Process Support for Mobile Work across Heterogeneous Systems

Alf Inge Wang* and Liu Chunnian†

April 4, 2001

Abstract

The emerging field of *mobile computing (MC)* studies systems in which computational components may change locations. In terms of hardware, mobile work is usually across heterogeneous systems in Web extended by novel mobile devices. In terms of software, mobile work technically involves mobile agents and new generation of middleware. However, in general mobile work presents a new challenge and great opportunities to research in software engineering as a whole. In this paper, we focus our attention on process support for mobile work, that is an important aspect of software engineering. We present a classification and characterisation of mobile work mainly from the process point of view, and specify the requirements of process support for mobile work. The last part of the paper compares three process centred environments in regards to mobility support, and identifies their shortcomings.

Keywords: *Mobile Computing, Heterogeneous Web-based Systems, Mobile Agents, Middleware, Software Processes*

1 Introduction

In the recent years, mobile computing has gained more and more attention. New technology has made it possible for people to move around on different locations while working and parts of the software system change location during execution. Traditionally, portable PCs have been the working environment for mobile work, but smaller computational devices such as personal data assistants (PDAs) and mobile phones are becoming increasingly important as working environments. In this mobile world, it is necessary to identify the new requirements mobility introduces to process support needed in software engineering. This paper classifies and characterises mobile work, identifies some

*Dept. of Computer and Information Science, Norwegian University of Science and Technology (NTNU), N-7491 Trondheim, Norway, Phone: +47 73594485, Fax: +47 73594466, Email: alfw@idi.ntnu.no, Web: http://www.idi.ntnu.no/∼alfw

†Beijing Polytechnic University (BPU), Beijing, P.R. China, Email: ai@bjpu.edu.cn. Chunnian Liu's work was supported in part by the Natural Science Foundation of China (NSFC), Beijing Municipal Natural Science Foundation (BMNSF), and the 863 High-Tech Program of China, as well as by NTNU, Norway.

requirements needed to provide process support for mobile computing, and investigates how these requirements are fulfilled in existing process centred environments.

The paper is organised as the following: Section 2 defines and describes mobile work, and discusses different kinds of mobile work. Section 3 identifies requirements to provide process support for mobile work. Section 4 describes briefly three process centred environments and compares their ability to provide support for mobile work requirements. Section 5 summarises and concludes the paper.

2 Classification and Characterisation of Mobile Work

Mobile work refers to systems in which computational components may change locations [2]. Such systems are extensions to more traditional Web-based distributed systems due to the rapid development of component miniaturisation; high-speed wireless communication, as well as mobile software agent technology. Mobile work usually needs information sharing and activity coordination, so it can be also regarded as a novel kind of **computer-supported cooperative work (CSCW)** [3].

In traditional CSCW research community, mobile software agents have received a lot of attention, where agents move along a fixed network structure. The new issues in mobile work are the moving hardware, and hence the moving of software agents in dynamic networks. The new research project MOWAHS at IDI/NTNU [1], as a successor of the CAGIS project on agent-based CSCW [6, 9] also conducted at IDI/NTNU, addresses the new research issues raised by mobile work.

There are various kinds of mobile work, and new form of mobile work continues to emerge every day. Here we try to categorise different kinds of mobile work, so that we can proceed in the next section with the process support for each different kind of mobile work.

In [2], physical and logical mobilities are distinguished on a coarse level. From a more practical point of view, we can recognise the following different kinds of mobile work:

- **Hardware Mobility**: Computational hardware moves in physical space. We can further recognise the following sub-categories:
  - *Fixed Network*: Here we have the hardware infrastructure for traditional Web-based CSCW.
  - *Mobile Hardware Attached to Fixed Network*: Here we have a fixed core network and a fringe of mobile hardware. The latter can be connected (via base stations) to the core network, or disconnected (working alone). We can think out a lot of examples of this kind of mobile work: Notebook computers used in travel; Cellular telephone with limited Internet access; PDAs with built-in wireless capabilities; Global positioning systems such as information kiosks connected to a fixed network, providing tourist information to cars over a low power wireless link.

  - *Temporary Network*: Here we have a network consisting of mobile hardware exclusively and existing in a short time for a particular task. A good example is that at a conference where every participant has a laptop. These laptops can form a temporary network to facilitate conference discussion via wireless communication. After the conference, such a network will stop existing.

- **Software Mobility on a Fixed Network**: Computational processes migrate inside a fixed network. This kind of mobile work includes code on demand, remote evaluation, and more recently, mobile agents. Software mobility on a fixed network has been studied for years, and the underlying assumption has been the hardware stability. A consequence of this assumption is that the mobility space reflects directly the structure of the underlying network.

- **Combined Mobility**: This represents the current trend of mobility, allowing mobile software to migrate among both mobile and stationary hardware. In this case, the mobility space is more complicated. A good example is the remote access to database (or email server): a mobile agent moving back and forth between a (fixed) email server and a mobile notebook computer that is disconnected and reconnected to the server from time to time. Even when the connection is cut off, the user of the notebook still can issue queries (or send emails), the agent residing on the notebook computer collects the queries (or the sent emails); When the connection is resumed, the agent moves to the database server to execute the queries (or sends the out-going emails and receives the coming emails), while the notebook can be disconnected; Finally, when the connection is reestablished again, the agent will move back to the notebook presenting the query results (or the received emails).

3 The Requirements of Process Support to Mobile Computing

In the previous section, we distinguish hardware mobility from software mobility. Note that from the software engineering point of view, the two kinds of mobility present quite different perspectives. Hardware mobility, being possible due to component miniaturisation and high-speed wireless communication, presents end-users new opportunities, advantages and convenience in their professional work and daily life. But for software engineers of distributed applications, it represents a very challenging, new target execution environment. On the other hand, software mobility may be transparent to end-users, while giving a new design tool for the developers of distributed applications. The aim of the research in software engineering for mobility is to develop models; techniques and tools for this new generation of distributed systems that fulfil the requirements of the changed execution environments posed by hardware mobility. Software mobility can be regarded as the weapon in the research.

Various branches of software engineering are effected by mobility (formal modelling, algorithms, security, etc.). In this paper, we focus on the process aspect of software engineering. We would like to outline the requirements of process support for mobile work.

A *software process* is the total set of software engineering activities needed to transform a user's requirements into functioning software. A *software process model* is a framework providing the definitions, structures, standards, and relationships of the various process elements so that common technology, methods and measurements can be applied by any software project.

In general, all software development projects need process support in terms of process modelling, process architecture, and process support tools. If the development is a co-operative effort (CSCW), the process become more advanced and complicated, involving new issues such as group awareness, multi-user interfaces, concurrency control, communication and coordination within the group, shared information space and the support of a heterogeneous, open environment which integrates existing single-user applications. Finally, Mobile Work can be regarded as a novel type of CSCW: while information sharing and activity coordination are also needed as in general CSCW, it opens a new dimension of mobility of both hardware and software. Thus, the process support for mobile work will be even more complicated.

In the following, we will outline the requirement of process support to mobile work. The process support will be discussed in terms of process modelling, process architecture, and process support tools. For each category, we will first outline the generic requirements for all CSCW, then point out the new challenges concerning mobile work.

1. **Process Modelling**

   1.1 *Generic for all CSCW*: Process models define action flow/events and communication flow/events. We need a process modelling language (PML) to specify the models. Usually the PML allows object-oriented definitions of process elements (activities, artifacts, resources, *etc.*) in a hierarchical manner (through sub-typing and aggregation, *etc.*). Process models are the basis for any systematic process understanding, analysis and enaction. Without models, the process support can only be provided here and there, in an ad-hoc way. Based on the process model, process elicitation, process planning/scheduling/enaction can be done systematically in general or even automatically (for strict process).

   1.2 *Specific for Mobile Work*: Here in addition to the generic process modelling, we need to model mobility. **Mobility Modelling** consists of three main parts:

      1.2.1 **The unit of mobility**: the smallest component in the system that is allowed to move. A natural choice is to make the unit of mobility coincide with the unit of execution. For hardware, a unit of mobility means a physical device, while finer-grained units would be parts of devices and unsuitable to reasonable modelling. For software, a unit of mobility means a mobile agent. Though in practice finer-grained units (such as code on demand) are pervasive, their modelling presents a challenge to research community. Each unit of mobility in the system should have a *state* in mobility modelling.

      1.2.2 **The location**: the position of a mobile unit in space. The type of location is affected by the choice of unit of mobility. For example, locations for

moving devices can be described by their physical coordinates, for mobile agents by the addresses of their hosts, and for code fragments by the process identifiers. Location can be modelled by a variable belonging to the state of a mobile unit.

1.2.3 **Context changes**: Mobile computing is context-aware computing. Conventional computing fosters a static context with no or small, predictable changes, while in mobile computing changes in location may lead to abrupt and unpredictable changes in the context. Note that the location of a mobile unit cannot solely determine the context that the unit perceives, because the context relies on other factors such as administrative domains. Formulation of context and mechanisms for managing context changes are major challenges.
In order to handle context changes, the model must support some *coordination* mechanism that can specify how the unit of mobility interacts with its context separately from the behaviour of the unit itself. Mobility poses some novel challenges to coordination (*e.g.* context varies with high frequency; traditional naming schemes – such as Internet DNS – cannot be used for mobile agents and fluid network).

2. **Process Architecture**

2.1 *Generic for all CSCW*: Process architecture must possess the following features to support CSCW in general:

2.1.1 **Facilitating the creation of a community memory**
The intra-project memory facilitates important features in CSCW, such as group awareness, and change management to support process evolution.

2.1.2 **Supporting distribution of documents/work/resource/...**
The architecture should provide support for transparently distributed people (stakeholders, product users, and SPEGs – Software Process Engineering Groups), artifacts, process objects and execution behaviour. Each component of the architecture (processes, object state and behaviour, interpreter, user interface, etc.) can be accessed and executed by remote users through a standard *http* server and may be physically located on distributed machines throughout the Internet.

2.1.3 **Supporting integration with external tools**
The architecture should support bi-directional communication between internal objects and external tools and services. API-based integration can be supported by handlers (Java applets), allowing bi-directional calls to internal and external interfaces, tools, and services.

2.1.4 **Being platform independent**
To facilitate the installation and execution of a process support system, the architecture should be designed as a set of highly componentised, lightweight, concurrent elements (as Java applets), to be used across different platforms, easily transported across the WWW, or embedded in *html* pages. The architecture should also provide an integrated environment for operation and communication.

2.2 *Specific for Mobile Work*: We have seen from the above list that a lot of software mobility has been involved. However, there are some underlying assumptions: uniform type of client devices, on-line work only in a network, and the stability of the network. With drifting away of computing from traditional computers and hardware mobility, these assumptions do not hold any more, so the process architecture should have the following extra supporting features:

2.2.1 Supporting heterogeneity of computing devices: a workflow process framework and a multi-agent architecture that can run on anything from a desktop computer to a mobile phone.

2.2.2 Supporting mobility, especially the following off-line facilities:

2.2.2.1 off-line workflow capacities for PDAs and mobile phones

2.2.2.2 synchronisation between different devices after working off-line

2.2.2.3 handling workflow changes when devices are off-line

3. **Process Support Tools**

3.1 *Generic for all CSCW*:
The impact of modelling in CSCW is through a set of process support tools including: Tools for group awareness such as a process information system (PIS). With that, individuals can view pending tasks and requests, and find references to supporting information, process forms, *etc.*; Tools for information-sharing and coordination to reduce and solve conflicts occurring in cooperative work; Tools for communication and tools for change management.

3.2 *Specific for Mobile Work*:
The impact of formal modelling in mobile computing is through the development of appropriate middleware that rests above the operating system and provides developers with specialised mechanisms and services in a highly integrated fashion. Middleware specialised for mobility is a new generation of middleware. Hardware mobility and software mobility give rise of different middleware:

3.2.1 **Middleware for hardware mobility**: The tasks for this kind of middleware include: detecting location changes; specifying what belongs to the context of computation; relating location changes to context modifications; determining how the computation itself is affected by the changes in the context.
Note that middleware for hardware mobility should hide its work into the underlying runtime support, so that the hardware mobility can be transparent as much as possible to the developers of distributed application.

3.2.2 **Middleware utilising software mobility**: Middleware was originally invented by the earlier study of software mobility on a fixed network. The more comprehensive concepts of software mobility, as given in this paper, provide higher level of abstraction and flexibility that enable code and state relocation. The next generation of middleware should fully utilise the advantages of software mobility to support varying grains of mobility, different rebinding strategies, and different architectural styles for relocation, all in a single, uniform programming interface.

4 Comparing three PCE in regards to Mobility Support

In this section, we will compare three PCE in regards to mobility support. The three environment compared are the Endeavors workflow system, the ProcessWeb workflow system, and the CAGIS PCE. The next three subsections will briefly describe the three PCEs and they will be compared in regards to the requirements listed in section 3.

4.1 Endeavors

Endeavors is an open, distributed, extensible process execution environment developed at University of California Irvine, and has been licensed by Endeavors Technology Incorporated. It is designed to improve coordination and managerial control of development by allowing flexible definition, modelling, and execution of typical workflow applications. There are five main characteristics for Endeavors:

- **Distribution** Support for transparently distributed people, artifacts, process objects, and execution behaviour (handlers) using web protocols.
- **Integration** Allows bi-directional communication between its internal objects and external tools, objects, and services through its open interfaces across all levels of the architecture. Multiple programming languages are also supported through APIs.
- **Incremental Adoption** Components of the system (user interfaces, interpreters, and editing tools), may be down-loaded as needed, and no explicit system installation is required to view and execute a workflow-style process.
- **Customisation and Reuse** Implemented as a layered virtual machines architecture, and allows object-oriented extensions of the architecture, interfaces, and data formats at each layer.
- **Dynamic Change** Allows dynamic changing of object fields and methods, the ability to dynamically change the object behaviours at runtime, and late binding of resources needed to execute a workflow process. Process interpreters are dynamically created as needed.

The comparison in this paper is based on the University prototype of Endeavors and not the recent commercial tools. More details about the Endeavors can be found in [5].

4.2 ProcessWeb

ProcessWeb [14] is a web-interface based workflow system based on the ProcessWise [8] Integrator (produced by ICL) implemented by Information Process Group, University of Manchester. The web-interface is provided through the ProcessWise Integrator application interface. ProcessWise Integrator creates an environment enabling the activities of

people in an enterprise to be coordinated and integrated with the organisation's computing facilities. A process management system built using the ProcessWise Integrator has a client/server structure and consist of four main components as shown in figure 1: User Interface, Process Control Manager, Process description in PML, and an Application Interface. The most important component of ProcessWise Integrator is the Process Control Manager (process engine), which acts as the central server. Its main function is to interpret the PML description of the process. To ensure that processes may continue indefinitely, the Process Control Manager has been implemented using a persistent store technology.

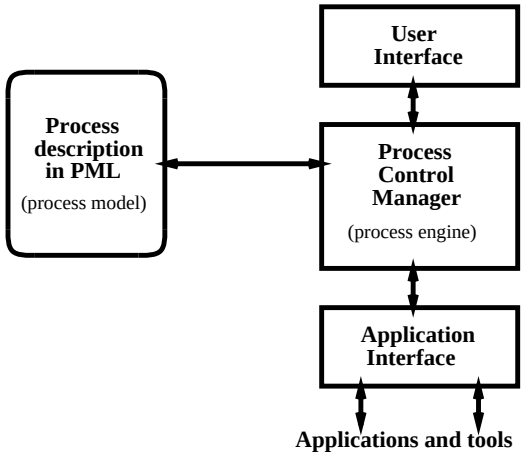

Figure 1: Structure of ProcessWise Integrator

4.3 CAGIS PCE

The CAGIS PCE is designed to model and enact cooperative and individual activities that can be distributed over several workspaces, and consists of three subsystems:

- **CAGIS SimpleProcess** is a workflow tool with a web-interface, used to model processes consisting of networks of activities executed by individuals. For users of CAGIS SimpleProcess, the activities will be shown as web-pages that can contain a work description, links to relevant document or tools, HTML-forms for entering necessary information, or a Java applet. A process in CAGIS SimpleProcess can consist of several autonomous *process fragments* that can be distributed over several workspaces. A process fragment can be anything from one single activity to several related activities. Hierarchical workspaces (i.e., a tree structure) are used model the organisation executing the process, and CAGIS SimpleProcess has support for moving process fragments between workspaces (between users) and between sites. A more detailed description of this workflow tool is found in [10].
- **CAGIS Distributed Intelligent Agent System (DIAS)** takes care of cooperative activities between workspaces in the CAGIS PCE framework [13, 7, 4]. The software agents in CAGIS DIAS can be used to coordinate artifacts and resources, negotiate about artifacts and resources, monitor the working environment for changes

or events, provide infrastructure for brainstorming, electronic meetings, trading services etc. CAGIS DIAS provides the infrastructure for creating cooperating agents, and consists of four main components: Agents, Agent Meeting Places (AMPs), Workspaces, and Repositories.

- **CAGIS GlueServer** To enable CAGIS SimpleProcess to interact with the CAGIS DIAS, we have built a CAGIS GlueServer. The CAGIS GlueServer is a middleware that uses a so-called GlueModel, where relations between process fragments and software agents are defined. The GlueModel can be seen as a part of the process model defining rules for interaction between people or groups of people, but also as a meta-model since it can specify changes of the process model. By using a *separate model* to describe the interconnection between agents and process fragments, it is possible to use other workflow tools (e.g. than CAGIS SimpleProcess) as well as other agent systems. This makes the CAGIS PCE more open-ended, and ready to meet future challenges. The GlueServer and the GlueModel are described in further detail in [12].

A typical interaction between the three main components in the CAGIS PCE is illustrated in figure 2 and can have the following sequence:

1. The workflow tool, *CAGIS SimpleProcess*, will report its process state to the *CAGIS GlueServer*.
2. The *CAGIS GlueSever* then looks for cooperative rules defining relationships between process fragments and software agents. If a relationship that maps the current process state is found, the *CAGIS GlueServer* will initiate one or more software agents in the *CAGIS DIAS*.
3. The software agents will typically execute cooperative activities where several roles are involved. When the cooperative effort has terminated, a result will be reported to the *CAGIS GlueServer* (e.g. successful negotiation).
4. The *CAGIS GlueServer* will then activate a reaction defined in the GlueModel specified for the result. Reactions can be executed in the *CAGIS SimpleProcess* workflow tool, the *CAGIS DIAS* or in the *CAGIS GlueServer* itself.

A more detailed description of the CAGIS PCE is presented in [11].

4.4 Comparison of Mobility Support

The comparison of mobility support in the three PCEs described above is based on the requirements listed in section 3. Since none of the PCEs we have compared has been designed initially to support mobile work, and we have investigated what is lacking in this respect. Table 1 presents the results from comparing the three PCEs with regards to mobility requirements.

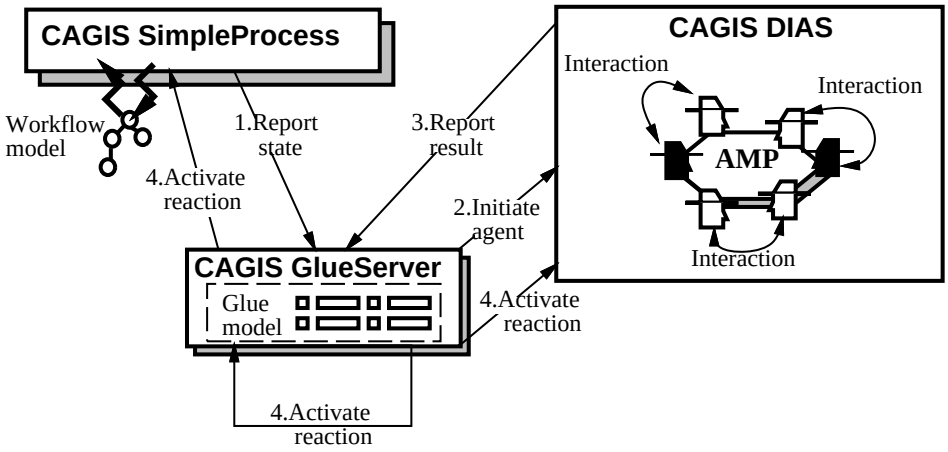

Figure 2: The CAGIS PCE architecture

4.4.1 Process Modelling

From the table, we can see that all PCEs have support for defining models for generic processes as expected. The way the processes are modelled in the three PCEs however varies. In Endeavors, processes are modelled as network of activities combined with artifacts and resources. The process models in ProcessWeb are represented as a set of roles with interactions. The interactions represent data- and control-flow between roles. CAGIS PCE uses a combined process modelling approach where individual activities are modelled as activities with pre-order dependencies and cooperative activities are represented by several cooperative agents. In addition, a GlueModel is used to model the dependencies between activities and agents.

The abilities to model mobile work are not so well supported as for general processes. In ProcessWeb, the process modelling language does not facilitate specific support for mobile work. Endeavors and CAGIS PCE both give some support for mobile work by allowing parts of the process model to move between sites and expressing the location of the mobile unit. In Endeavors, the execution of an activity represented as an applet can be moved to be executed locally. In CAGIS PCE, both process fragments (representing one or more activities) and software agents can be moved between workspaces. The location for the mobile unit is represented as an URL for both Endeavors and CAGIS PCE. None of the PCEs has extensive support for context changes for mobile units.

4.4.2 Process Architecture

All PCEs provide the most necessary features needed to provide a generic process architecture for cooperative work such as facilitating the creation of a community memory, supporting distribution of artifacts, supporting integration with external tools, and being platform independent. Integration of external tools for clients are rather limited in ProcessWeb and CAGIS PCE where most of the user interaction is provided through a Web-browser. Endeavors is also the only PCE that is completely platform independent (also the server), since it is entirely implemented in Java.

Requirement	Endeavors	ProcessWeb	CAGIS PCE
1.1 Generic Process Modelling	Yes	Yes	Yes
1.2.1 Unit of mobility	Applet	No	Proc. frag. / Mob. agent
1.2.2 Location	URL	No	URL
1.2.3 Context changes	No	No	No
2.1.1 Facilitating Comm. Mem.	Yes	Yes	Yes
2.1.2 Distrib. of artifacts	Yes	Yes	Yes
2.1.3 Integr. external tools	Yes	On server	Only Java-appl./plug-ins
2.1.4 a) Platform indep. server	Yes	No	No
2.1.4 b) Platform indep. client	Yes	Yes	Yes
2.2.1 Hardware indep.	Partly	Partly	Partly
2.2.2.1 Off-Line support	No	No	No
2.2.2.2 Sync off-line	No	No	No
2.2.2.3 WF changes off-line	No	No	No
3.1 Process Support Tools	Yes	Yes	Yes
3.2.1 Middleware HW comp	No	No	No
3.2.2 Middleware SW mobile	No	No	Partly

Table 1: Comparison of three PCEs in regards to mobility support

Requirements needed for the process architecture for mobile work is rather poorly supported by the three PCEs. All the PCEs supports workflow clients that can run on different platforms, but smaller mobile devices such as PDAs and mobile phones are not supported. Support for handling off-line workflow, synchronisation between devices after being off-line, and handling workflow changes when devices are off-line are not support by any of the PCEs. The commercial version of Endeavors has however capabilities to deal with such problems.

4.4.3 Process Support Tools

All the PCEs provide a set of process tools or building blocks for implementing tools for group awareness, for sharing information, cooperative activities, change management, etc. Middleware for hardware mobility used e.g., to detect location changes are not present. CAGIS PCE has some limited support for middleware utilising software mobility in the GlueServer component. The GlueServer makes it possible to specify that at certain events, parts of the process model (process fragments) should be moved from one workspace to another. In addition, this GlueServer can initiate mobile agents.

5 Conclusions and Future Work

This paper has identified some requirements to support mobile work. We have further compared three existing PCEs in regards to mobility support and found that they do not provide the needed facilities to support such work. In a new project called MObile Work Across Heterogeneous Systems (MOWAHS), we will use CAGIS PCE as a starting point

to a PCE that is able to support mobile work. We will also look at other existing technology enabling mobile work. Only future experiments will show how useful a PCE for mobile work is, and realistic scenarios should be used to evaluate our approach.

References

[1] Reidar Conradi (ed.). MOWAHS: Mobile Work Across Heterogeneous Systems, 2000. Research Project supported by the Research Council of Norway 2001-2004, IDI/NTNU.

[2] A.L. Murphy G-C. Roman, G.P. Picco. Software Engineering for Mobility: A Roadmap. In *The future of Software Engineering (ed. by A. Finkelstein), ACM Press, p.243-258*, 2000.

[3] Jonathan Grudin. Computer-Supported Cooperative Work: Its History and Participation. *IEEE Computer*, 27(5):19–26, 1994.

[4] Anders Aas Hanssen and Bård Smidsrød Nymoen. DIAS II - Distributed Intelligent Agent System II. Technical report, Norwegian University of Science and Technology (NTNU), January 2000. Technical Report, Dept. of Computer and Information Science.

[5] Arthur S. Hitomi and Dong Le. Endeavours and Component Reuse in Web-Driven Process Workflow. In *Proceedings of the California Software Symposium*, Irvine, California, USA, 23 October 1998.

[6] IDI NTNU. CAGIS - Cooperating Agents in the Global Information Space, 1996. in http://www.idi.ntnu.no/IDT/f-plan/prosjekter/cagis.html.

[7] Geir Prestegård, Anders Aas Hanssen, Snorre Brandstadmoen, and Bård Smidsrød Nymoen. DIAS - Distributed Intelligent Agent System, April 1999. EPOS TR 359 (pre-diploma project thesis), 396 p. + CD, Dept. of Computer and Information Science, NTNU, Trondheim.

[8] ICL Enterprises Process Management Centre, Enterprise Technology. *ProcessWise Integrator, PML Reference*. Staffordshire, UK, first edition, April 1996.

[9] Heri Ramampiaro, Terje Brasethvik, and Alf Inge Wang. Supporting Distributed Cooperative Work in CAGIS. In *4th IASTED International Conference on Software Engineering and Applications (SEA'2000)*, Las Vegas, Nevada, USA, 6-9 November 2000.

[10] Alf Inge Wang. Support for Mobile Software Processes in CAGIS. In Reidar Conradi, editor, *Seventh European Workshop on Software Process Technology*, Kaprun near Salzburg, Austria, 22-25 February 2000.

[11] Alf Inge Wang. *Using a Mobile, Agent-based Environment to support Cooperative Software Processes*. PhD thesis, Norwegian University of Science and Technology, Dept. of Computer and Information Science, NTNU, Trondheim, Norway, February 5th 2001.

[12] Alf Inge Wang, Reidar Conradi, and Chunnian Liu. Integrating Workflow with Interacting Agents to support Cooperative Software Engineering. In *Proc. IASTED Internation Conference Software Engineering and Applications*, Las Vegas, Nevada, USA, 6-9 November 2000.

[13] Alf Inge Wang, Chunnian Liu, and Reidar Conradi. A Multi-Agent Architecture for Cooperative Software Engineering. In *Proc. The Eleventh International Conference on Software Engineering and Knowledge Engineering (SEKE'99)*, pages 1–22, Kaiserslautern, Germany, 17-19 June 1999.

[14] Benjamin Yeomans. Enhancing the world wide web. Technical report, Computer Science Dept., University of Manchester, 1996. Supervisor: Prof. Brian Warboys.