its deadline) can be aborted without restart. When the deadline is soft (the outcome of a late transaction gradually loses its value), then the late transaction is continued with a lower priority. We do not believe that hard (meeting the deadline must be guaranteed) deadlines are needed in telecommunications applications.

One of the primary difficulties in designing an RT-DBS is the fact that the concurrency control of data access must be combined with real-time scheduling. The major design issues[1] in concurrency control are conflict detection and conflict resolution. Of the algorithms proposed in the literature our current favourite is the 2PL-HP algorithm[2]. In that algorithm a transaction with a higher priority aborts the transactions with lower priorities when a lock conflict occurs. The transactions with lower priorities are later restarted if the system still estimates that their deadlines will not expire. However, the Rodain project will carry out an evaluation study of the concurrency control algorithms for real-time databases.

In the Darfin project we have learned the basic understanding of real time, object orientation, and database needs in telecommunications. In particular, the simple prototype implemented gave us valuable experience on the problem space.

[1] see i.e. Raatikainen, K. and Taina, J. 1995. Design Issues in Database Systems for Telecommunication Services. Proc. of IFIP TC6 Working Conference on Intelligent Networks. August, Copenhagen, Denmark. and Yu, P. S., Wu, K.-L., and Son, S. 1994. On Real-Time Databases: Concurrency Control and Scheduling. Proceedings of the IEEE 82,1 (January): 140–157.

[2] Haritsa, J. R., Carey, M. J., and Livney, M. 1992. Data Access Scheduling in Firm Real-Time Database Systems. Journal of Real-Time Systems 4, 3 (September): 203–241.

The database architecture should be flexible due to different kinds of needs present in telecommunications[3]. Based on the experience we believed that the easiest way to implement the required functionality is to use an object-based approach. An open question was whether a fully object-oriented architecture—something similar to ODMG-93[4]—is really needed. We believed that it was better to use a reduced set of object features than to use a fully object-oriented database when we developed a real-time database to be used in telecommunications. In Rodain we started from a real-time database system and included the necessary object features.

6.4.3 Object Models

In the Darfin project we evaluated four different object models: X.500, X.700, OMG, and ODMG[5].

Of those models the X.500 model is the most mature. However, there are some major shortcomings in X.500. The two most serious ones are the formal organisation of data into a directed graph and the lack of atomic transactions. Minor shortcomings include the lack of multiple inheritance, operations, relationships, extents, keys, and transient objects.

The formal organisation as a directed graph makes it complicated to use X.500 as the object model of the database system for telecommunication services. As pointed out in the usage scenario, the queries have different criteria in the select operation: the line number and the dialed digits. Moreover, the management needs at least two different views into the data: one in which all services of a single subscriber can be found and another in which all subscribers of a single service can be found.

The Management Information Model defined in the X.700-series of ITU-T Recommendations is a full object model. Although a distributed management system as defined in X.700 is actually quite close to a database management system, most drawbacks identified in X.500 are also present in the X.700 Information Model. In particular, it is stated in the X.700 recommendations that the local databases containing the information are independent from the recommendations. When taking the management's point of view, this is flexible. However, the approach leaves everything open when the point of view is that of an object-oriented database system.

[3] for details see Taina, J. 1995. Requirements Analysis for Database Services in Telecommunications. Report C-1995-17. Department of Computer Science, University of Helsinki.

[4] Cattell, R. G. G. (Ed.). 1994. The Object Database Standard: ODMG-93. Version 1.1 edition. Morgan Kauffmann, San Francisco, Calif.

[5] for details see Raatikainen, K., Karttunen, T., Martikainen, O., and Taina J. 1995. Evaluation of Database Architectures for Intelligent Networks. Proc. of Telecom95 Technical Summit, Vol 2, 549-553. ITU, Geneva, Switzerland.

The ultimate goal in OMG is portability and interoperability. These goals are also very important in telecommunications. However, they are so general that the OMG Core Object Model does not take into account the specific needs of databases. Therefore, our choice as the baseline object model is the ODMG Object Model. However, the OMG—particularly the OMG Object Request Broker—will obviously have an important role in future telecommunication systems.

The ODMG-93 is designed to be a general-purpose object database system. Therefore, all features presented in the ODMG-93 Release 1.1 are neither appropriate nor needed in a real-time database system[6]. An *"RT-ODMG"*, that is a real-time ODMG, will be a subset of ODMG-93 with predefined types of `real-time object` and `real-time transaction`. Type `real-time object` will include attributes that specify the correctness criteria and resource demands of operations. Type `real-time transaction` will include attributes that may overwrite the correctness criteria of operations and that specify the deadline and criticality for the instances of this type.

In Rodain we developed a data model tailored for the needs of telecommunications. We examined how the object model of ODMG-93 should be modified to meet those needs. The interfaces of data access were flexible. In particular, applications using the database services could use INAP as specified in the IN Capability Set 1 of ITU-T (Q.1218), CMIP as specified in X.700, ODP channels as specified in RM-ODP (X.900 series), and CORBA as specified by OMG. In addition, we provided an Object Query Language (OQL) interface which was a subset of the ODMG OQL.

6.4.4 Correctness Criteria for Real-Time Transactions

Correctness of transaction results in real-time databases is an interesting area of research. In real-time applications concurrent transactions do not necessarily need *serializability*[7] to produce a correct result. In many situations the result of a transaction can be regarded as correct as long as the result corresponds to the real-world situation presented in the database even if the transaction does not serialise with other transactions.

We introduced a correctness criteria called *τ-serializability* [8]. It was used to reduce the number of serialisation conflicts. In τ-serializability a transaction was allowed to read old data as long as the update is not older than a predefined time.

---

[6] Taina, J. and Raatikainen, K. 1995. Design Issues and Experimental Database Architecture for Telecommunications. Intelligent Networks 1995. Chapman and Hall, London, UK.

[7] Eswaran, K. P., Gray, J., Lorie, R. A. and Traiger, I. L. 1976. The Notions of Consistency and Predicate Locks ib a Database System. Communications of the ACM 19, 11 (November): 624–633.

[8] Raatikainen, K., Karttunen, T., Martikainen, O., and Taina J. 1995. Evaluation of Database Architectures for Intelligent Networks. Proc. of Telecom95 Technical Summit, Vol 2, 549–553. ITU, Geneva, Switzerland.

6.5 Perspective

The research must be classified as a success. Juha Taina and Jan Lindström successfully defended their Ph.D. theses in 2004. Tiina Niklander achieved her Ph.Lic. degree in 2003. The project puhlished a lot (see Section 6.6). Petri Elovaara, Jukka Kiviharju and Pasi Porkka completed their M.Sc. thesis.

In one sense it was unfortunate that the project line ended in 2002. However, related work continued in the Non-Stop Real-Time Linux -project (funded by Nokia Research Center). Toni Strandell and Vilho Raatikka completed their excellent Master theses related to databases. Toni's thesis was about Open Source databases for telecommunications, and Vilho's about cacheline-friendly index structures.

One of the problems in the research area of real-time databases was that it also addressed issues in the *Information Systems* section. Inside the NODES Group the theme was bundled together with operating system research, particularly with Linux developments. We discussed with the Information Systems about a joint "venture", but that did not work out. In 2003/4 we also tried to establish a group working on mobile data management, but did not get enough support.

Operating systems was a strong research area at the department in the late 70s and early 80s. Many M.Sc. theses were completed but the persons moved to industry (that is Nokia: Ari Ahtiainen, Tero Peltola, Mauri Saksio, Jari Simolin, to name a few examples). In the 90s Linus Torvalds developed the Linux operating system. As a research topic, however, operating systems were not in the focus until the late 90s.

In the EC/Esprit project HP-GIN Auvo Häkkinen, Juha Sievänen and Taneli Vähäkangas developed an I_2O device driver for Linux. Gradually our group involved in Linux software analysis and developments increased through funding from Nokia. Recently, the focus has been on timeliness and high-availability in Linux. Today we are increasing our research efforts in the area of operating systems. In particular, we will examine new operating system concepts for reconfigurable and context-aware end-user systems.

6.6 Publications

1. Kimmo Raatikainen: Database Access in Intelligent Networks, in Intelligent Networks (Chapman & Hall, 1995), pp. 173–193.

2. Juha Taina: Problem Classes in Intelligent Network Database Design, in Intelligent Networks (Chapman & Hall, 1995), pp. 194–207.

3. Kimmo Raatikainen: Information Aspects of Services and Service Features

in IN CS-1, Technical Report C-1994-45, Univ. of Helsinki, Dept. of Computer Science.

4. Juha Taina: Evaluation of OMG, ODMG, X.500, and X.700 Data Models, Technical Report C-1994-54, Univ. of Helsinki, Dept. of Computer Science.

5. Kimmo Raatikainen and Juha Taina: Design Issues in Database Systems for Telecommunication Services, Technical Report C-1995-16, Univ. of Helsinki, Dept. of Computer Science. Also in Proc. IFIP IN'95 Conference.

6. Juha Taina: Requirements Analysis for Database Services in Telecommunications, Technical Report C-1995-17, Univ. of Helsinki, Dept. of Computer Science.

7. Juha Taina, Tapani Karttunen, Olli Martikainen, and Kimmo Raatikainen: Evaluation of Database Architectures for Intelligent Networks, Technical Report C-1995-29, Univ. of Helsinki, Dept. of Computer Science. Also in Proc. TeleCom'95, Vol 2., ITU, pp. 549–553.

8. Juha Taina, Mika Rautila, and Kimmo Raatikainen: An Experimental Database Architecture for Intelligent Networks, Technical Report C-1995-30, Univ. of Helsinki, Dept. of Computer Science. Also in Proc. IFIP IN'95 Conference.

9. Juha Taina and Harri Töhönen: On Implementing a Prototype IN Database Architecture, Technical Report C-1995-42, Univ. of Helsinki, Dept. of Computer Science.

10. Juha Taina, Petri Elovaara, Tiina Niklander and Kimmo Raatikainen: Platform Requirements of a Real-Time Object-Oriented Database Management System for Telecommunications University of Helsinki, Department of Computer Science, Series of Publications C, No. C-1996-30. May 1996.

11. Kimmo Raatikainen: RODAIN: A Real-Time Object-Oriented Database System for Telecommunications. Presented at TINA'96 Workshop, September 2, 1996. Heidelberg, Germany.

12. Juha Taina and Kimmo Raatikainen: Experimental Real-Time Object-Oriented Database Architecture for Intelligent Networks. Engineering Intelligent Systems 4, 3 (Sep 1996): 141–154.

13. Juha Taina and Kimmo Raatikainen: RODAIN: A Real-Time Object-Oriented Database System for Telecommunications. Presented at DART'96 Workshop, November 15, 1996. Rockville, Maryland.

14. Kimmo Raatikainen: Design and Architecture for Database Solutions in Future Telecommunication Services. Presentation in BARit 97, Luleå, Sweden, September 16-17, 1997.

15. Jukka Kiviniemi, Tiina Niklander, Pasi Porkka and Kimmo Raatikainen: Transaction Processing in the RODAIN Real-Time Database System. In Real-Time Database and Information Systems, Bestavros, A. and Fay-Wolfe, V. (eds.), Kluwer Academic Publishers, September 1997.

16. Tiina Niklander, Jukka Kiviniemi and Kimmo Raatikainen: A Real-Time Database for Future Telecommunication Services. In Intelligent Networks and Intelligence in Networks, Gaïti, D. (ed.), Chapman & Hall, September 1997.

17. Pasi Porkka and Kimmo Raatikainen: CORBA Access to Telecommunications Databases. In Intelligent Networks and Intelligence in Networks, Gaïti, D. (ed.), Chapman & Hall, September 1997.

18. Juha Taina and Kimmo Raatikainen: Database Usage and Requirements in Intelligent Networks. In Intelligent Networks and Intelligence in Networks, Gaïti, D. (ed.), Chapman & Hall, September 1997.

19. Kimmo Raatikainen: Real-Time Databases in Telecommunications. In Real-Time Database Systems: Issues and Applications, A. Bestavros, K.-J. Lin, and S.H. Son (eds), Kluwer, May 1997.

20. Kimmo Raatikainen and Pasi Porkka: Database Usage in Telecommunications through CORBA. In Proceedings of the 7th IEEE Intelligent Network Workshop, Bordeaux, France, May 10-13, 1998.

21. Kimmo Raatikainen: Recent Developments in Telecommunications Software Architectures. In Proceedings of the FDPW'97–98: Developments in Distributed Systems and Data Communications, Petrozavodsk University Press, 1998, pp. 64–87.

22. Tiina Niklander: Fault Tolerance Features of RODAIN Database Architecture for Intelligent Networks. In Proceedings of the FDPW'97–98: Developments in Distributed Systems and Data Communications, Petrozavodsk University Press, 1998, pp. 112–129.

23. Tiina Niklander and Kimmo Raatikainen: RODAIN: A Highly Available Real-Time Main-Memory Database System. In Proceedings of the IEEE International Computer Performance and Dependability Symposium (IPDS '98), IEEE, 1998, p. 271.

24. Jan Lindström: Dynamic Adjustment of Serialization Order Using Timestamp Intervals in Real-Time Databases. In Workshop on Modelling, Measuring and Quality of Service. Proceedings of the 7^{th} Summer School on Telecommunications, Research report 62, Lappeenranta University of Technology, 1998, pp. 24–40.

25. Jan Lindström, Tiina Niklander, Pasi Porkka and Kimmo Raatikainen: A Distributed Real-Time Main-Memory Database for Telecommunication. In Jonker W. (ed.) Databases in Telecommunications; International Workshop, Co-located with VLDB-99 Edinburgh, Scotland, UK, September 6, 1999, Proceedings, LNCS 1819, Springer, 2000.

26. Juha Taina and Kimmo Raatikainen: Requirements Analysis of Distribution in Databases for Telecommunications. In Jonker W. (ed.) Databases in Telecommunications; International Workshop, Co-located with VLDB-99 Edinburgh, Scotland, UK, September 6, 1999, Proceedings, LNCS 1819, Springer, 2000.

27. Juha Taina and Sang H. Son: Towards a General Real-Time Database Simulator Software Library. In Proceedings of Active and Real-Time Database Systems, 1999.

28. Jan Lindström and Kimmo Raatikainen: Dynamic Adjustment of Serialization Order Using Timestamp Intervals in Real-Time Databases. In Proceedings of 6^{th} International Conference on Real-Time Computing Systems and Applications (RTCSA-99), 1999.

29. Jan Lindström: Extensions to Optimistic Concurrency Control with Time Intervals, Proceedings of the 7^{th} International Conference on Real-Time Computing Systems and Applications (RTCSA'2000). IEEE Computer Society, 2000, pp. 108–115.

30. Jan Lindström and Kimmo Raatikainen: Using Importance of Transactions and Optimistic Concurrency Control in Firm Real-Time Databases, Proceedings of the 7^{th} International Conference on Real-Time Computing Systems and Applications (RTCSA'2000). IEEE Computer Society, 2000, pp. 463–467.

31. Tiina Niklander and Kimmo Raatikainen: Using Logs to Increase Availability in Real-Time Main-Memory Database. In Rolim J (ed.) Parallel and Distributed Processing; 15 IPDPS 2000 Workshops Cancun, Mexico, May 1-5, 2000; Proceedings, LNCS 1800, Springer, 2000, pp. 720–726.

32. Jan Lindström and Kimmo Raatikainen: Using Real-Time serilaizability and Optimistic Concurrency Control in Firm Real-Time Databases. In Caplinskas A (ed.) Databases & Information Systems Proceedings of the 4th IEEE International Baltic Workshop Vilnius, Lithuania, May 1-5, 2000, pp. 25–37.

33. Jan Lindström: Using Optimistic Concurrency Control in Firm Real-Time Databases. In HeCSE Winter School 2000: Next Millenium Computing, Report Series of the Laboratory of Information Processing Science of the Helsinki University of Technology, 2000.

34. Jan Lindström: Benchmark for real-time database systems for telecommunications. Databases in telecommunications II: VLDB 2001 international workshop, Rome, Italy, September 10, 2001, pp. 88–101.

35. Tiina Niklander: Increasing availability of real-time main-memory database. Supplement to the International Conference on Dependable Systems and Networks, 1-4 July 2001, Göteborg, Sweden, pp. B-48–49.

36. Jan Lindström and Tiina Niklander: Impact of operating system on real-time main-memory database system's performance. In. Proc. 9^{th} International Conference on Real-Time and Embedded Computing Systems and Applications, RTCSA 2003, Tainan, Taiwan, 18-20 February 2003, pp. 448–459.

Chapter 7

Our First Steps in European Co-operation

Kimmo Raatikainen

The fourth European framework program started in 1995. In the ACTS [1] program, which was a part of the fifth framework, a new concept of *National Hosts* was introduced. In Finland VTT gathered a consortium of industry and academia. Kirsi Valtari from VTT invited our department to the Finnish National Host "organisation" that called itself the *Mobile National Host*.

The idea behind the national hosts was that national orgranisations provided infrastructure and services for the ACTS projects. Needless to say that the concept did not work. Projects did not want to spend money on services provided by the national host. Our department offered Dryad and Mowgli software as well as ODP consultation and courses. By selling one Mowgli license (to the DOLMEN project) we were high on the list of generating revenue.

Nevertheless, the Mobile National Host served us well in getting the first contacts to Europe. Late in 1994 VTT arranged an ACTS Mobile Conference in Otaniemi. From the department we had a large delegation. Martti Tienari, Tima Alanko, Kimmo Raatikainen, Markku Kojo, Petri Kutvonen, Mika Liljeberg, and Juha Taina participated in the conference. Tommi Björklund, Lea Kutvonen, Liisa Marttinen, and Tiina Niklander occupied our stand.

Kirsi Valtari from VTT had arranged a meeting with Paul Reynolds from the University of Plymouth and Steve Hope from Orange. The people from the UK presented a project idea in which the ODP reference model and object-orientation had a central role. Some partners of the forthcoming consortium met in Brussels, and Sebastiano Trigila from FUB[2] was selected to be the coordinator. Later

[1] Advanced Communications Technologies and Services

[2] Fondazione Ugo Bordoni, Rome, Italy.

there was a meeting in Rome, in which the proposal under name Raphael was completed. The concepts of *Service Architecture* and *Service Machine* took the central position in the proposal, but the ODP reference model was still there.

In the submission phase the coordinator arranged one additional surprise. The day after the submission deadline, Sebastiano Trigila announced hat he had submitted the proposal under the name DOLMEN. His motivation was that the name Raphael has been widely around when we were looking for suitable partners. Perhaps this was not a bad move. The DOLMEN project got an invitation to negotiations with commission. Our budget was cut only by c. 30%. We were also told to get additional partners to make the consortium stronger. In Finland, we tried to get Nokia involved but we were not successful. Instead, we made Telecom Finland interested, and they joined the DOLMEN team.

The first half year of DOLMEN was chaotic. The coordinator almost raised the red flag. Nevertheless, the project turned out to be a very successful one. In particular, the Finnish "subteam" (VTT, Telcom Finland, and we) got a crucial role. Together with KPN and Lucent from the Netherlands and Intracom from Greece we created the DOLMEN success story.

During the project, the ODP reference model was replaced by TINA[3], OMG's CORBA and Mowgli's moderator approach. As described in the Mowgli Jubilee Book[4], our DOLMEN work led to an OMG standard[5] in 2002.

In retrospective, the DOLMEN opened for us new paths to European co-operation. By taking an active role in ACTS co-operation we got involved in other European projects including:

MONTAGE: MObile iNTelligent AGEnts in Accounting, Charging, and Personal Mobility Support. An EC/ACTS project that examined agent technology in the TINA[3] context. Our group was involved in personal mobility solution and prototype implementation.

PRIME: PRomoting Interoperability for Multimedia communication in Europe. An EC/ACTS that acted as a bridge between ACTS projects and certain standardization forums.

CRUMPET: Creation of User-friendly Mobile Services Personalised for Tourism.

[3] Telecommunications Information Networking Architecture; Inoue, Y.; Cuha, D. and Berndt, H.: The TINA Consortium. IEEE Communications Magazine, IEEE 36, 9 (Sept. 1998): 130–136. Appeldorn, M.; Kung, R. and Saraccor, R.: TMN+IN=TINA. IEEE Communications Magazine 31, 3 (March 1993): 78-85.

[4] Kimmo Raatikainen: *Wireless Internet: Challenges and Solutions—First Ten Years of Research in Mobile Computing at Helsinki University Computer Science Department and Challenges for the Next Ones.* Report B-2004-3, Helsinki University Computer Science Department.

[5] Raatikainen, K.: Wireless Access and Terminal Mobility in CORBA. *Int. Journal on Wirless and Optical Communications* 1, 2 (Dec. 2003) pp. 147–163.

An EC/IST project that developed an agent-based tourist guide for handheld devices.

VAAWIT: Wireless Internet Applications for Agriculture. An EC/AsiaIT project that examined the applicability of wireless Internet for agriculture in developed (Scotland) and developing countries (Sri Lanka, Thailand, Malaysia).

As a subcontractor for Nokia Research we got involved with the EC/IST projects BRAIN (Broadband Radio Access over IP Networks) and MIND (Mobile IP based Network Developments). Later the academic partners of BRAIN/MIND created the ANWIRE-project (Academic Network for Wireless Internet Research in Europe) that brought together 12 leading European universities in Wireless Internet.

In the sixth framework program we are involved in one Network of Excellence (*INTEROP*). The new megalomanic (over 100 person years per year) Integrated Projects do not seem to be effective: management overhead exceeds the volume of research activities.

Chapter 8

The Next Ten Years

Kimmo Raatikainen and Lea Kutvonen

8.1 Nodes 1995–2004

Each research group evolves over time. The same must also be true with the NODES Group. Our background is in the KOPS[1]. We evolved from operating systems and performance analysis research to distributed and concurrent systems. Later we also got involved with mobile computing and communication.

In the very beginning NODES research topics were: **distributed systems** and **telecommunication software**. After five years, that is in 1999/2000, we had four topics, each with some subtopics: **Nomadic Computing** (*Communication over wireless* and *Adaptation to tempo-spatial changes*), **Distributed Software Systems** (*Co-operation of autonomous system* and *CORBA performance*), **Formal Methods** (*Protocol verification*), and **Operating System Enhancements** (*High-speed network device drivers* and *Real-time database systems*). Today we have five topics: **Wireless Internet** (*Communication over wireless (all protocol layers)* and *Mobile middleware*), **Open Distributed and Collaborative Systems** (*Co-operation of autonomous systems*), **Formal Methods** (*Protocol verification*), **Trust, Privacy, Security**, **Linux Developments** (*Timeliness and high availability in Linux* and *Linux–RTOS interworking*).

The most notable change during the last five years has been the inclusion of the theme *Trust–Privacy–Security*. However, inside the four "old" themes there have been changes in the content as indicated in the previous chapters. During 2004 we have recognized the need for reconsidering our mission. The current mission slide is shown in Figure 8.1 The slogan *"Any technology distinguishable from magic is insufficiently advanced"* is still valid. As long as computing

[1]KOPS was an abbreviation of *Käyttöjärjestelmät—OPerativ System*, that is Operating Systems in Finnish and Swedish.

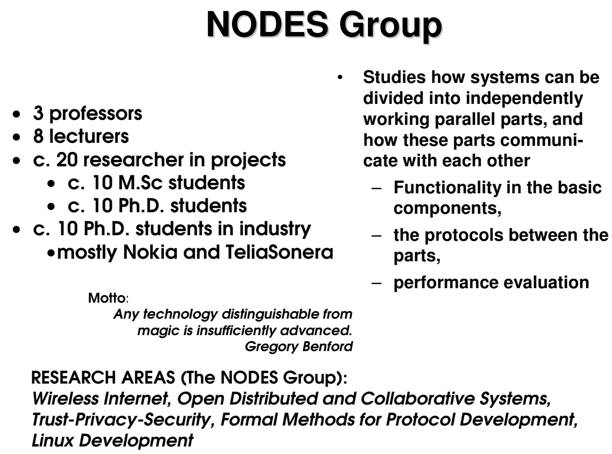


Figure 8.1: NODES Mission Slide

and communication is not ubiquitous, that is, invisible, we have someting to improve. The same is perhaps also true with the high-level research focus of *"how systems can be divided into independently working parallel parts and how these parts communicate with each other."* However, we need to elaborate the research topics of *functionality in the basic components*, *the protocols between the parts*, and *performance evaluation* so that our mission could guide our future research projects.

Until today the NODES Group has arranged and positioned itself by research areas (see Figure 8.2) that have one-to-one mapping to our subgroups. For a while this was fine since the NODES Group was small enough. However, during the last couple of years we have met the co-ordination challenges: *Who is doing what.* In a team of around fifty persons everybody cannot know what all the others are doing without an internal organisation, at least an informal one.

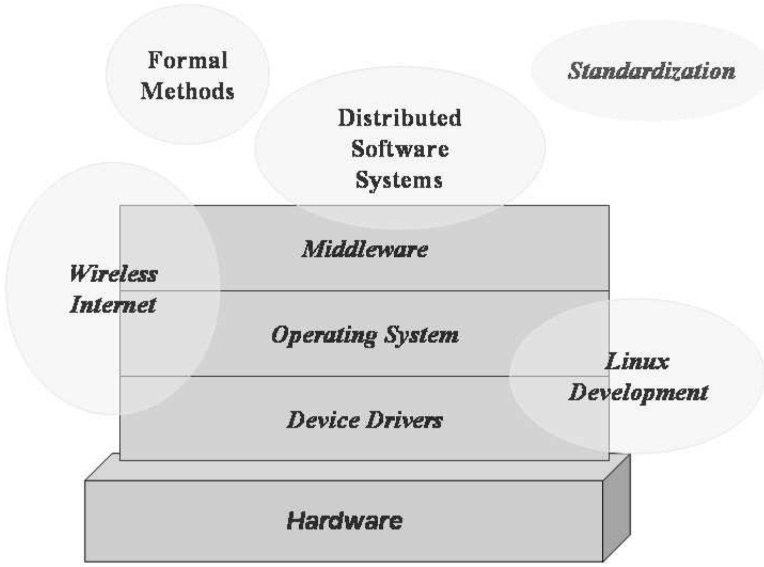


Figure 8.2: NODES Research Areas

8.2 Reconsiderations

In reconsideration we have taken into account various mission papers[2]. These visions can be summarized as *future applications will be context-sensitive, adaptive and personalized, and the systems will be reconfigurable*.

Until recent years, the computing and communication have been mainly driven by technology. Engineers have developed technologies that consumers have figured out how to use. It has worked, we have got positive suprises, but this path is close to an end. We believe that in the next ten years we must move from gadget (and technology) centricity to user-centricity. Producers want to be sure that their products will sell well enough.

The studies of end-user expectations set the requirements of infrastructure

[2]M. Weiser: The Computer for the Twenty-First Century. *Scientific American*, September 1991, pp. 94–104. M. Weiser: Some Computer Science Issues in Ubiquitous Computing. *Communications of the ACM*, July 1993, pp. 74–84. R. Bagrodia, W.W. Chu and L. Kleinrock: Vision, Issues, and Architecture for Nomadic Computing. *IEEE Personal Communications*, December 1995, pp. 14–27. M. Satyanarayanan: Pervasive Computing: Vision and Challenges. *IEEE Personal Communications*, August 2001, pp. 10–17. K. Ducatel *et al.*: Scenarios for Amb ient Intelligence in 2010. Technical report, ISTAG, February 2001. G. Banavar, J. Barton, N. Davies and K. Raatikainen: Special feature on middleware for mobile & pervasive computing. *ACM SIGMOBILE Mobile Computing and Communications Review*, October 2002. R. Tafazolli (ed.): *Technologies for the Wireless Future*. Wiley, 2005. K. Geish: Midleware Challenges Ahead. *IEEE Computer*, June 2001, pp. 24-31.

research. However, this is not a one-way road. The feedback loop gives cost estimates: how expensive certain features are when deployed locally, nation-wide, world-wide. In the forthcoming years we need to take a radically new attitude in order to realise Mark Wiser's vision of ubiquitous computing[3]: the computing and communication is here but you do not need to bother.

In bringing the dream of invisible computing into reality for mass markets, that is for hundreds of millions of people, we still have a lot to do. In fact our claim is that we must go back to the fundamentals; to reconsider the foundations of mobile computing and communications.

The need of this reconsideration has its roots in the fundamental changes in usage patterns. Communication and computing devices move; users move and change their devices; (sub)networks in cars, trains and airplanes move; software moves from one execution environment to another.

All of us hope that we can reuse existing software technology as far as possible. It saves us a lot of money now but may in the future turn out to be extremely expensive. Perhaps we should adapt the slogan used by Queen Beatrice of the Netherlands: *"The natural resources are not a heritage from our parents but a loan from our children."* In mobile computing and communication the *natural resources* should be replaced by the *human-made artifacts*. The attitude will move the short-term focus from the current state we have to the future stage we leave behind us.

If we are not ready for a revolution all the time, we may miss the train and we may find ourselves in the trap of basing next releases of our products on existing legacy. We do not claim that today is the right time to forget all legacy systems. However, tomorrow it is even more costly to replace them. We should ask ourselves whether or not we want to produce pullovers for dinosaurs although the climate has already started to cool and sooner or later the dinosaurs will disappear.

Based on the visions we have formulated three fundamental questions:

1. Do the current operating systems support transparent and seamless reconfigurability?

2. Are the current programming models and tools adequate for context-aware applications?

3. Do the current middleware solutions support development of such applications?

[3] M. Weiser: The Computer for the Twenty-First Century. *Scientific American*, September 1991, pp. 94–104.

Unfortunately the answer to each of the three questions above is negative. Our current legacy base does not support adaptive, context-sensitive, personalized applications.

The operating systems claim to support plug and play functionality. However, more than often you end up rebooting your system which can take several minutes. Current programming models assume that exceptions are rare and independent from each other. However, the usual situation in reconfigurable systems will be that the situations currently regarded as exceptions will be the usual case. Hardly ever will you have the “full system” available; some parts will be unavailable, some parts will be only partially available. The current mainstream of middleware is based on the object-oriented client/server paradigm. However, the emerging personal networking requires a different paradigm that takes event-based middleware and reflective middleware into account.

8.3 New premises

Below is an attempt to extract the essential features needed in successful applications for tomorrow. In some form they can be found in the technological vision papers and recent sociological studies of the so-called information society.

Forget end user terminals! For too long a time we have concentrated on integrated end user terminals such as desktops, laptops, PDAs, and smart mobile phones. We must start to examine reconfigurable end user systems that dynamically change their configuration according to user preferences and available nearby resources.

Stop thinking of users in isolation! Human beinga are still social animals. People want to be involved in various kinds of communities. Lonely hackers belong to the past. Ultimate individualism has strong roots in western ‘civilization’, but here we have a difficult dilemma. If you cannot show others that you are an individual, are you really an individual? In other words, in order to be an individual, a person needs to have a social context, in which to demonstrate difference from other persons.

Users want to be involved! Most laymen users are afraid that they are controlled by the devices and systems. Instead, humans want to be involved but not answering stupid yes/no questions, imbecile confirmations. The users want to have the feeling that they are in control and that the systems only ask them what to do when really needed. On the other hand, if the system takes wrong actions, most users stop using it and drop the gadget in the nearest wastebasket.

Devices are for the users! All engineers should remember that devices are for users, to augment their capabilities, to assist them. User-centric design of devices and user interfaces has progressed a lot. This is mainly due to the involvement of people from disciplines outside EE/CS. In engineering disciplines we must emphasize co-operation with other disciplines.

People are different! We, in the so-called western world, should very quickly drop the idea of a homogeneous user base. People and their expectations are different in different parts of the world. We have a lot to learn from different cultures. The challenge is how to take differences into account, how to manage them, how to do mass-market scale personalization.

Trust is King! Mobile and ubiquitous services and devices are becoming extensions of users and their environments. Thus the system must be safe and trustworthy. Systems that compromise the trustworthiness abuse our right to privacy and will be considered an insult and a fraud. It is time for a reality check: it does not pay to play Russian roulette with computer security.

8.4 Environmental expectations

Applications are not any more to be considered in isolation. Instead, the context and process of use, percieved by users, should be taken into account. Furthermore, the application environment and context includes those peer services the application needs for providing its user the expected service.

Research in the NODES Group has addressed various levels of middleware services, and this trend is still valid. However, we need to understand the role of various middleware layers as selectable aspects in the overall, global computing infrastructure. Figure 8.3[4] illustrates a commonly used hierarchy of different middleware types.

Depending on the application domain or type, the topmost layers change in character: general application services get introduced. In addition, we can split the stack by application type: applications in enterprise context, applications in mobile environment, applications with quality of service requirements, context-sensitive adaptive applications. Each of these types have their special focus areas in the infrastructure technology. For example, in inter-enterprise collaborations it is important to consider business process level interoperability and information representation consistency. The effective solutions for each application area problems may span multiple layers of middleware.

[4] Douglas C. Schmidt, Middleware for real-time and embedded systems, *Communications of the ACM* 45,6 (June 2002) pp. 43–48.

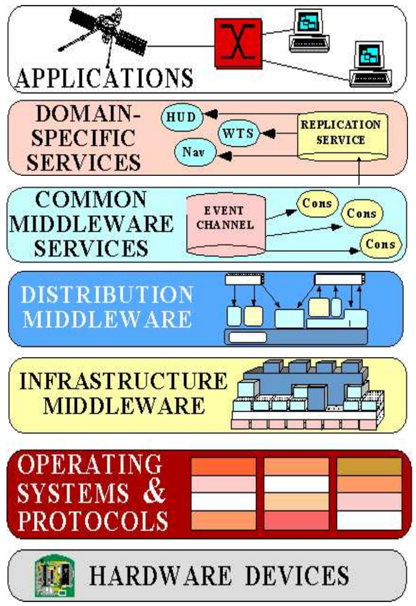


Figure 8.3: Layers of middleware. [D. Smith]

Furthermore, providing a specific application layer service may require features from multiple types, for example mobile e-Commerce applications require aspects of inter-enterprise support and mobility.

We need to understand also the role of middleware services in relationship of the process of service production process. This role can be visualized as in Figure 8.4[5] that relates software engineering, tool development, and more theoretical computer science.

The current trend is towards peer-to-peer computing (in contrast to client-server paradigm), and composition of autonomous services and service orientation (that hides internal implementation of applications providing the service). In composition, control of choreographies between elements is fundamental, and expected to have general support services that allow cost-effective management of changes and evolution of elements. Therefore, a revolution of the distribution management architecture infrastructure has already presented itself, and the areas of tools, platforms, and products need to reflect this in a harmonious way. How-

[5]Figure is adapted from Ilkka Haikala who draw it on the blackboard as Ph.D. thesis opponent in 2000.

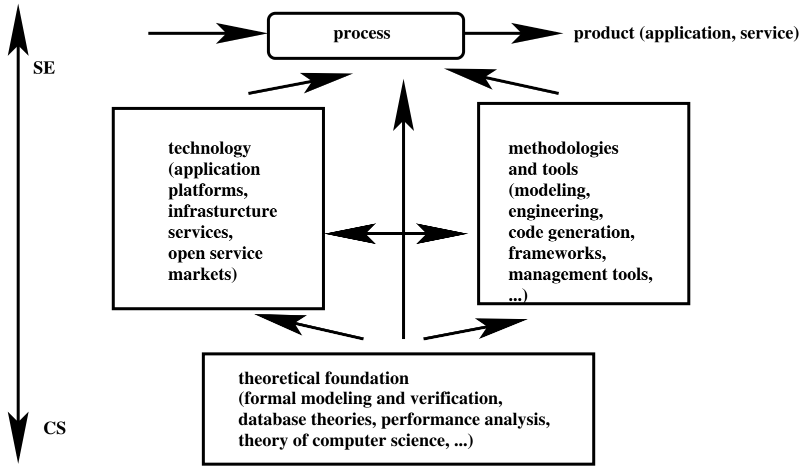


Figure 8.4: Relationship of software engineering, tools, and products.

ever, the new tools and concepts need to be initially constructed using technology and tools designed for old paradigms. Thus, the new vision generate related research and development challenges on all related areas: software engineering, platforms, application architectures, and theoretical background.

8.5 Research challenges

In reformulating the mission statement of the NODES Group we ended up in a research challenge matrix shown in Figure 8.5. On the rows there are six thematic topics: Reconfigurability, Security–Trust–Privacy, Context Awareness, Mobility, Efficient Always-On Connectivity, and Interoperability. These topics are, by no means, independent in the sense that one issue can appear in more than one topic. On the columns there are four layers of software, of which three belong to the scope of the NODES Group: Operating Systems, Internet Protocols, and Middleware. In addition, there is the cross-issue of software engineering, programming paradigms and models that are out of the scope of the NODES Group, but that are necessary in building an operational system.

It should be noted that our list of thematic topics is not the only possible one. The current list is derived from the assumption that the future applications must be context-sensitive, adaptive and personalized, and that the future systems must be reconfigurable. We have also discussed about other thematic topics: *distribution*, *autonomous decisions*, *autonomous systems*, *resource awareness*, *self-awareness*, *proactivity*, *scalability*, *evolvability*, *delegation*, and *incremental deployment*. Al-

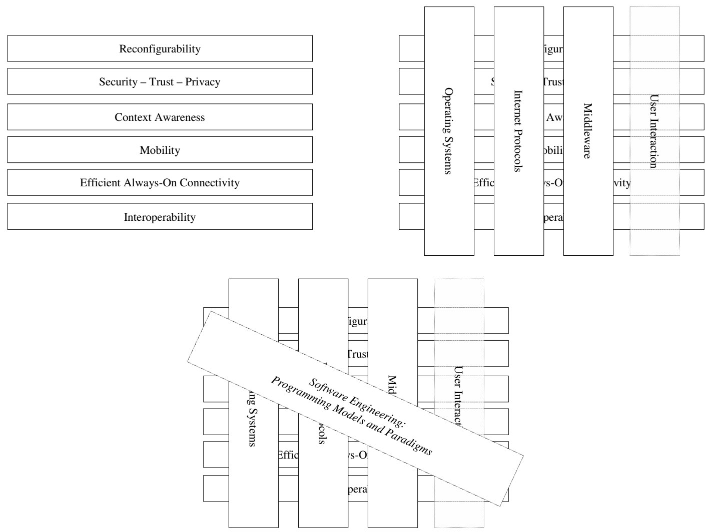


Figure 8.5: NODES Research Challenge Matrix

ternative lists of research challenges can also be found in the vision papers[2].

Peter J. Denning[6] has divided the computing into 30 Core Technologies[7], Computing Practices (*programming*, *engineering systems*, *modeling*, *innovating*, *applying*), Design (*simplicity*, *performance*, *reliability*, *evolvability*, *security*), and Mechanisms (*computation*, *communication*, *coordination*, *automation*, *recollection*); see Figure 8.6.

The reason why we ended up in a matrix is the fact that in order to address a thematic topic, reconfigurability, for example, you will need support on each software layer. Some of the thematical topics are discussed in the *Mowgli Jubilee Book*[8]. Below we briefly summarize the topics:

Reconfigurability: The end-user devices of today are primarily integrated units like PDAs, laptops, or mobile phones. However, the situation will change in the future. A personal trusted device, we call it *FuturePhone*, will be the

[6] P. J. Denning: Great Principles of Computing. Communications of the ACM 46, 11 (Nov. 2003): 15–20.

[7] algorithms, artificial intelligence, compilers, computational science, computer architecture, data mining, data security, data structures, databases, decision support systems, distributed computation, e-commerce, graphics, human-computer interaction, information retrieval, management information systems, natural-language processing, networks, operating systems, parallel computation, programming languages, real-time systems, robots, scientific computation, software engineering, supercomputers, virtual reality, vision, visualization, workflow

[8] Kimmo Raatikainen: *Wireless Internet: Challenges and Solutions—First Ten Years of Research in Mobile Computing at Helsinki University Computer Science Department and Challenges for the Next Ones.* Report B-2004-3, Helsinki University Computer Science Department.

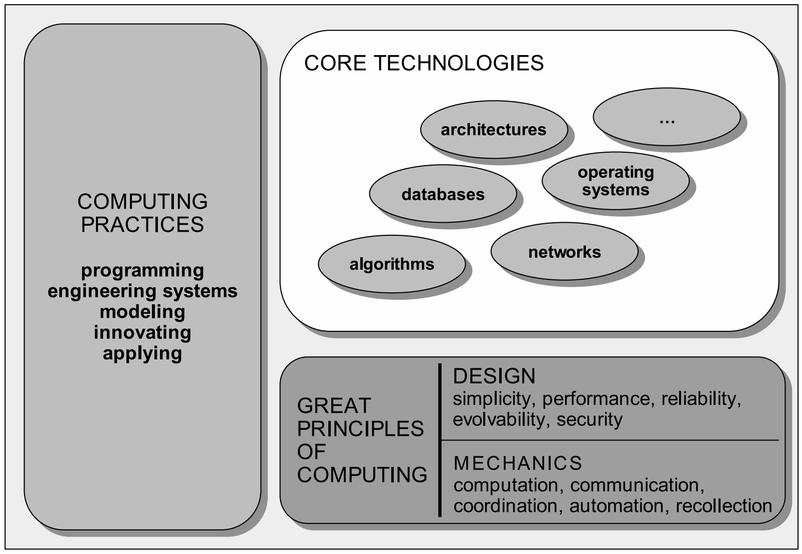


Figure 8.6: Principles-based potrait of computing [P. Denning]

core of the personal networking system. It probes its surrounding looking for suitable peripheral devices such as displays, input devices, processors, fast access memories and access points to communication channels. It dynamically builds up the most appropriate end-user system that can be auto-configured. The *FuturePhone* also probes for other similar devices in order to establish suitable ad-hoc communities and different kinds of sensors in order to extract context information associated to the current smart place. The *FuturePhone* also tries to detect actuators which provide the means to affect properties of the smart place.

Security–Trust–Privacy: The success of the future Internet will totally depend on consumers' trust. The current Internet is vulnerable to worms, viruses, spam and fraud. It clearly demonstrates that *"fix it later"* does not work. Security, trust and privacy must be addressed from the very beginning of system design and on all levels: hardware, operating system, protocols, middleware. Moreover, trust is not of type on-off. Different tasks need different levels of trust. In addition, different persons require different levels of trust for the same task.

Context Awareness: Almost any piece of information available at the time of interaction can be seen as context information. Context-awareness means that one is able to use context information. A system is context-aware if it can extract, interpret and use context information and adapt its functionality to the current context of use. The grand challenge is to create a flexible context modeling framework. The objective is to have efficient means of presenting, maintaining, sharing, protecting, reasoning, and querying context information. It should be noted that different applications in use typically have different views of context information. Therefore, also distributed data management is an essential enabler for context-aware applications.

Mobility: Current 2.5G and 3G networks adequately support terminal mobility for voice services. The Session Initiation Protocol (SIP) solves most of the challenges in user mobility. Mobile IPv6 can handle global mobility, and there are several proposals to support local mobility. The challenges in mobility that remain to be solved include (sub)network mobility (hundreds of active users on a fast train, for example), seamless mobility preserving network Quality-of-Service, middleware and application layer support for load-balancing and efficient use of resources.

Efficient Always-On Connectivity: Despite recent developments, we still have a lot to do in communication protocols. On the link layer we should examine adaptation to varying physical conditions. On the network layer we need seamless co-operation between mobility mechanisms, Quality-of-Service solutions, and security. On the transport layer we must find solutions for multiple simultaneous flows and different transport protocols. On the presentation layer we need a wire format that is efficient to process and does not consume a lot of bandwidth. On the application layer we need conversation and dialog protocols that can reasonably cope with long latencies. In addition, we must also address cross-layer interferences and co-operation. In summary, we must have a reasonable solution on each layer since the performance can be destroyed on each layer.

Interoperability: Users become more and more exposed to the use of remote and remotely administered services. In simple cases, the locally running user interface application helps to communicate with external services; in more complicated cases, complete business processes are participated with multiple peers. Interoperability, or capability to collaborate, means effective capability of mutual communication of information, proposals and commitments, requests and results. Interoperability covers technical, semantic, and pragmatic interoperability. Technical interoperability means that messages can be transported from one participant to another. Semantic interoperability means that the message content becomes understood in the same way by the senders and the receivers. This may require transformations of information representation or messaging sequences. Finally, the pragmatic interoperability captures the willingness of partners for the actions necessary for the collaboration. The willingness to participate involves both capability of performing a requested action, and policies dictating whether the potential action is preferable for the enterprise to be involved in. An aspect of special interest is the trust between collaborating partners (this is different from the trust on Internet infrastructure addressed above). In the pragmatic view, process-awareness in terms of collaborative business process model is needed, augmented with non-functional aspects, some of which are related to business policies. For middleware that supports interoperability, the challenges are found on areas of modeling, software production, and runtime system with middleware/infrastructure services. Special attention has to be placed on supporting independent evolution of services at application level and separating control/management facilities from the application components.

8.6 Postscript

Based on the high-level descriptions of the research challenges above, we will reformulate the NODES mission and produce more concrete research position statements to guide our research. It must be noted that—even with our current team of fifty—we cannot address all relevant issues in the six themes and three software layers. Nevertheless, our research will remain experimental: **we design, model, implement, evaluate, and analyze.**

As experimental scientists we must keep it in our minds that *"it works"* is neither evaluation nor analysis. It is not *science* to assemble parts to "see what happens." Years ago Denning[9] stated three common misconceptions of experimental computer science:

1. It is not novel to repeat an experiment.
2. Mathematics is the antithesis of experiment.
3. Tinkering[10] is experimental science.

For some reason these misconceptions are unfortunately still quite common. In particular, "hacking" is not experimental computer science: it may improve the personal knowledge of the hacker, but it does not contribute to our sum of knowledge. *If we do not live up to the traditional standards of science, there will come a time when no one takes us seriously.*

[9] P. J. Denning: What is Experimental Computer Science? *Communications of the ACM* 23, 10 (Oct. 1980), pp. 543–544.

[10] In computing we use the word *hacking* rather than *tinkering*.

We believe in running code, but our implementations are primarily for evaluating the research results and assessing the implementability of our ideas. However, we also need to produce product-quality open source software so that future research can be based on our previous results.

International standardization will remain in our focus. In addition to scientific articles and Ph.D. theses, contributions to international standards, particularly in IETF and W3C, measure our impact. In the distributed wireless world research results that do not become formal or defacto standards do not have any direct impact of practical importance.

Chapter A

NODES Theses 1995–2004

A.1 Ph.D. Theses

Kaivola, Roope: Equivalences, preorders and compositional verification for linear time temporal logic and concurrent systems. (Report A-1996-1).

Kutvonen, Lea: Trading services in open distributed environments. (Report A-1998-2).

Kähkipuro, Lea: Performance modeling framework for CORBA based distributed systems. (Report A-2000-3).

Karvi, Timo: Partially defined Lotos specifications and their refinement calculus. (Report A-2000-5).

Lindström, Jan: Optimistic concurrency control methods for real-time database systems. (Report A-2003-1).

Helin, Heikki: Supporting nomadic agent-based applications in the FIPA agent architecture. (Report A-2003-2).

Campadello, Stefano: Middleware infrastructure for distributed mobile applications. (Report A-2003-3).

Taina, Juha: Design and analysis of a distributed database architecture for IN/GSM data. (Report A-2003-4).

Luukkainen, Matti: A process algebraic reduction strategy for automata theoretic verification of untimed and timed concurrent systems. (Report A-2003-7).

Manner, Jukka: Provision of Quality of Service in IP-based mobile access networks. (Report A-2003-8).

Gurtov, Andrei: Efficient data transport in wireless overlay networks. (Report A-2004-1).

A.2 Ph.Lic. Theses

Kutvonen, Lea: The Role of Trading Function in Open Distributed Processing Infrastructure. (C-1996-84)

Taina, Juha: Database Architecture for Intelligent Networks. (C-1997-50)

Kähkipuro, Pekka: Object-oriented middleware for distributed systems. (C-1998-43)

Laamanen, Heimo: Serveability issues in mobile distributed systems. (C-1998-44)

Luukkainen, Matti: Timed semantics of concurrent systems. (C-2000-4)

Lindström, Jan: Optimistic concurrency control methods for real-time database systems. (C-2001-9)

Helin, Heikki: Supporting nomadic agent-based applications in FIPA agent architecture. (C-2001-63)

Manner, Jukka: An IP-based mobile and QoS-aware network architecture. (C-2002-13)

Miettinen, Kari: Key management and trust in 3G networks. (C-2002-25)

Gurtov, Andrei: Efficient transport in 2.5G3G wireless wide area networks. (C-2002-43)

Niklander, Tiina: Fault tolerance in real-time main-memory databases. (C-2003-23)

A.3 Selection of M.Sc. Theses

Laudatur Grade

Liljeberg, Mika: Performance of World-Wide Web Access over a Cellular Telephone Network, 1996.

Torvalds, Linus: Linux: A Portable Operating System, 1997.

Colussi, Gian Donato: Equation-Based Layered Multicast Congestion Control, 2004.

In addition, Pauli Misikangas (*Suunnittelumallien automaattinen tunnistaminen olio-ohjelmista*, 1998) and Vilho Raatikka (*Cache-conscious index structures*, 2004) have been working in NODES group. Their thesis of laudatur grade, however, were done for sections of software engineering and information systems, respectively.

1995

Glad, Lauri: Prosessin suorituksenaikainen siirto hajautetussa järjestelmässä

Kojo, Markku: Etäproseduurikutsu heterogeenisessa järjestelmässä

Kokkola, Tommi: Analysis and measurements of the performance characteristics of GMD data services

Lapinlampi, Jarkko: Sirtymäsysteemin tilatehokas minimointi

Viljanen, Lea: Kansainvälisen sähköpostiverkon hakemistopalvelu

1996

Helin, Heikki: Mowgli-järjestelmän geneeristen tapahtumien kuvauskieli

Huttegger, Nina: Lyhytsanomapalvelut Mowgli-arkkitehtuurissa

Kiiskinen, Jani: Langattomalle tietoliikenneyhteydelle soveltuva protokolla ja sen suorituskyky verrattuna TCP/IP-protokollaan

Koskimies, Olli Oskari: Agentti-sovelluskehys Mowgli-arkkitehtuuriin

Lappalainen, Matti Petteri: Prosessin suoritusaikainen siirto varmennetussa tosiaikajärjestelmässä

Luukkainen, Matti: Prosessialgebrojen soveltaminen tietoturvallisuuden mallinnukseen ja verifiointiin

Mäenpää, Jari: Data communication and application protocols of a mail user agent in a GSM mobile host

Ollikainen, Jussi: TINA-kommunikaatioistunnon toteutus ATM-verkossa

Sivonen, Timo: Security management on distributed systems

1997

Björklund, Leif: Mobile Internet Packet Routing with Cellular Networks

Hamberg, Max: ODP-viitemallin näkökulmakuvaukset

Karhu, Vesa: Sanomanvälitysalgoritmien verifiointi käsin temporaalilogiikan avulla

Kaskenpalo, Mika-Petteri: Mowgli-arkkitehtuurin tietoturva

Lavonius, Ville: POSIX/DMX -yhteiskäytön toteutus DX200-järjestelmässä

Lilja, Sami: DX200-puhelinkeskustietokoneen ohjelmiston sovittaminen käyttämään Chorus-mikroydintä

Puuskari, Mikko: Standardization Framework for Slow Wireless Link Protocols in an Indirect Architecture

Raja, Marita: Mowgli-yhdyssolmun hallintajärjestelmä

Surakka, Janne-Pekka: Globaali postilaatikkojärjestelmä älypuhelimille

Turunen, Katri: Virtuaalimuistin käyttö reaaliaikaisessa hajautetussa muistitiedostojärjestelmässä

1998

Britschgi, Juhana: Radioresurssien hallintaprotokollan suunnittelu UMTS-järjestelmään

Korhonen, Jouni: Sähköposti liikkuvassa järjestelmässä

Muhonen, Ahti: Point-To-Point Services in General Packet Radio Service

Mäkelä, Tero: Charging and Billing in GPRS

Rönkkö, Minna: Puhelun tunnistamisen hallinta radiopuhelinverkossa

Sevanto, Jarkko: Control and decision making in a mobile workstation

Stenman, Jorma: Paradigman muutos DX200-ohjelmistotuotannossa

1999

Ahonen, Mikko: CORBA Mowgli-ympäristössä

Halme, Jussi Petteri: Yrityksen suljetut tietoverkot

Hurtta, Tuija: The Functionality of Packet Data Access Node in Future Wireless Packet Data Networks

Kauppinen, Tero: IP over Bluetooth

Koskelainen, Juha: DHCP ja se toteutus A2-alustalla

Manner, Jukka: TCP over GPRS: Performance Analysis

Meskauskas, Paulius: Mobile Agent-based Intelligent Network Environment

Nyman, Ran: Pistokepohjaisten järjestelmien siirtäminen avoimiin hajautettuihin järjestelmiin (DCOM, CORBA ja RMI)

Pulli, Harri Juhani: Mowgli-arkkitehtuurin tietoturvan toteutus Secure Shell -järjestelmää käyttäen

Stenberg, Markus: Evaluation of Communication Interfaces for Distributed Systems

Tuomela, Frans: Protocols of Media Gateway Controller

Tykkälä, Kimmo: FIPA ACL -kommunikoinnin optimointi liikkuville työasemille

2000

Boström, Jani: Providing value-added services for corporate users in 3G networks

Gurtov, Andrei: TCP Performance in the Presence of Congestion and Corruption Losses

Leppänen, Sari: SDL-kuvausten kompositionaalinen verifiointi

Luotola, Jari: Internet-multimedianeuvottelun hallinta

Martikainen, Tommi: Quality of Service in Internet Protocol Suite for Mobile Terminals

Månsson, Jani: Location-Based Services in Wireless Local Area Networks

Mäkinen, Juha Tuomas: Käyttöjärjestelmäriippumaton oliopohjainen hajautettu palveluntarjonta tietoverkoissa

Nybergh, Rasmus: Interconnection Networks for DX200

Tarkoma, Sasu: User Dialoque Management in the FIPA Architecture

Tuominen, Marja: Käänteistekniikkojen käyttö reaaliaikaisessa hajautetussa järjestelmässä

Vuorento, Jussi: The Effects of Power Control In Bluetooth Networks

Vuorinen, Eeva: The impact of XML in e-commerce

2001

Hyytinen, Meri: Avoimet liitännät DXT:hen

Kräkin, Toni: Komponenttiviitemalli ohjelmistotuotteen teknologiastrategian ohjausvälineenä

Kuhlberg, Panu: Effects of Delays and Packet Drops on TCP-based Wireless Data Communication

Laukkanen, Jussi: An Evaluation of IPv6 Transition Mechanisms in Implementation of UMTS Internet Access

Pennanen, Mika Matti Tapani: Agents in Virtual Home Environment

Rinta-Aho, Teemu: Anonymous Mobile Internet Access

Sarolahti, Pasi: Performance Analysis of TCP Enhancements for Congested Reliable Wireless Links

Tolvanen, Jarkko: Device security

Tolvanen, Mika: X.509-sertifikaattien elinkaari ja sertifikaattien hallintaprotokollat

2002

Björklund, Jon Patrik: Adaptive Load Balancing in a Cluster Computer

Haakana, Jaakko: A statically translating instruction level simulator for ARM4 Architecture

Hannula, Tero: Paikallisen tietopalvelun toteuttaminen Bluetooth-tekniikkaa ja WAP-protokollaa käyttäen

Koskinen, Aaro: Menetelmä hajautettujen järjestelmien mallinnukseen ja suorituskyvyn analysointiin

Ponka, Ilja: Ohjelmistoagentit verkkokaupassa

Riihimäki, Juha: Meklaripalvelun reaaliaikavaatimukset

2003

El-Khouri, George: Load balancing for Location Requests in GSM Location Systems

Kulve, Tuomas: Analysis of Concurrent TCP and Streaming Traffic over a Wireless Link

Löytynoja, Jenni: Järjestelmäintegraatio ja sen mahdollisuudet yritys A:ssa

Peri, Indrek: IP QoS in Ad Hoc Wireless Networks

Saarto, Jarno: WWW Traffic Performance in Wireless Environment

Vähäaho, Markku: Arkkitehtuurikuvauksia hyödyntävä meklaus.

2004

Haavisto, Juhani: Palveluhakemistojen käyttäjien yksityisyyden tarpeet ja niiden hallinta

Keränen, Ari: Luottamuksen hallinta internet-verkoissa

Laamanen, Topi: Julkisen hallinnon sähköiset yhteispalvelut

Ojanen, Jouni: Adaptoituva rich client -mobiilisovellusarkkitehtuuri .NET-ympäristössä

Rantala, Taneli: FIFO-piirin spesifiointi ja verifiointi PVS-teoreeman todistajalla

Ruokolainen, Toni: Komponenttien yhteensopivuuden tarkistaminen protokollatasolla

Tapola, Lassi: Selvitys kuluttajille suunnatuista Internet-maksujärjestelmistä Suomessa