Information Support for User-Oriented Development Organisation – Considerations based on the Construction and Evaluation of Knowledge Storage

Marko Nieminen
Helsinki University of Technology
Department of Computer Science and Engineering

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Department of Computer Science and Engineering for public examination and debate in Auditorium T2 at Helsinki University of Technology (Espoo, Finland) on the 22nd of October, 2004, at 12 o'clock noon.

Helsinki University of Technology **Abstract of the Thesis**

Author: Marko Nieminen **Department:** Department of Computer Science and Engineering	**Subject of the Thesis:** Information Support for User-Oriented Development Organisation – Considerations based on the Construction and Evaluation of Knowledge Storage **Number of Pages:** 228 (206 + appendices)
Chair T-121	**Supervisor** Kari Kuutti, doc., prof., PhD

User-centred development of interactive systems and devices has received increasing importance in product development organisations. So far, the answer from the usability engineering community has been the offering of different types of methods that can be applied during the development work in different stages of the development process. As the amount of applicable methods increase and the stakeholder population utilising these methods broadens, support for the management of this usability engineering work becomes important. This includes considerations about the arrangement of the organisation performing the development work as well as the tools and methods that supports this.

The objective of this study is the modelling and construction of information support for product development that takes user-centred issues into consideration. The main research question of the study is "What kind of information management system can provide support for user-oriented product development?" Boundaries for the main research question are presented with the focus areas of the work that point out insights from the organisational standpoint as well as from the standpoint of the methods and tools.

The methodological standpoint has a longer tradition in the HCI field than the organisational one. In the literature review part of this thesis, methods that are used rather widespread are introduced in order to find out the characteristics and requirements that they point out for the surrounding information support systems and the hosting organisation. The organisational standpoint is studied and reflected in the three empirical studies that illustrate the contemporary arrangement and organisation of product development. Product development is addressed also in the literature review.

Building on the findings, a framework of the characteristics for a hosting organisation is presented. The framework consists of five levels: organisational orientation (values, attitude), life-cycle considerations (business/process/product), generic development support (methods and tools), quality instructions (organisation-specific adjustments of the generic level), and information support (operative level). This framework points out the position of an information support system in the organisational surroundings. After this 'positioning', more detailed modelling, design and implementation of the supporting information support system, the "Knowledge Storage" is presented.

The results from the construction and evaluation of the Knowledge Storage point out needs for information support applications (developer community, roles, awareness, contribution evaluation). The results also reveal difficulties in the making of these kinds of applications attached to real development projects and activities (migration of existing knowledge base, 'suitable' project, application integration, implementation of baseline functionality vs. value-added features).

Teknillinen korkeakoulu **Väitöskirjan tiivistelmä**

Tekijä: Marko Nieminen **Osasto :** Tietotekniikan osasto	**Työn nimi:** Käyttäjäkeskeisen tuotekehitysorganisaation tietotuki – Pohdintoja ”Knowledge Storage” –tiedonhallintaympäristön toteutuksen ja arvioinnin perusteella **Sivumäärä**: 228 (206 + liitteet)
Professuuri: T-121	**Työn valvoja:** Kari Kuutti, dos., prof.

Käyttäjäkeskeisyyden merkitys tuotekehitystoiminnan yhtenä suunnittelunäkökulmana on kasvanut viimeisen vuosikymmenen aikana merkittävästi. Käyttäjäkeskeistä suunnittelua painottavan tutkimusyhteisön tarjooma käytännön suunnittelutyölle on perinteisesti ollut joukko erilaisia suunnittelumenetelmiä, joita on mahdollista hyödyntää eri vaiheissa suunnittelua. Menetelmien määrä samoin kuin niitä hyödyntävien suunnitteluosapuolten määrä onkin näiden johdosta kasvanut. Tämä on luonut tilanteen, jossa käyttäjäkeskeisten menetelmien soveltamista ja niillä kerättyä tietoa on tarpeen hallita entistä systemaattisemmin keinoin.

Tämän tutkimuksen tavoitteena on käyttäjäkeskeisen tuotekehityksen tueksi soveltuvan tietotukiympäristön mallintaminen, kehittäminen ja arviointi. Tutkimuksen pääkysymys on: "Millainen tiedonhallinnan järjestely voi tarjota tukea käyttäjäsuuntautuneelle tuotekehitykselle?" Pääkysymystä tarkentavat rajaukset nostavat esiin kaksi näkökulmaa: organisatorisen ja välineellisen.

Metodiikkapainotteisella näkökulmalla on vahvempi perinne ihminen-tietokone-vuorovaikutuksen tutkimusalueella kuin organisatorispainotteisella. Työn kirjallisuuskatsauksessa tarkastellaankin yleisesti sovellettuja käyttäjäkeskeisen suunnittelun menetelmiä hakemalla niistä piirteitä ja vaatimuksia, joita niiden soveltaminen asettaa ympäröivälle organisaatiolle. Organisatorista näkökulmaa tarkastellaan kirjallisuuskatsauksessa tuotekehitystoiminnan yleisten piirteiden kautta sekä lisäksi kolmen empiirisen nykyaikaista tuotekehitystoimintaa selvittävän tutkimuksen avulla.

Selvitysten tulosten perusteella työssä kehitetään viitekehys, jonka avulla tarkastellaan käyttäjäkeskeisesti toimivan organisaation edellytyksiä soveltaa käyttäjäkeskeistä suunnittelua ja siihen liittyvää tietotukea. Viitekehyksessä tunnistetaan viisi elementtiä: organisatorinen suhtautuminen (arvostukset, asenne), elinkaaritarkastelut (liiketoiminta/prosessit/tuotteet), toiminnan yleisen tason välineistäminen (menetelmät), laatuohjeistus (organisaatiokohtainen menetelmien soveltaminen) ja toiminnan operatiivisen tason tietotuki. Viitekehys sijoittaakin tietotukiratkaisut laajempaan organisatoriseen ympäristöön.

Viitekehyksen esittämisen jälkeen työssä esitetään "Knowledge Storage" -tietotukiratkaisun kehittäminen. Tulokset tästä kehitystyöstä osoittavat vaatimuksia organisatorisessa viitekehyksessä toimiville tietotukiratkaisuille (yhteisöllisyys, roolijako, tilannetietoisuus, palaute yksilöllisistä kontribuutioista). Tulokset nostavat esiin myös merkittäviä haasteita ja vaikeuksia tämän tyyppisten ratkaisujen kokeilussa ja toimivuuden arvioinnissa todellisten käytännön projektien yhteydessä (aiemman tietoainespohjan hyödyntäminen, sopivan luonteinen projekti, perustoiminnallisuusvaade vs. uusi lisäarvoa tuottava toiminnallisuus).

Foreword

Once upon a time in the late 1980's, during one of my first jobs in an industrial company as a PC support person I was amazed by the fact that intelligent, experienced, and highly productive people had tremendous problems in using computers in their daily working activities. At first I thought that these people just need to be trained to use the applications and that fluent utilisation of the word processing and spreadsheet programs will take place right after that. Therefore, training sessions, both classroom and personal ones were arranged. And yes: some of the features of these applications seemed to become understood by these users. At the same time, however, many additional problems arose despite the fact that the basic functionality seemed to be understood.

After a while having seen the continuing struggle with the applications, it started to become evident that it is not only a question of training. There are more fundamental problems – but not in the direction of the ones who use the applications. The functionality of the applications did not seem to conform to the thinking models that the users had: the match between the users' tasks, goals, and expectations towards the tools that were supposed to help them with the daily activities seemed to be pretty far from the users' conceptual models.

These personal findings made me wonder: Are there any ways that we could use to improve the match between the users' world and the concepts and functionality that computer applications and other interactive devices offer to these people. How could we make it easier?

A couple of years from that personal finding, I found (as part of my undergraduate studies) that this topic is an active research field in the academia. Starting from user interface design issues, user-centred design, participatory design and usability engineering were topics of interest in the area of human-computer interaction (HCI). This multi-disciplinary and bridging area between human activities and technology raised my interest and provided an area to study further these interesting phenomena and research area.

This thesis appears to be a description of that research path in my case.

The writing of this thesis has taken more than ten years of my life. During that time, I have been – should I say "privileged" or "forced" – to follow the creation and writing of the doctor's thesis of my father. At the time of writing and finalising of his thesis, I figured out having found out how not to do a doctor's thesis. Based on the contemplation about these issues I have a hunch that a doctor's thesis should not be a thorough and to the finest detail complete description of ones own life-long work within specific field of academic research. Instead, I believe that that the doctor's thesis should be the almost first academic deliverable that is the starting point for a successful researcher career. There is probably room for extra research after the thesis.

Despite the example and experience that I have had, I feel not having succeeded too well in this matter. From this standpoint, there is still room for improvements in the world around us that is left for the next generations – our children – to do. This thesis is in many ways a closure of my ten-plus-something-years long researcher career – and may also somewhat suffer from that addressing a bit too broad thematic area. On the other hand, this lengthy period of creating the thesis may have provided some additional standpoints and insights that would, perhaps, otherwise have been left out, who knows?

The oldest parts of the work go back to years 1994 and 1995, to some extent even further back in the history. Because of the lengthy process, this thesis makes use of data that has been gathered during several research projects. These projects have been funded by the Academy of Finland, TEKES (the National Technology Agency), and the Finnish Work Environment Fund. Several industrial companies have also participated in the funding – and in the realisation of the various studies for this work. Without this possibility and access to real-life and "ecologically valid data" this thesis would not have been completed.

During the planning, data gathering and writing of this thesis, lots of people have provided their valuable time, effort, and intellectual consideration about the topics included in the work. I present my deepest gratitude to them all. First, I want to present gratitude to my supervisor professor Kari Kuutti for his continuous interest towards my work. The both encouraging and critical comments about the text and progress of my work pushed me always forward – sometimes perhaps slowly but surely. I also wish to thank the persons who have provided me the fundamental basics for doing research in the most intriguing academic field during the years: Martti Mäntylä, Matti Vartiainen, and Anneli Pulkkis. And my close research colleagues: Jyrki Kasvi, Lauri Repokari, Toni Koskinen, Sirpa Riihiaho, Turkka Keinonen, and Simo Säde; You all have had an enormous effect on my work via the interesting discussions throughout the years. Without the possibilities to work closely with industry, this thesis could not have been completed. The long-term co-operation with Pirkko Jokela and Eija Suikola made this possible in an important way.

This thesis would have remained undone, if my closest people had not given their full support for this work. My parents Anni-Marja and Heikki-Tapio: Your unceasing interest towards the topic of my studies and the name of my thesis work has given it a special flavour. Your support for Your children's life-long studies has been so important.

After all, the ones who have given me the everyday possibility to work with this almost obsession-like work are my own beloved wife Saija and children Mikael, Lassi, and Milja. I know that You have not always understood the reasons why I am doing all this – me neither. After having completed this work, I hope to be able to get back, close to the missed moments in time, joy and play. – Back to real life.

Marko Nieminen

Information Support for User-Oriented Development Organisation – Considerations based on the Construction and Evaluation of the Knowledge Storage

Marko Nieminen
Helsinki University of Technology

Table of Contents

1 Introduction: Make IT Easy with Human-Centred Design

Futures made of virtual insanity now
Always seem to, be govern'd by this love we have
For useless, twisting, our new technology
Oh, now there is no sound - for we all live underground

– Virtual Insanity, Jamiroquai (Jay Kay / Toby Smith) –

1.1 User-Centred Design – Heading Towards Recognised Design Approach

The recent trend "computers are everywhere", embedded and pervasive computing, makes almost everyone a user of a computer, an interactive device. Currently, however, there are loads of problems and hindrances in using these devices and systems. We have plenty of examples about these not only from our daily lives but from the reported usability studies as well; some of these problems have been identified and summarised also in Helsinki University of Technology in the mid 1990's ("Usability defects": Koivunen & al. 1996, Nieminen & al. 1996[1]).

In principle, designing easy-to-use products with interactive functionality has been recognised as an important issue in many product development companies. Despite this, real resourcing and the real-life development practices do not always seem to provide enough support to produce this desired result. A basic assumption and hypothesis in this work is that the usability problems emerge – at least partially – as a result of insufficient emphasis in the user-centred issues in user interface design work. This statement points out that the product development work that is behind the functionality of the realised products may be improved with activities that have more focus in the user-related issues.

[1] A short summary of studies conducted in the usability laboratory of Helsinki University of Technology during years 1993-1996 (usability tests of 14 computer systems and 10 embedded devices).

The general rationale and utility for adopting a human-centred *design process* in product development work is presented in the ISO 13407 (1997) standard as follows. The products and systems that are developed according to human-centred development principles

- "are easier to understand and use, and they reduce the training and support costs
- reduce discomfort and stress and improve user satisfaction
- improve the productivity of users and operational efficiency of organisations
- improve product quality, aesthetics and impact and can provide a competitive advantage."

The user-oriented approach in product development is neither a trivial nor an easy issue to cope with. The difficulties relate to the existing development practices and development approaches that often have a strong emphasis on technological issues. The introduction of user-centred approach in this kind of an organisational environment raises many challenges for instance from the standpoint of gathering user requirements in the early stages of the development process, modelling user in the appropriate context of use, and taking into account several uncertain, vague and sometimes even arguable soft issues. The user-centred, or user-oriented, development approach is supposed to help developers and development organisations to deal with this complicated area. In this thesis, the aim is to study and find out some possibilities on how to organise and arrange some of this type of development work.

This thesis is positioned in the academic discussion in the area of human-computer interaction (HCI), especially the sub-field that can be considered nowadays as "usability engineering" (this positioning is discussed more in the latter parts of this work). In the 1980's, there was initially lots of emphasis on detecting the problems in the interaction between the human and the computers, modelling the human-computer interaction, and developing user interface evaluation methods[2]. Alongside with these topics, a research line with the focus on the integration of user-centred design (UCD) with other areas of product development got started during mid- and late 1980's. At that time, *usability engineering* was introduced and defined by several researchers[3].

[2] See e.g. the works of Card & Moran (GOMS, 1983), Lewis (thinking-aloud, 1982), and Norman (1986, seven stages of interaction).

[3] For instance Bennett (1984), Good & al. (1986), and Whiteside & al. (1988) introduced and discussed the concept usability engineering. In the beginning of 1990's Nielsen (1993) expanded the concept of usability engineering to be closer to a "discipline" than just a collection of systematic methods.

This usability engineering research line can be seen to have continued and evolved during the late 1990's towards a design approach in the development work that is of central importance – or desired to be as such. This progress has also led to the academic discussion about the "enabling surroundings" and organisational aspects on arranging the development environment and infrastructure in such a way that it is able to support the user-centred approach efficiently. This thesis is aligned with this continuation.

One aim in this study is to lay out initiatives and mechanisms that embed the management of these user-related issues along with other, already existing issues in product development: How should the management of user-related issues be organised in a development organisation that has a strong intent towards user-centred issues? What are the elements of user-centred information space within the context of product development and how can these elements and information be best utilised? By studying the possibilities for the management of user-centred information and the enabling pre-requisites also at the organisational level we may be able to proceed towards more systematic and "culturally acceptable"[4] (within the context of product development) usability engineering.

1.2 Information Management Tools for User-Centred Development Work

Usability points out characteristics that have an influence on human-computer interaction. These issues are dealt with user-centred design. Usability defines factors that cover the characteristics of the task, the user, the environment and the functions of the system (Eason 1984, ISO 9241 1996). Nielsen (1993) presents that usability can be evaluated by measuring learnability, memorability, effectiveness, error rate, and users' subjective attitude.

Usability of a certain device can be looked at from two directions: the *development* of the tool and the *utilisation* of the device. But even if we look at usability from the direction of development, there is a need to reach out to the "real life" in the utilisation of the product. One strong and much emphasized approach has been usability testing.

[4] The affect of development culture has been studied in e.g. Ngwenyama & Nielsen (2003) in their paper about "Competing Values in Software Process Improvement: An Assumption Analysis of CMM from an Organizational Culture Perspective".

Traditionally, usability testing has been considered to be of main importance in the usability engineering field[5]. Usability testing has been among the most important usability engineering practices during the 1990's. Despite the obvious utility and practicality of the method, there are challenges with this *usability-testing-only* approach: Very often there are no possibilities for massive further improvement of the tool/product at this stage in the development process[6]. All major decisions about the features and constellation of the product have already been done before usability takes place in the development process. One may end up in a situation in which we can only recognise the problems and start living with them. It may be possible to bundle the product with additional instructions for use tied with the operating manuals. Despite this, a more *proactive* approach is needed that is capable of addressing the usability problems in advance to overcome them.

To affect the usability characteristics of a product related to its functional features, there is a need to address the user-centred and usability issues already in the very early phases of the development process. Such a positioning of user-centred development emphasises the importance of more versatile and thorough modelling of the user, the environment, and the tool under development in the context of the other artefacts around the user. This type of approach towards broader and more comprehensive modelling of the user and his or her context-of-use is reflected e.g. in the AUTO model presented by Boy (1998). This model takes into account information about the artefact (A), user profile (U), task (T), and organisational environment (O) providing important additional user-centred viewpoints for consideration by the developers. The arrangement of the development activities in the development organisation should enable the consideration these issues during the different stages of development process.

The development process of a tool/product includes several stages that require (or at least provide possibilities for) input from user-centred standpoint: Pre-studies, design, implementation, testing, production and customer delivery are examples of such stages (e.g. Ulrich & Eppinger 1995). All these stages involve a huge amount of design information, among which usability and user-related information is – or should be.

[5] This is reflected e.g. in the Nielsen's book "Usability Engineering" from 1993 that dedicates much of its contents to usability testing.

[6] It must be noted that this approach does not take into account the development of a tool with many versions: With "product version" type of approach, usability testing may well be used for gathering improvement suggestions and requirements for the next release.

Development work seems to be turning from *tech-only* activity into *multi-disciplinary* work. More often already in the very first stages of design, a group of people with different backgrounds and skills is involved. Due on this finding, communication and composition of development stakeholders are of great importance in this study. The results of different (user-centred) design activities need to be communicated to all relevant stakeholders. To do this, the outcome from the (user-centred) methods that are applied need to be understood by the representatives: what are the contents of each applied method? What kind of information does the method deal with? What does the outcome mean? How sure can we be about the findings with the method?

Effective communication requires that participants of different expertise areas really understand what others present. The implementers of software want to have detailed technical specifications before starting to implement a system. Marketing personnel detects customer wishes and technological requirements set by competitors. End users evaluate products' standpoint according to their own goals and activities. They may not fully understand all detailed technical issues that may often be seen as crucial from the developers' side. But: users are experts on the task domain. A translation mechanism of these topics may be required in order to enable systematic and effective development procedure.

To support communication in the early stages of product development, means for early modelling provide the steps for the translation of the specifications. Modelling techniques may vary from user characterisation and task scenarios to the creation of prototypes of the product. Examples of methods that may emerge in the development process and that need to be understood by the participants are as follows. In user characterisation, a figure of a typical user with proper background and environment information will help in the early usability design[7]. Based on the user, task and environment information, usability goals can be set and they in turn can help in the construction and design of a usability test. A usability test in a laboratory or real environment with observation and thinking-aloud method can be conducted e.g. according to the short instructions presented by Gomoll & Nicol (1990).

From this standpoint, the amount of user-centred information in a development project may become huge and involve a large amount of stakeholders. It may turn out that the

[7] Contextual Inquiry (Duncan & Beabes 1995, Beyer & Holzblatt 1998) and Contextual Design incorporate much of this type of information and these are experimented in this study.

management of this information becomes problematic without appropriate tools and design methods.

In a panel held in CHI’99, Ken Dye presented that “usability engineering has not developed methods for preserving user data for use throughout the development process”. Additionally, he continued that “Organizations may collect excellent data about users in the planning phase, but not use that data consistently while coding the product.” (Rosenbaum 1999) This statement emphasizes the need for organisation-wide information management tools that are capable of supporting the gathering, storing, sharing, and analysing of user-related information that is often dispersed in separate business functions of a company.

Even though the characteristics of the human being do no change rapidly, the observations about human activities in changing contexts may well change the requirements that are set to the product under development. From this standpoint, the nature of user-related information is changing and even fluctuating. The management of this type of information requires flexibility and adaptability that may be grasped with modern information systems and supporting conceptual models like active design documents (Boy 1997).

This thesis work aims at constructing a conceptual framework and a supporting information support application for product development work that enables the utilisation of user-centred methods in a multi-disciplinary development organisation. From this standpoint, there is a need to address also the development of a computer-supported collaborative information support environment for design and development work.

1.3 From Individual Practitioners to Organisational Entities

This research started very much from the idea that providing appropriate supporting tools to manage important information improves the quality of design. Therefore, the initial idea in making user-centred development function more systematic has been to research means for supporting the management of user-related information in the development organisation – or development network.

However, during the implementation of the research it became very clear that additional tooling and instrumentation for developers’ use is not the whole solution when improving the user-orientation in a development organisation. As user-centred development is somewhat new concept in most organisations, there are additional practical, and should it be said "mental" as well as organisational hindrances on the way as well. These problems require other solutions than instrumentation or "tooling". Due to this reason, an additional aspect that is being dealt with in this thesis is the organisational framework and its capability to "host" user-centred development work and corresponding instrumentation.

In addition to the findings during the implementation of this thesis, it was recognised in the beginning of 1990's that due to both organisational and technical reasons, user-centred design was not conducted widely in practice. This finding has more recently, at the end of 1990's, received additional academic discussion in the field of "strategic usability" that has a strong emphasis on organisational issues. One can see that the discussion has turned from individuals who are performing single UCD practices into UCD professionals (instead of the generic term "product developers") and even to UCD organisations that perform a complete set of business-related UCD activities. However, the operation and management of such an organisation is not modelled yet. With the support of the empirical material gathered in this research, an overall framework about the hosting and supporting organisational surroundings for UCD is presented.

Despite the extension towards these organisational issues, this thesis does not only present conceptual considerations about the arrangement and organisation of user-centred product development work but also practical piloting of information management instrumentation for user-centred product development work that supports the functioning of such an organisation. The underlying assumption here is that the form of the performing organisation is dependent on its supportive "tooling" – and that the tooling is dependent on the form of the organisation. Which of these comes first is somewhat unclear and, therefore, considerations on the other part may require considerations on the other one.

With the aim to point out concepts and ideas that provide basics towards practical results that can support the development and improvement of such a development organisation, we must take into account another additional field of research. Computer supported collaborative work (CSCW) is a field of research that has its focus in questions that address the collaborative information management tools. Cross-functional management of user-related information falls into this category. Therefore, this thesis addresses the two fields of research: HCI – that has already been introduced shortly – and CSCW – that will be introduced later.

2 Research Approach

2.1 Research Question

Motivation for the work. As presented in the introduction of this thesis, introspective findings about the utilisation of computer systems at work pointed out the need to do more detailed and comprehensive designs of the systems so that they provide better support for users. During various discussions with development practitioners it has turned out that the inclusion of user standpoint is not a trivial and "just technical" topic to deal with. There are several hindrances in the real-life development organisations that prevent user-orientation to be regarded as a truly legitimate design approach.

Sometimes heard statements about the need and importance of the user interface design and user-centred development activities are following: "the development of a user interface is trivial and anyone can do it", or "all our products do have a user interface", "discussions with users – they are just using common sense", "we conduct customer studies and, therefore, already know what they want". Even though it may be possible to reach good results with the approaches behind these statements, solid and repeatable high-quality development work requires improved systematics and processes that produce in-depth data about users' needs, user characteristics, and users' environment – in which the resulting product will also operate. Motivating the practitioners to consider user-related issues as important as technical issues is one aspect in this field, provision of appropriate methods and tools to support the work is another. A subset of the latter topic is the provision of appropriate information management mechanisms that focus especially to user-centred issues.

Even though user-centred design has already gained popularity during the last years also among traditional technical product development areas, the practices and processes that provide explicit support for systematic management of user-related issues are in many ways vague at the moment. Many other activities and processes in other disciplines and areas of product development can be seen to be considerably further in managing development work and information management. For example issues that relate to manufacturing, logistics, quality control, integrated manufacturing, production control, even marketing have often established and settled procedures and processes that are often described in detail in company specific quality instructions.

If user-centred design is to be regarded as an important element among all the other themes, one can claim that similar systematic working practices, procedures and supporting tools as the other disciplines within development work are required. One way to approach this issue is to try to look at the other working practices and apply similar techniques in the context of user-centred design. An interesting question for serious consideration is, of course, how well activities in other disciplines fit to the non-technical, as very often considered, user-centred development activities.

Studies about the possibilities on how to promote and support user-orientation in a real-life development organisation are required. User-orientation in the development organisation is not, however, the ultimate goal: user-centredness in a development organisation is not of central importance per se; it serves as an intermediate activity that makes the work and life of the end-users easier with their interactive tools. But, without this intermediate activity, the final goal cannot be reached.

Personal motivation. In addition to the general motivation for the thesis work, there is also personal motivation involved in the research presented in this thesis. The foundation for the research topic comes from the idealistic belief that technology can and will help people in all kinds of real-life-activities. This help can unfortunately be ruined if technology is developed without taking the human aspect into account. To take the human aspect into account, special engineering working practices and supporting tools are needed. In the field of usability engineering, such methods and tools have been introduced, but there is still room for improvement. My contribution in this thesis is in the field of “information support instrumentation for user-oriented development organisation”.

Additionally, there is personal motivation involved and visible also in the setup of the research arrangement. My approach to research in this thesis can be called *constructive*: In order to gain conceptual level results (models, theories) that may be applicable wider, somewhat restricted “partially-phenomena-explaining” applications may help in determining the meaningful factors affecting the phenomena. The applications often require a practical approach through the creation of working prototype implementations. These implementations need to be examined and evaluated carefully in order to get results from their usage. Being an engineer, this “modelling-via-implementation-and-evaluation” approach fits easily to the mindset of an “engineer-researcher”. Creating something that runs and works is somehow fundamental to engineers and their work.

Research question. The main research question of this study is

What kind of information management system can provide support for user-oriented product development?

A short clarifying and restricting remark of the research question needs to be pointed out right away: before being able to answer the more practical question about the “information management system”, there is a need to define the characteristics of the organisation that acts in a user-oriented way. From this standpoint, there are two questions:

1. What kind of organisational environment (or “framework”) is required that can
2. embed (or “host”) systematic user-centred development activities

The concept “system” refers not only to a technical implementation of a tool but it also addresses the organisation, its activities, working processes and procedures, people and their tasks. The interaction between the individual and the surroundings is of major interest from this standpoint[8].

[8] In many ways, this approach is aligned with *activitity theory* that originates in the 1920’s and 1930’s in the works of Vygotsky and his colleagues Leont’ev and Luria. Even though this thesis does not aim at providing a comprehensive view of this topic, some links towards the academic discussion in the fields of HCI and CSCW are pointed out (for “academic positioning” see next chapter). Activity theory looks at human activities in the context of his/her surroundings based on the interpretations that the person makes. Kaptelinin & Nardi (1997) present that “Activity Theory … provides a broad conceptual framework for describing the structure, development, and context of computer-supported activities”. In CSCW and HCI research areas activity theory has provided much of the theoretical basis for analysing the interaction of the individual with the surrounding people and artefacts. A practical example about the existence of the topic in the research field is for instance the tutorial held by Kaptelinin & Nardi (1997) in the CHI’97

To answer the research question, following issues provide boundaries for the main research question:

1. What is user-oriented product development? What is position and relation of user-oriented, or user-centred, activities to other product development activities and tasks? How is user-oriented product development defined and arranged in practice?
2. What is information management that supports the activities of a development organisation? How can information support be arranged in a contemporary product development environment?
3. What kind of services should an information support environment for user-oriented development provide? What is an appropriate structure and technical realisation of an information support system that is feasible to implement and test in contemporary product development environments?

The topics that are not being addressed (explicitly left outside the scope of the work) and covered by this work include

- decision support ("How decisions are finalised". However, this thesis will address the gathering and of background information that will be used in the decision making process)
- in the last part of the work, synchronous collaboration will not be addressed ("on-line collaboration with other developers")

With these constraints, this thesis reaches towards **the architecture of user-centred product development that is supported by an information support system capable of managing user-centred design information.**

---

conference with the topic "Activity Theory: Basic Concepts and Applications". Kaptelinin, Kuutti & Bannon have written a chapter with the same title in an edited book by Blumenthal et al. (1995, cf. Kaptelinin & Nardi 1997). A collection of links about activity theory related issues can be accessed at http://carbon.cudenver.edu/~mryder/itc_data/activity.html that is created and updated by M. Ryder (checked 8-Jul-2004, last update 1-Jul-2004). Recent discussion about the relationship of activity theory and CSCW is available in the "Special Issue of CSCW" with the title "Activity Theory and the Practice of Design" edited by David Redmiles (2002). Kuutti (1996) has presented links between HCI research and activity theory in a book chapter (Nardi (ed.) 1996) entitled "Activity theory as potential framework for human-computer interaction research".

2.2 Academic Positioning: CSCW applied to HCI

Contributing academic research fields. This research topic can be positioned to two research fields: human-computer interaction (HCI) and computer-supported co-operative work (CSCW). In this thesis, HCI is in the form of providing contents for the framework that in many ways falls into the field of CSCW. HCI, or "usability engineering" to be more precise in the scope of this work, points out several user-centred design and development activities that may be carried out with the support of a CSCW application. Bannon (1993) presents several CSCW perspectives[9] that this research field incorporates. This thesis aims at addressing not only one of these perspectives (but certainly aims *not* at addressing them all) but building on a somewhat holistic view of the field taking into account the technological "constructionist"[10] direction aligned with the social and organisational aspects. The structure of the last chapters of this thesis reflects this.

The application area of this work to which the CSCW approach is applied, is usability engineering (that is strongly attached to the human-computer interaction research). Usability engineering has been actively studied in the academia since the late 1980's[11] even though the concept "usability" has been presented already earlier[12]. At the moment the *usability engineering practices* are being introduced into practice in product development organisations. This activity within product development does not yet have fully established organisational forms, working methods, and supporting tools. In order to create products whose features really fit users' needs, characteristics, and contexts,

[9] The different perspectives are presented later in this work in the part that presents the positioning of this work to the theoretical background from the CSCW research field.

[10] Bannon (1993) presents that "Howard (1988) describes two distinct though very varied communities within CSCW. He coined the term "strict constructionists"' to describe those in the field focused on the development of computer systems to support group work, who tend to use themselves as objects of analysis in the provision of support tools."

[11] The concept "usability engineering" was introduced and launched in the mid and late 1980's to describe the various activities that are (or can be) performed in development work in order to get usable (or easy-to-use) products as a result. The concept appears, for instance, in the article by Whiteside, Bennett & Holzblatt (1988). Nielsen's (1993) book "Usability Engineering" can be seen to make the concept more widespread.

[12] The word "usability" has been introduced already in the 1970's. For instance Smith & al. (1980) refer to the article "Incorporating Usability into System Design" written by Bennett in 1978. Bennett has also written an article "The commercial impact of usability in interactive systems" (1979). Additionally, Shackel (1986, 52) presents that "the definition of usability was probably first attempted by Miller in 1971 in terms of measures for 'ease of use'."

these need to be considered in more detail. This work contributes to this part of usability engineering by presenting one experiment of supporting collaborative tooling for the usability engineering practitioners' community.

Research work has also been done in linking user-centred design or human-centred design (UCD/HCD) to knowledge management by Dieng & al. (1999). Their presentation about methods and tools for corporate knowledge management points out that "corporate memories are not entirely new systems; they are adaptations, evolutions or integrations of existing systems" and continue by pointing out examples of such systems that include knowledge-based systems and CSCW systems. From this standpoint, the research in this thesis may contribute to the knowledge management field as well, even though it does not aim at being in the very core of this research field. Dieng & al. (1999) also present that the underlying approach in their research is UCD/HCD. The UC[H]D approach is considered fruitful for knowledge management because it "ensures that the memory is defined in terms of users' needs" (Durstewitz 1994, cf. Dieng & al 1999).

Research Approach: Constructive Research. The core of this study is conducted under the research approach "constructive research". The study, however, starts as descriptive and exploratory (interview studies and case descriptions in order to find out the "right" problems; see stage 1 in the list by Kasanen & al. of stages for constructive research below) but the last parts of it take a step towards constructive research. At the end of the work, the results are analysed and conceptual model about the position of the construction (and underlying model) in systematic "Design for Usability" oriented product development environment is outlined.

In their paper "The constructive approach in management accounting", Kasanen & al (1993) present that constructive research is "managerial problem solving through the construction of models, diagrams, plans, organizations, etc.". Järvinen & Järvinen (1996) present that constructive research aims at creating new constructions and real world artefacts based on existing research knowledge. Constructive research is close to *design science* that is regarded parallel to traditional sciences (March & Smith 1995, Simon 1981 cf. Järvinen & Järvinen). Design science aims at creating artefacts that serve human intentions. Design science consists of building a prototype and evaluating it (Järvinen & Järvinen 1996 p. 74-). Kasanen & al. (1993) point out following stages for constructive research:

1. Find a practically relevant problem which also has research potential.

2. Obtain a general and comprehensive understanding of the topic.
3. Innovate, i.e., construct a solution idea.
4. Demonstrate that the solution works.
5. Show the theoretical connections and the research contribution of the solution concept.
6. Examine the scope of applicability of the solution.

Overview of Research Methods. This thesis consists of parts that have been done with a variety of research methods. Initially, general introduction to the issues is done with a *literature review*. The empirical part starts with an *interview study* and gets a more detailed view of the general level issues in a *case study*. The case study has been conducted with a retrospective interview and analysis of design process documents. The third part is a *constructive part*: in that part, the conceptual model of systematic management of user-centred issues in product development is created and a prototype based on the model is created. The implementation provides a working prototype of the conceptual model that follows the creation of the Knowledge Storage concept. The Knowledge Storage concept is created with a design technique called "Contextual Design", a user-centred development methodology developed by Beyer & Holzblatt (1998). At the end of the thesis, an *evaluation of the prototype* is conducted with a "pilot case study". Each research method is described in more detail in the beginning of each section.

2.3 Relevance of this Work to Academia and Industry

Relevance of this work to the academic community. The research question presented in this work has been considered important by e.g. Bannon & Bødker (1997), who mention that "shared, or common, information spaces are an under-researched topic". In their study that addresses this under-researched topic, Bannon & Bødker start their paper (1997) by mentioning "One of the distinguishing features of the CSCW field is its persistent attempts to come to terms with the sociality of work, with a view to better understanding the nature of cooperative work as a basis for designing genuinely 'supportive' computer-based information systems". This statement underlines the supportive characteristic of a CSCW application that is considered as a very important element in this work.

Another aspect in this thesis that can be seen to contribute to the CSCW research area is in the evaluation of CSCW systems. As different types of CSCW applications rapidly enter markets mainly due to the global networking infrastructure (Internet in its various forms), the evaluation of these systems as part of their development activity becomes more important.

Despite the emphasis on CSCW in the latter parts of this thesis, the work presented in the first part has its origins in the human-computer interaction (HCI) research field. An even more precise characterisation within HCI would be *research on usability engineering*, its organisation; processes and working practices. In this thesis, the supportive characteristic of the resulting CSCW application (and corresponding conceptual model) is restricted to the support for usability engineering practices in user-centred development work: How can product developers manage user-centred issues (that have been gathered with user-centred development methods) shared with each other in a decentralized and distributed development organisation? From this standpoint, this work contributes to the usability engineering and HCI fields.

As usability engineering should appear as an integral part of development work[13], background from *design and development research* contributes to this work and vice versa. When this is combined with the CSCW approach, we may expect that this work contributes to design research, particularly on collaborative computer-supported design.

Relevance of this work to industry and practice. In addition to the relevance for the academic community, the theme of this research is supposed to gain also practical results. This is done by applying scientific research methods and academic discussion to industry-relevant questions. From the real-life engineering standpoint, practical and evaluated results are an important way to improve development and design activities. This research aims at creating a conceptual model and supporting instrumentation that can act as a medium in incorporating the management of user-related issues within product development. As user-related issues are often discussed in multi-disciplinary teams, different types of catalyst tools may be applied to support the communication of these interdisciplinary issues. This work aims at presenting one such a tool and framework.

[13] Maguire (2000) presents that "Usability engineers are still not part of any product development, usability engineering methods are still the exception and not the rule, and usability engineers are still not seen as main trigger of innovation". Implicitly, this statement points out the desire that usability engineering should be an integral part of everyday development work.

From the standpoint of computer support for usability engineering, the question addresses also an interesting topic. There is a lot of activity in the research of design methods for user-centred development, but not that much research about the instrumentation and support of these methods for practitioners[14]. One example of increasing interest towards this issue is the project proposal by a research group in the University of Dundee in UK about "Computer-based support tools for usability engineering"[15]. This project proposal approaches the topic from the standpoint of computer-based interviewing.

In order to get practical results that are applicable in real settings, we need to create a model from the basis of this theoretical information supported by empirical observations. To prove the utility of such a model we need to implement the model in the form of a working procedure that makes use of supporting methods and tools that enable the management of user-centred issues.

2.4 Structure of This Thesis

The issues contributing to the research questions are dealt with in following sections of this work:

What is user-oriented product development? What is position and relation of user-oriented, or user-centred, activities to other product development activities and tasks? What information are produced and used in user-centred product development?	3.1 Usability - The Underlying Concept for User-Orientation 3.2 Characteristics of Product Development for User-Centred Design 3.3 User Centred Design: Processes and Method 4 Architecture of User-Oriented Development Organisation
How is user-centred product development defined and arranged in practice? What is its relation to other product development activities and tasks?	3.4 Empirical Background: Organisation of Product Development Activities

[14] Maguire (2000) presents an overview of six EU projects (Esprit & TAP) that have addressed usability during the last 15 years in Europe. These projects include HUFIT, MUSIC, MAPI, INUSE, RESPECT, and EMMUS. In many ways, the projects have had an emphasis on the definition of usability metrics and methods (incl. processes) for usability engineering. Computer-supported instrumentation for these methods and processes has not been mentioned in this short paper showing that the topic has not been addressed in the analysed EU projects.

[15] http://www.computing.dundee.ac.uk/projects/use-it/ (checked 8-Jul-2004)

What is information management that supports the activities of a development organisation? How can information support be arranged in a contemporary product development environment?	5.1 Information Support 5.2 Collaborative Information Support Tools for Usability Engineering
What kind of services should an information support environment for user-oriented development provide? What is an appropriate structure and technical realisation of an information support system that is feasible to implement and test in contemporary product development environments?	5.3 The Knowledge Storage Concept 5.4 Implementation of the prototype 5.5 Proof-of-concept evaluation: Pilot Case 5.6 Conclusions about Knowledge Storage

In chapter 4, an overall conceptual model about positioning the information support mechanism in product development activities is outlined. Some contribution for academia can be seen to be in the "architecture of user-oriented development organisation" presented in chapter 4 even though its main purpose is to serve the positioning of the Knowledge storage concept and prototype implementation in the surrounding product development activity environment. Chapter 5 provides the answer to the main research question. This chapter contains the description of the development, construction and evaluation of an information system called "Knowledge Storage" that makes use of the findings in the previous chapters.

Each of the chapters with empirical results (3.4.1 Current practices in product development, 3.4.2 Case "MyPhone", and 3.4.3 User-centred development activities) is regarded as somewhat independent from each other. Each chapter contributes to the model that is constructed in chapter 5. Therefore, each empirical chapter contains the conclusions on its own and chapter 5 acts as the conclusion chapter from all these empirical findings. The main "Conclusions" chapter combines all the work done together. The overall construction and conclusions are discussed against the research question in the last chapter, "Discussion", where also directions for future research are outlined.

3 Construction Elements for User-Oriented Product Development

Product development is the context in which new ideas and innovations of future devices, applications, tools, and services are created and refined. The intensity of this activity is rising all the time. We are facing the ever-increasing possibilities that technological progress offers. As new technologies emerge, possibilities for new innovations and products rise. However, these new possibilities need to be identified and developed, and they are not commercially ready to implement and do marketing until a need for these applications emerges or has been identified, too. From the standpoint of product development, the forthcoming end-users and other utilisers of a product express these needs. The gathering and verification of these needs and their translation into applicable product features is a process that can be seen to require explicit focus in development projects and process. User-orientation in a development organisation provides one approach. This issue can be seen to get realised through user-centredness, user-centred development practices.

User-centred software design has been an issue of growing interest during the last two decades. An indicator of this is the existence and growth of the ACM SIGCHI (Association for Computing Machinery / Special Interest Group on Computer-Human Interaction) that was established in the early 1980's (with a predecessor called SIGSOC; the social aspects of computing; since 1969) (Borman 1996). The concept of usability engineering was launched in the mid 1980's (usability: Eason 1984; usability engineering: Whiteside & al. 1988). Since then, the concept of usability (now defined e.g. in the ISO9241-11 standard) has been of central importance for user-centred software development.

Lots of research on methods, tools and practices in this field of user-centred software design has been done during this time (see e.g. Gulliksen 1996). This research is considered to provide basis to the research question “What is user-centred development?” Many methods and tools have been, however, presented within academia. The development organisations and individual designers are, however, expected to be the main users of these tools. This stresses for practical considerations about the utilisation of the methods and modelling of real-life development activities.

The research issue “How is user-centred product development defined and arranged in practice” aims at addressing the empirical issues of user-centred development activities and is addressed in the empirical part of this work. Following restricting questions are presented to provide boundaries for the main question:

- What kind of topics are to be addressed in user-centred development / usability engineering?
- What are the phases of the development process in which the usability methods can be used? What does user-centredness or usability imply for the process?
- What stakeholder groups are involved in product development?
- What does usability approach mean in the levels of individual designer and development organisation?

The underlying models to these questions above are dealt with in this chapter. This part of the study is realised with literature review and gathering existing experience from previous studies. This background information is then applied in the empirical parts to support the construction of the model for information support for systematic user-centred product development work.

The methods for capturing user perspective and defining the relevant user-originated product features have been popular among the methodology developers of user-centred design. As the use of these methods gets more mature and their use expands, the management of the information produced with these methods and the appropriate working practices (processes, communication means, etc.) becomes more important. The research question about information support addresses the facilitation of these activities. Information support is addressed first mainly via literature review but the construction of the information support system ("Knowledge Storage") at the end of this thesis aims at packaging the concepts into executable form.

3.1 Usability - The Underlying Concept for User-Orientation

In order to provide a focus for the information space about user-centred issues and about user-centred activities, it must be defined, what usability and user-centredness mean. User-centredness may be defined as the approach that provides the practices that lead the development towards usable products. Usability, then, defines the contents of this activity and basics for the user-centred information space.

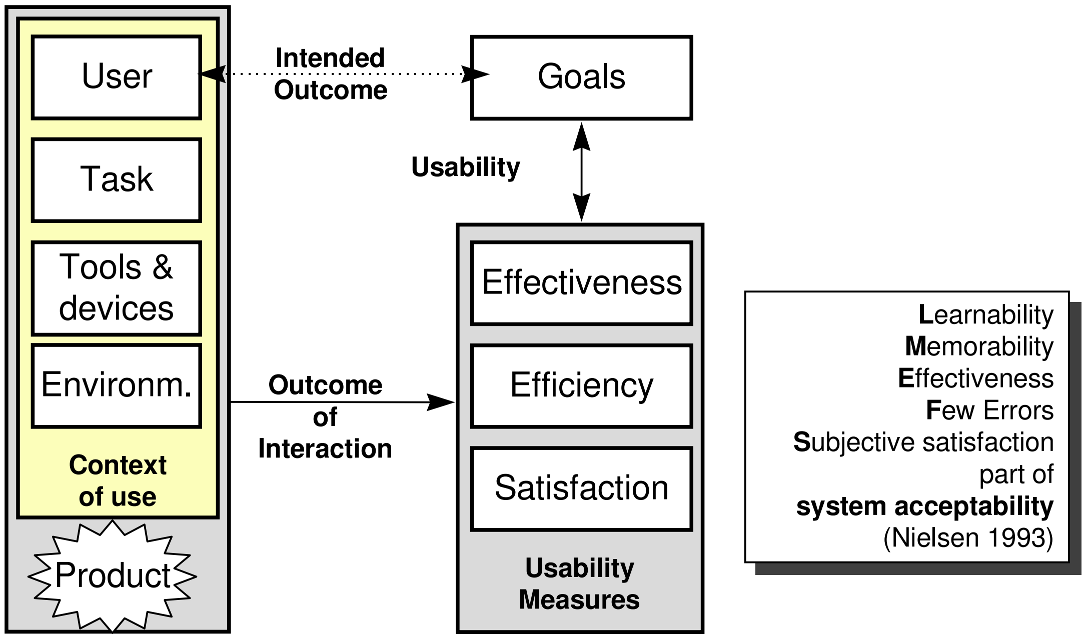


Figure 1. The framework for usability design and evaluation according to ISO 9241-11 (DIS, 1996). Nielsen (1993; on the right side of the figure) has defined five usability evaluation attributes that can be used to point out the effect of usability design.

ISO 9241 defines the conceptual framework for usability (see figure 1 above). In short, usability is defined as

The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

The design of usability consists of consideration about user, user's task, tools and devices that are used, and the physical and social (including organisational) environment. These elements need to be considered when designing interactive applications. The product that is going to be developed will be operating in this environment. Should a development organisation comply with this standard, the consideration of these issues must happen explicitly. The management of these elements of usability can be done, for example, according to the instructions given by ISO 13407 and Usability Maturity Model that are presented later in this work.

Implication for this work. The information support mechanism that is considered in this research must be able to provide systematic means for the management of "user-centred" information categories mentioned in ISO 9241. From the standpoint of user-centred development work, the information support mechanism must be able to point out these important aspects about user, task, tools and user's environment for the development practitioners. The concept of usability contributes to research question "What is user-centred product development" – but does not, of course, cover it comprehensively.

3.2 Characteristics of Product Development for User-Centred Design

As this study aims at providing a model that embeds the management of user-centred issues and usability design activities into product development practices, it is required to get a view of product development, its activities and organisation. An information support mechanism must be able to comply with the structure, stakeholders, organisation, tools, and existing systematics like quality systems of product development. This chapter outlines these requirements set by product development environment for the information support system.

3.2.1 Basis for Product Development

Product development is a collection of activities that aim to the creation of a commercially available product. According to a basic book for practitioners use, these activities cover the research and design of technological and economic issues (Kangasluoma 1979). The basis for product development lays in the new possibilities that technology offers. But in addition to that, the users' (consumers') needs serve as a directing force (see figure 2). Research on market events and trends gives information about the current and near term needs that customers, consumers – or users have.

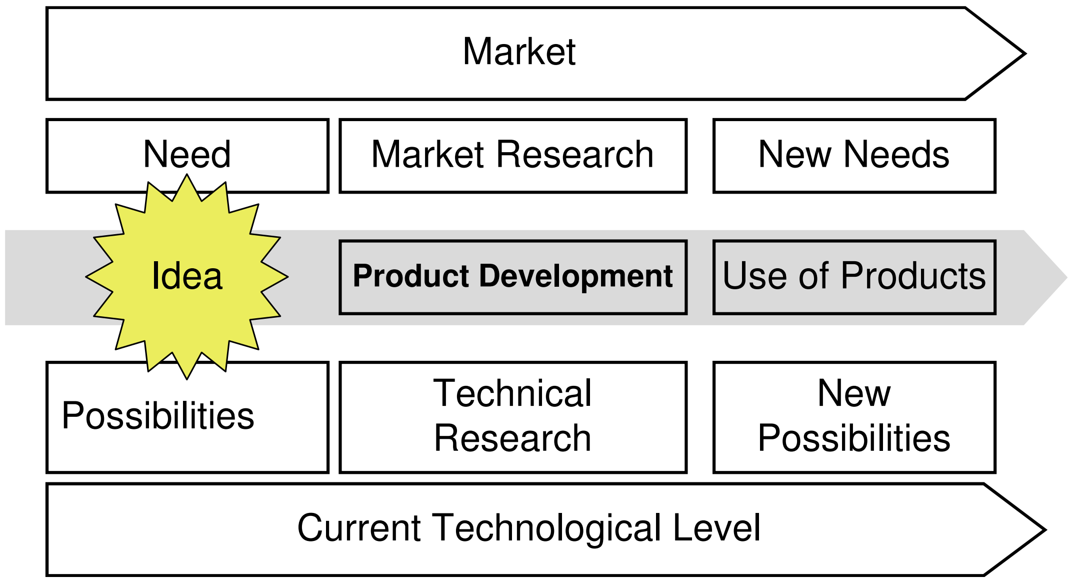


Figure 2. Product development and its environment (Jaakkola & Tunkelo 1987). Product development is driven by the new technological possibilities but market needs must be taken carefully into account for commercial success. One part of "market needs" can be seen to be the users' demands and requirements. With this approach, user-centred design and usability engineering can be included in the development work and respective processes.

Jaakkola & Tunkelo (1987) refer to an early study named *Successful Industrial Innovations* by Marquis Myers (1969). Already that study reported that successful innovations base themselves to three sources: technical possibilities (25%), production improvement (30%) and possibilities in market (45%). Based on the figures presented with the sources, it seems that best possibilities for good results may be gained through close analysis of possibilities in market. From the standpoint of user-centred design, this can be translated to include the end-users' needs, requirements and wishes.

In addition to right sources of information, characteristics of successful product development projects include (Jaakkola & Tunkelo 1987):

- Focusing to customer needs over development organisation's own internal assumptions
- Close co-operation with customers instead of indirect communication only (not to rely on market research only)
- Only technically completed products are delivered to customers, not unfinished ones
- Active support from the management.

Focusing to end-users is more than just getting the product a good look and feel. It is an essential source for successful products. The internal factors of the developing organisation do not have that much power to form the basis for successful product. However, an exception to this seems to be the expertise of the development staff (Kulvik 1977). From one standpoint, this argument implicitly emphasizes the importance of internal (in development organisation) creation of product requirements, specifications and implementation. This approach could lead to traditional technical focus in the development if it is put too much weight on. Another standpoint can be presented that emphasizes the importance of the high expertise in a multi-disciplinary development organisation. This approach underlines the importance of communication and knowledge sharing between development specialists that have different background from each other. Developers need smooth communication mechanisms that enable them to exchange information about important and relevant development topics. Of course, this requires that issues that are considered important and relevant are accepted as such in the development organisation.

As usability stresses for user requirements, *user involvement* (or "end-user participation") can be treated as one meaningful attribute of development work (Booth 1989). The involvement of users in the process may even provide simple evaluation criteria for the quality of the process. Booth (1989) presents levels that describe a user-centred design process (a more detailed and thorough human-centred assessment model called "Usability Maturity Model" is described later in this study):

1. User is thought of in the development work indirectly (via usability rules)
2. Users participate in the process via commenting the plans
3. Users test prototypes and give feedback from them
4. Users iteratively support the process. Users give feedback from prototypes and present development ideas.

User involvement is often a prerequisite in the case of tailored products created directly to specific customers and users. In these kinds of situations, user requirements play a significant role in the creation of product specifications. User involvement may happen in various ways, and the levels presented by Booth indicate that the more active users are in the process, the better it gets from the standpoint of user-centred design. Active involvement may, however, have its drawbacks as well: active user representatives in the development project turn after a while more towards designers and somewhat lose their ability to evaluate different possibilities purely from the standpoint of the task.

It does not seem to be an uncommon situation that product developers are seldom or have never been in touch with the actual users of the product that they are developing; this has been reported already by Gould (1988)[16]. If the development organisation has a fluent process for the documentation and communication of end-user demands, this may not be a problem. But if such a process does not exist, it can be expected that the product does not fully meet the user needs.

Even though user needs serve as an important background issue for product development, several other aspects like sales channels and product logistics, production possibilities and management of the business area have to be considered (Jaakkola & Tunkelo 1987). These include issues dealing with internal topics of the development organisation.

Ulrich and Eppinger (1995) present important characteristics of successful product development that address these internal topics of the development organisation. The characteristics are:

- **Product quality**. How good is the product resulting from the development effort? Does it satisfy customer needs? Is it robust and reliable?
- **Product cost**. What is the manufacturing cost of the product?
- **Development time**. How quickly did the team complete the product development effort?
- **Development cost**. How much did the firm have to spend to develop the product?
- **Development capability**. Are the team and the firm better able to develop future products as a result of their experience with a product development project?

Depending on the approach that we take in the user-centred development, the characteristics presented by Ulrich and Eppinger can be applied fully or just partially. From *product oriented approach*, the only applicable characteristic is product quality. If we define usability as a product feature, the quality of the product depends on its usability. In this product oriented approach we can verify the usability of the product by,

[16] A personal discussion that I had with the managing director of small company pointed out this phenomenon as well: The company was about to purchase a production control system. During the discussions about requirements for tailored functionality to be included in the system, the vendor explicitly denied direct discussions and interaction between the end-users in the company and the implementation developers in the system vendor. A basic argument for this was that the implementation designers require a peaceful environment to do their work. Direct interaction with the designers was considered to break this peace and lead to unmanaged further development of the system.

for instance, applying usability tests that have been designed according to realistic task scenarios. Using the list above, usability does not directly concern product cost as it is more a design feature than a component feature (however, usability may imply for specific components that provide for instance better contrast, better gripping features etc.). From the standpoint of the user, development cost, development time and development capability are not product features; product features are just the ones that appear on the physical (or logical) product itself.

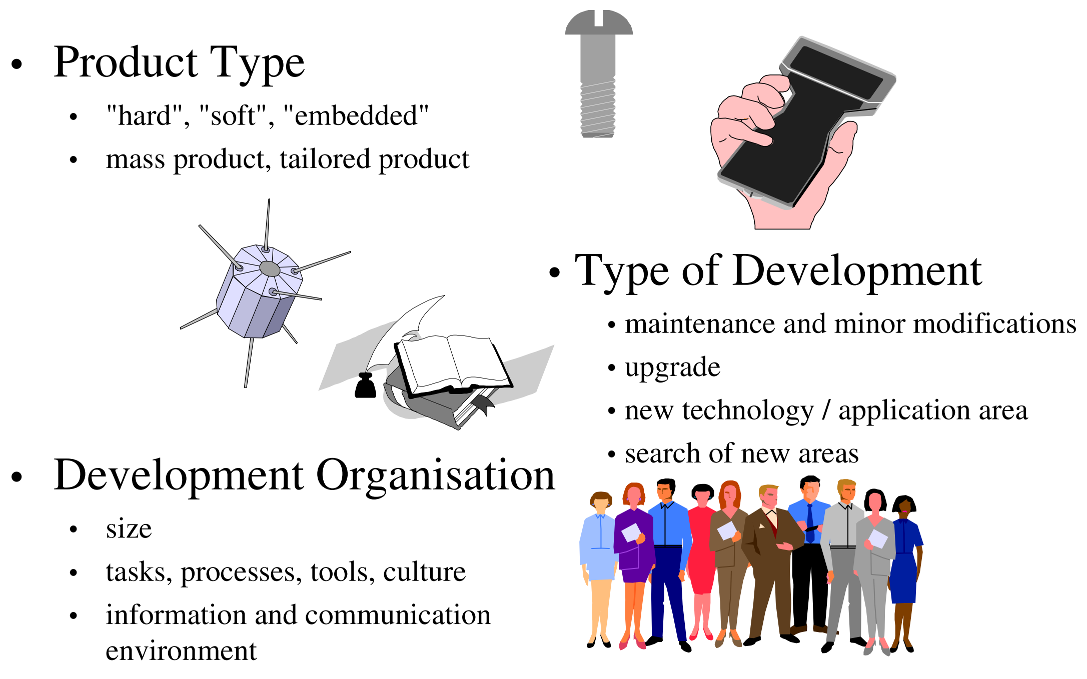
A more comprehensive point of view takes into account the development of the product as well. From *process oriented approach*, we can address all of the characteristics mentioned above. *User-centred design*, as design activity that requires resources, is certainly something that affects development time and development costs. And, indeed, user-centred design sets requirements for the development capability. This process-oriented approach provides a more complete view to address usability and human-centredness from the standpoint of the developing organisation. Process-oriented approach emphasises the idea of "qualified development results in qualified products". Whether that is true or not, it can be argued that well-established development processes have greater possibilities to produce qualified results than tangled ones.

Implication for this work. The basics of product development indicate that user-centred approach has a lot to give to development work. User-centred approach provides support for successful development work as it is heavily based on customer needs, demands and requirements. The definition of usability in ISO 9241 takes customer requirements into account from the standpoint of users' ordinary working conditions. These requirements can be used to complement and provide structure for other customer requirements that relate to market requirements (competitors, market shares) and current technological level (product components, emerging technologies).

3.2.2 Characteristics and Attributes of Product Development

Product development may be characterised with several attributes that affect its arrangements. Figure 3 summarises the attributes that are presented next.

Differences in product types have an effect on development work. From the usability point of view, the creation of purely technical non-interactive component may not require attention as long as it does not affect end-users' working with the device. Thus, usability needs to be focused to only in the case of interactive devices and components



- Product Type
  - "hard", "soft", "embedded"
  - mass product, tailored product
- Type of Development
  - maintenance and minor modifications
  - upgrade
  - new technology / application area
  - search of new areas
- Development Organisation
  - size
  - tasks, processes, tools, culture
  - information and communication environment

Figure 3. Characteristics that have an effect on the arrangements of product development.

that have an effect on the situation on use[17]. There has been also discussion about the differences between the creation of software and embedded and completely mechanical devices. The development of embedded and mechanical devices requires additional consideration from the standpoint of physical components compared to software. And it is evident that the restrictions in the development of a device with physical dimensions are often greater than when it is a question of software. But when it is a question of modelling and describing the user, users' tasks, current tools and devices used by end-users, and the physical and social environment, the methods and documentation techniques may not differ that much depending on the type of the product under development.

All product development does not aim to the creation a completely new product. Development effort is put to products that are in *different phases of the product life cycle*. Most of the effort in a development organisation is typically put to support of existing products that are already in the market (81% of costs of development organisation). This consists of maintenance like work (small improvements, 34%) and

[17] Even technical components tend to have nowadays some kinds of "configuration interfaces" that do require consideration from the usability standpoint.

enhancements to existing products (47%). Research and development for new products takes 12% of the costs of the development organisation and the search for completely new technologies and business areas covers 7% of the costs. (Jackson 1983, cf. Jaakkola & Tunkelo 1987)

An example of different products can be as follows. When it is as question of creating a new type of hammer or a new generation mobile service application, the tools and methods for the development are different, and especially the emphasis on different important issues. In the case of the hammer, characteristics of the material and requirements for strength are key issues and must indeed be treated as such. In the creation of the interactive new generation mobile network based service, process descriptions of the users' tasks and their wishes for task improvement need to be carefully gathered. In the first case material databases may be a key source of information. In the latter case key issues may be the results of user interviews. Other related topics to product differences include issues like possibilities for the creation of early functional prototypes and models. The creation of a physical model of the hammer may be a lot demanding and expensive when compared to provide applications that can run on computer screen only.

One attribute of product development is the *organisation* that does the work. This attribute can obviously be separated into several parts including the size of the organisation, developers' capabilities and their expertise mixture. Small development organisations have more restricted abilities to do comprehensive plans and analyses than larger ones. They can, however, react rapidly to emerged needs and adjust their products to meet these needs. Other important factors in product development organisation are also working methods and communications structures, which gain more importance as the organisation grows.

Implication for this work. The organisation of product development sets important requirements for information support systems. This part provides the general underlying conceptual basis about the issue “what is product development”. An information support system must take into account and arrange following concepts: the type of the product (hardware, software), the type of the development project (version upgrade, “from the scratch” product, platform development, exploration of new technologies), and the organisation (centralised, distributed, size, stakeholders, process). These characteristics are addressed in more detail in the following chapters.

3.2.3 Product Development Process

A traditional product development process, at least in the field of software engineering, is known as "the waterfall model" (see figure 4). This model has been criticised for being too straightforward with no loops back and resulting in products with features in an unwanted form. In the field of usability engineering, iterative development has been long considered as a more user-centred approach (Gould & Lewis 1985). User participation has typically happened in the beginning of the project without major loops back to earlier stages.

When the waterfall model is being applied, it is expected that the product under construction can be thoroughly and completely defined before the technical development and implementation starts. In such a case, technical developers who create the implementation just need to follow the specifications that are simply “handed over” to them. In a strict sense, the original non-iterative waterfall process model does not tolerate well intermediate and late changes to product requirements and specifications as there is a need for rather early process milestone “specification freeze”. From this

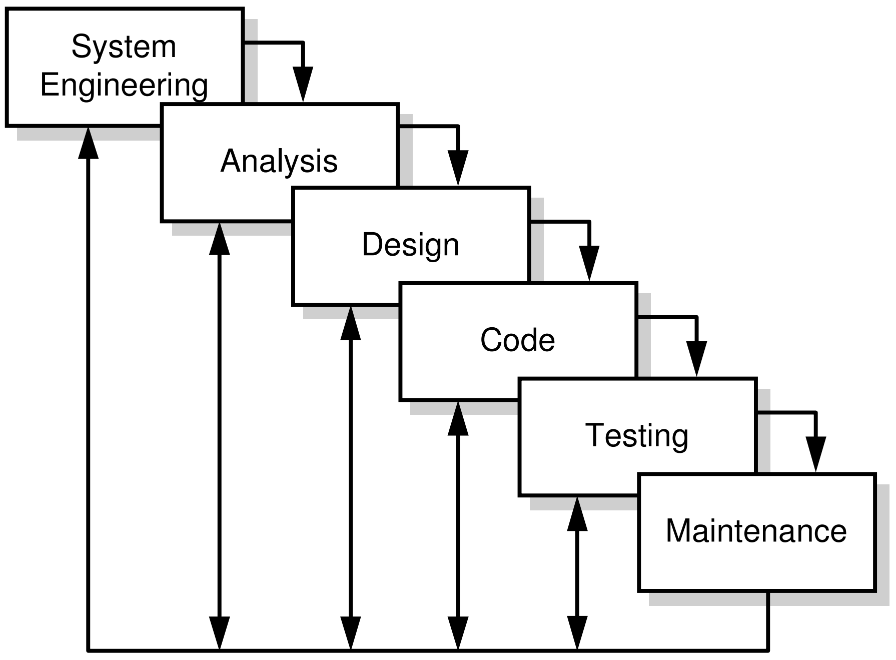


Figure 4. The Waterfall model (Pressman 1987, see also Sommerville 1985). This model originally assumes that product specifications can be fully defined and freezed before the actual implementation begins. The arrows that provide loopbacks to earlier stages indicate the need for iterations as well as refinements and changes even in the late stages of the process.

standpoint, the waterfall model does imply that the technical realisation and implementation of the implemented product or its development version does not have an effect on the requirements and specifications. The more iterative models with loop backs to earlier stages in the process aim at addressing this challenge.

As product specifications are created by the developing organisation, the specifications are easily very technical by nature. These may not always be correctly understood by all stakeholder groups, end-users and end-user organisations tend to have problems in transcribing the eight-hundred pages long technical specifications into practical presentation. This is a challenge for the transformation and fluent communication of the requirements and specifications. This "communication demand" in the development process emphasizes the importance of transformation and systematic management of design information that can be of very versatile by nature.

Additionally, the more iterative the process becomes, the more discussion takes place during the process. This dialog between the participants of the development work produces large amounts of additional data, viewpoints, and requirements that need to be captured, analysed and put into effect in an appropriate way. As an example of a more complex realisation of a development process there is the model presented by Rauterberg (1991; see figure 7). We can see that here is a need for improved and enhanced development time dialog facilitation and more generally information management.

Even though the waterfall model has faced a lot of criticism, generic development process descriptions tend to contain the same phases but with iterative loops and more intensive user participation. Ulrich and Eppinger (1995) present a generic model about product development process (see figure 5). It resembles the structure of the waterfall model and incorporates focus on user demands to the process. Ulrich and Eppinger do not explicitly present a process that has loopbacks between the different phases. Instead, they present methods that are supposed to reveal precise user needs effectively.

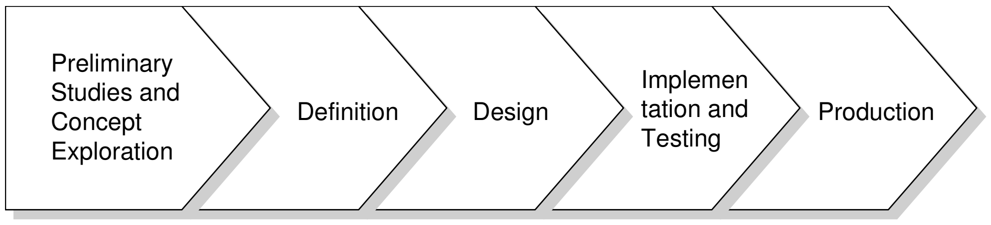

Preliminary Studies and Concept Exploration
Definition
Design
Implementation and Testing
Production

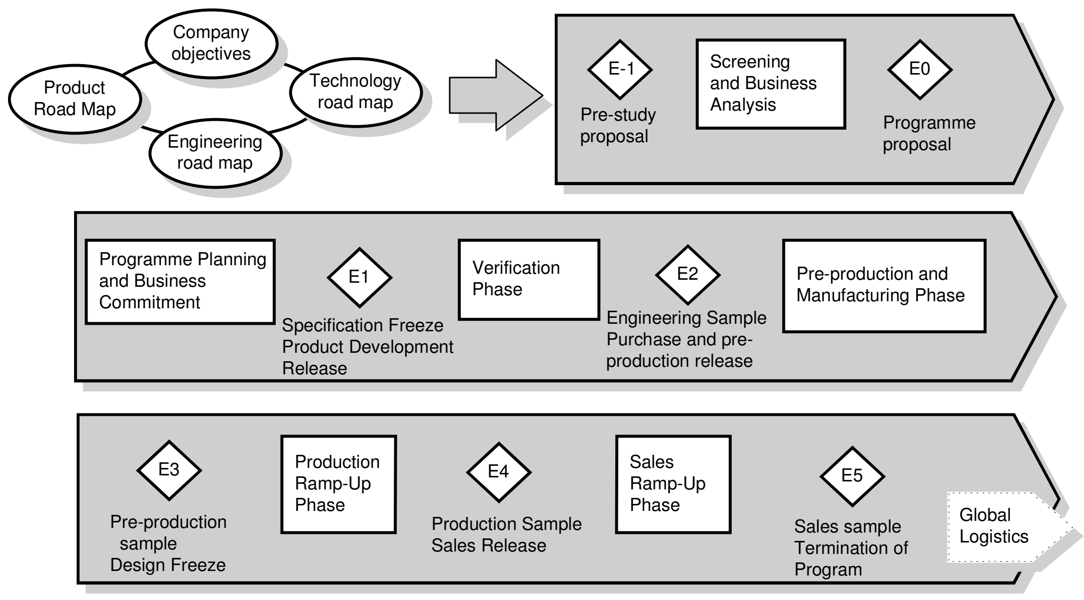
Figure 5. A generic development process (Ulrich & Eppinger 1995).

Ulrich & Eppinger have divided product development process roughly to five phases (see figure 5): preliminary studies and concept exploration, product definition, detail design, implementation and testing, and production. This is quite close to the classic life cycle ("the waterfall model") presented by Pressman (1987). It consists of system engineering, analysis, design, code, testing, and maintenance. Ulrich & Eppinger's model can be seen to reflect the development of a traditional "physical product" whereas Pressman presents a model for software engineering.

Despite criticism towards waterfall model, we may argue that all design work contains the activities that the model points out (at least analysis, design, implementation, testing). They may, however, not occur in the pre-defined straightforward order. The iterative development model suggests that repetitive loops must occur in development. This way the specifications and detailed implementations get settled more precisely and design results are better. The spiral model (Boehm 1988) encompasses features of the phased lifecycle as well as the prototype lifecycle. It uses the stages from the waterfall model. The spiral model includes four major elements, which include planning, risk analysis, engineering, and customer evaluation.

As a rough combination of the phases presented above, we may form the sections of generic development as follows: (1) conceptual design, (2) detail design and (3) implementation, and (4) testing. The usability approach stresses for an additional section (5) follow-up as usability gets verified primarily in the users' daily activities. By mentioning that usability can be reliably detected only in the ordinary activities with the product in real environment, this last phase, or section, becomes evident.

More detailed process descriptions of the development processes reveal a bit of the complexity that takes place in the practical implementation of the generic process models. Valjus (1994) has presented a concurrent engineering development process of an embedded consumer product (see figure 6). The actual development process starts from preliminary specifications whose possibilities at market are evaluated. The process



Company objectives
Product Road Map
Technology road map
Engineering road map
E-1
Pre-study proposal
Screening and Business Analysis
E0
Programme proposal
Programme Planning and Business Commitment
E1
Specification Freeze Product Development Release
Verification Phase
E2
Engineering Sample Purchase and pre-production release
Pre-production and Manufacturing Phase
E3
Pre-production sample Design Freeze
Production Ramp-Up Phase
E4
Production Sample Sales Release
Sales Ramp-Up Phase
E5
Sales sample Termination of Program
Global Logistics

Figure 6. Development of an embedded product with software, electronic, electric, and mechanical parts (Valjus 1994).

terminates, when the production has started and the sales organisation has received its first products.

In this model, product specifications are frozen before the technical development work begins. During the technical development, no modifications are made to the product. If needs for modifications arise, they are put aside and implemented perhaps in the next version. The development cycle in this case is supposed to be rather short, from six months to one year.

Valjus (1994) presents that concurrent engineering and global logistics together with the detection and analysis of real customer needs are the key factors that have had an effect on this process model. Following questions are considered as key issues behind this process model:

- How to ascertain the inclusion of real customer needs
- How to speed up product development cycles
- How to optimize the product from the standpoint of logistics
- How to make sure that the customer is satisfied after the delivery of the product

Even though the waterfall model has been used for a long time in software development, iterative process models seem to be common as well (Rauterberg 1991, Hix & Hartson 1993). In software development, prototypes can be rather easily designed to appear like final products. In some cases, prototypes may even be used as the basis of the final software itself. In the case of products with physical dimensions, computer based prototypes may not give that good impression about the product under development to the end-user.

Rauterberg (1991; see figure 7) presents that an iterative software development process consists of four phases: (1) analysis, (2) specification, (3) implementation and (4) trial and assessment. Users' tasks and requirements are gathered in the analysis phase. In the specification phase, not only the visible user interface is created but also descriptions of contextual and organisational interfaces. The software application is created during the implementation phase. In the last phase, trial and assessment, the technical correctness is tested and the accomplishment of user requirements is verified.

Characteristic to all presented product development models is that they terminate when the production starts or to first sales samples are released. The analysis of the result in real settings is often beyond development work. As the gathering of end-user requirements and experiences is a central issue in the development of new versions and completely new products, a communication procedure between different parts of the developing and selling organisation must be created. This can be done as part of a quality system.

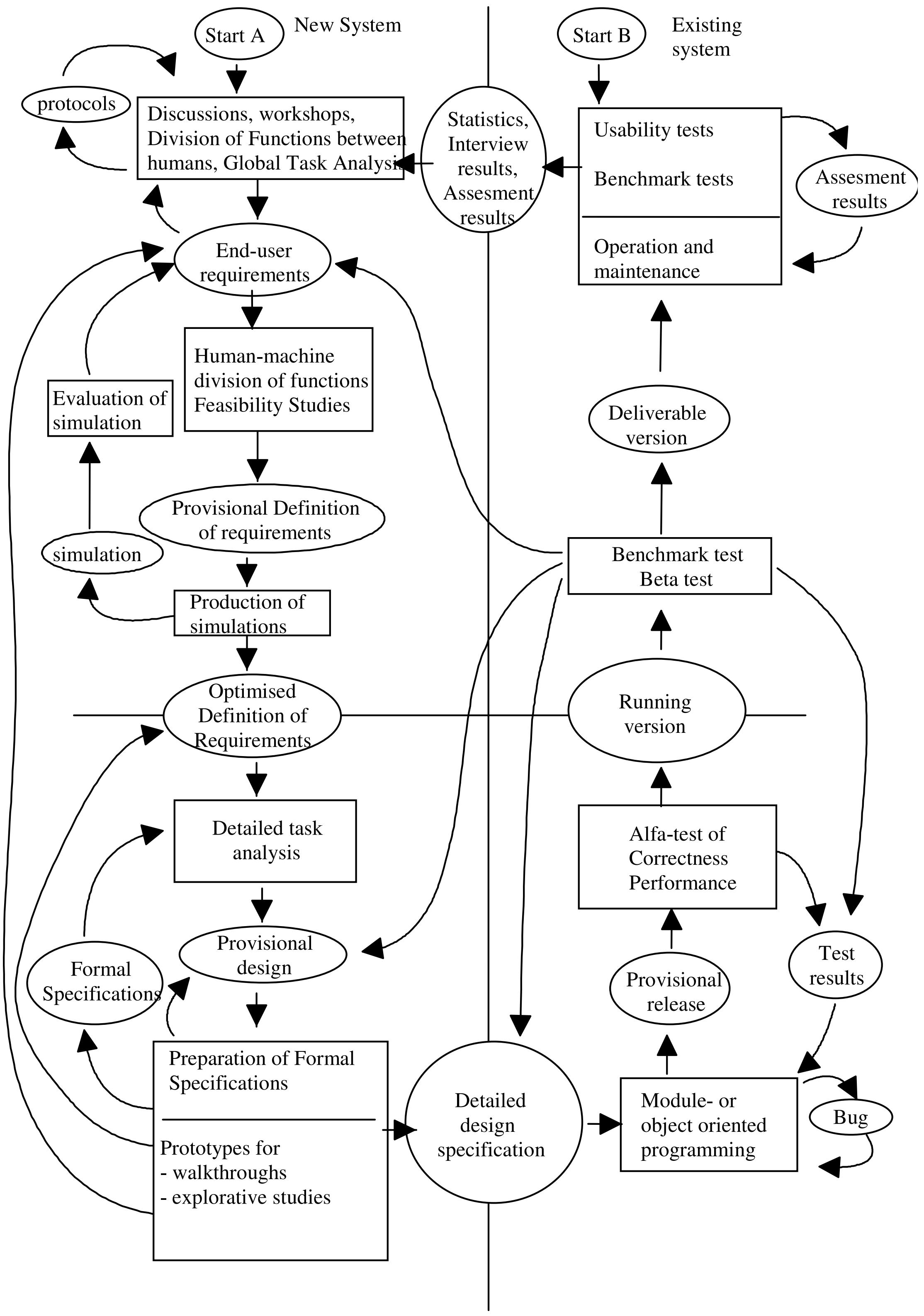

Start A
New System
Start B
Existing system
protocols
Discussions, workshops, Division of Functions between humans, Global Task Analysis
Statistics, Interview results, Assesment results
Usability tests
Benchmark tests
Operation and maintenance
Assesment results
End-user requirements
Human-machine division of functions Feasibility Studies
Evaluation of simulation
Deliverable version
Provisional Definition of requirements
simulation
Benchmark test Beta test
Production of simulations
Optimised Definition of Requirements
Running version
Detailed task analysis
Alfa-test of Correctness Performance
Provisional design
Formal Specifications
Test results
Provisional release
Preparation of Formal Specifications
Prototypes for
- walkthroughs
- explorative studies
Detailed design specification
Module- or object oriented programming
Bug

Figure 7. Iterative and user oriented software development process (Rauterberg 1991).

Implication for this work. Process models provide systematic action sequences for development work. As most development work is done according to some kind of process model, an information support system should provide "context awareness"[18] from the standpoint of the phase of the development process. Different kinds of issues are addressed in different stages of the development project. These issues should be attached to the stages in order to provide the possibility to track and review "stage specific information". Additionally, process models conform to the quality system of the particular organisation. An implementation of a process model can provide support for quality systems as well. This makes the use of an information support system more applicable and practical in the development organisation. The description of product development processes defines partially the issue "what is product development". Process models provide basics for systematic development activities.

3.2.4 Product Development and Quality Systems

Certified quality assurance systems are a common requirement nowadays. Especially companies in the business-to-business area have pressure towards a certified quality system (ISO 9001, Yli-Olli & Hokkanen 1991). This concerns more and more development organisations.

Basically, a quality system requires that a company has procedures for assuring that its work processes produce what they are meant to produce and that it is properly managed ("Quality is conformance to specification/documentation"; Bevan 1997). This way quality can be characterised as knowledge about the operations and happenings. ISO 9000 based quality systems do not require a certain process model for development work but there seems to be a tendency to traditional waterfall-like process (Yli-Olli & Hokkanen 1991).

[18] Here, *context awareness* means that the information support system is capable of following the context of the development project i.e. the progress of the project by e.g. detecting the stage of the project (not necessarily automatically, however) or the role (that may change between projects or project stages) that the particular person has in the project. This is a restricted view of the overall *context* that surrounds the different collaborating stakeholders in the development environment. Context awareness is dealt with a more technical viewpoint in the research area of *ubiquitous computing* (Weiser 1993). Schmidt & al. (1999) present that context consists of human factors (user, social environment, task) and physical environment (conditions, infrastructure, location; light, pressure, acceleration, audio, temperature, time). A recent viewpoint of context is the *social context* that is addressed by e.g. Tamminen & al. (2004).

In the context of product development, the important issue in a quality system is that the organisation is aware of the current practices and construction of the product. The practices and changes in product construction have to be accepted in a defined way, for example in review meetings that are held regularly after each stage of the process. In the meetings the accomplishment of the goals is verified and decisions of further actions are made.

Quality systems can provide descriptions of development and communication workflow procedures. Thus, a quality system with corresponding quality manuals can support the gathering of user needs and correct interpretation and communication of these needs.

Implication for this work. As quality systems require detailed documentation about all actions and activities that take place during development, the use of an information support system that records development actions and provides means for side-work documentation facilitates the work conducted by product developers. An important aspect in the introduction of new tools into an organisation is the ease of use and the possibility to integrate existing systems. Should an information support system integrate the organisation's quality system in it, it makes the work more manageable and systematic. Quality documents should be contained in the information space provided by such a system and the system may even propose and direct all work to be done according to quality instructions. This issue provides a partial answer to research question "what does systematic development mean".

3.2.5 Stakeholders in Product Development

Product development is activity that requires different people and skills. Several groups of people and sources of information have an effect on it. These different groups point out different types of needs, requirements and expectations towards the development work. Some of these groups and viewpoints are presented in figure 8 that also emphasizes the end-users' role and respective user-centred analysis and evaluation methods. Representatives of marketing, design and manufacturing organisations are typical participants (Ulrich & Eppinger 1995). In larger development projects, this multi-party nature may create problems (e.g. in communication) but it can also be a strength if people with multiple disciplines and skills can contribute to it. Sharp & al. (1999) present that the baseline stakeholders in the requirements engineering process are users (direct and indirect), developers, legislators (professional bodies, government agencies, trade unions etc.), and decision-makers (management, financial controllers).

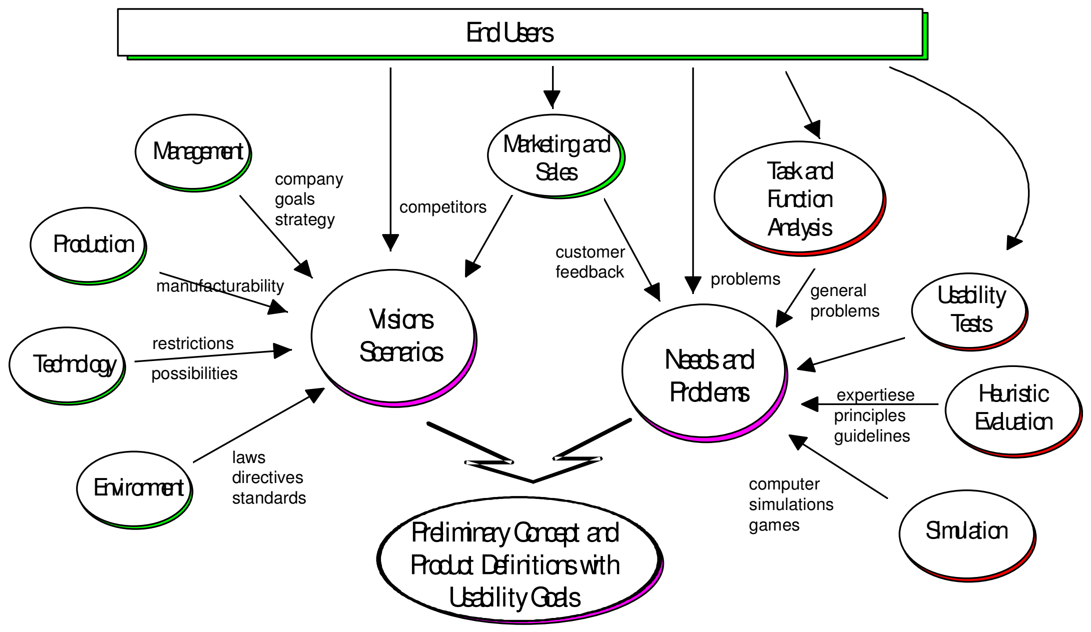


Figure 8. Stakeholder groups and information sources in product development. The needs and requirements from different stakeholder groups need to be communicated, combined, and consolidated during the implementation of a product development project. One part of this is covered by the user-centred design and development methods (Finnish Federation of Electrical and Electronics Industry 1994).

As the trite saying expresses, the customer should be in the focus of the development work. Thus, customers and end users must be treated as "key players". Their needs and wishes need to be carefully listened to and this information must be delivered further to product designers and other interest groups within the development organisation. Customers are, however, not always the users. The distinction between the customer and user is sometimes obscure, but perhaps the illustration of a father buying a skateboard to his son may clarify this issue: The qualifications of a good skateboard are different to the father than to the son. For the father, low price may be desirable as well as durability. For the son, the fancy colour of the grip, the pictures and stickers in the bottom of the board as well as the construction of the trucks (for great “railing” experience) and bearings are much more important. The customer may not always know what the real utilisation of the product means. In addition to listening to the customer, there is the need to listen to the real end-users during requirements gathering and development.

Information about users' demands can be gathered by several different means. One way to gather it is in marketing and sales happenings like trade shows and exhibitions. This activity involves, of course, people in sales and marketing. Within these happenings, users pose questions and set up more or less clear and bright wishes about solutions for their current problems. These wishes need to be listened to carefully and the ideas need to be put into further examination by other development representatives. Another task for sales and marketing is to keep in touch with current state of the industry and to report major changes to product development. The co-operation between marketing and product development is seen very important (Kangasluoma 1977). Gathering information for product development via sales and marketing only is obviously not a way to produce a comprehensive list of requirements for a product under development. These stakeholders do, however, have important pieces of information that may well be attached to the collection of requirements set by other stakeholders, too.

Customer oriented development and production (Luhtala & al. 1994) is a major issue in commercial environment. All operations should be evaluated through the value added from customer's perspective. However, we need to make a distinction between a customer and a user. Customer is not always the user of the product. In the case of industrial acquisition, specialised buying staff or management may do the decision about an investment of a product or a system. The real end user requirements expressed to developers may be strongly biased as they go through the filtering stakeholders. In order to create successful products, both the buyers and end-users point-of-view needs to be examined.

Users' complaints approach the end-user issues from another perspective that is more practical and detailed. These are always critical incidents that must be recorded carefully by the developing organisation. Some kind of complaint handling procedure must exist in every organisation. Complaints can be used both for functional error detection of the particular product and for the creation of new innovations.

It can not be expected that users will always explicitly communicate their needs and wishes thoroughly to the representatives of the development organisation. A great amount of latent and hidden needs exist, too (Ulrich and Eppinger 1995). To get in touch with these, effort on direct user involvement should be thought of. One way to do this is with the aid of different kinds of usability tests and user observations. Heuristic rules and simulations can be of help in the evaluation of existing systems. Later in the process, these methods can be used to evaluate the preliminary concepts and product definitions.

These and some other methods for communicating with users and getting in touch with usability problems are described under the topic of Usability Methods for Product Development.

Even though the salesmen's slogan says, "The customer is always right", the customer is not always right (Nielsen 1993). Customers, or users, may wish to include options and functionality to the product that have no importance to the actual tasks. They may feel that if they do not express all their wishes to the designers now, the “perhaps needed” functions will never end up into the system. Additionally, they may exclude some very important basic issues because of their implicit existence. At its best, gathering of task information may reveal the tacit needs and point out unneeded functionality. This provides at least means for weighing different needs to focus development work. Methods for task and function analysis like Contextual Inquiry (see e.g. Holzblatt & Beyer 1995 and Duncan & Beabes 1995) help this work.

Management support is important here, too. At first, managers are supposed to have a view of the strategic areas of the company. From this point of view, they are responsible for focusing the development work and must support the ongoing work. In a large organisation, their duty is to see that all parts of the organisation support each other optimally.

The internal issues of the developing organisation have their place in the evaluation of the possibilities that technology, production and existing experience have to offer. If these topics are discussed in the beginning of a development project, preliminary preparations or improvements can get started to meet the forthcoming needs. If there are major obstacles, they may have an effect to the desired development plan.

The outside world poses requirements for the developed products, too. Not only competitors but also officials via laws, regulations and standards affect the work done.

Sharp & al. (1999) present "baseline stakeholders" in their paper "Stakeholder Identification in the Requirements Engineering Process". The baseline stakeholder groups represent the different viewpoints that take part in a development project. On one hand the stakeholders act as information sources and on the other hand as developers of the information that is used to define the functionality of the product under development. As people may participate in several development projects, their role may differ (the role may be different even within a single project for instance in a small project where the project manager acts as a developer as well). People who act in different roles have different needs towards the information that they want to know and stay in touch with during the progress of the project.

In large projects, the vast amount of information may become difficult to manage. This finding stresses for stakeholder-specific (or role-specific) information filtering to avoid unnecessary information overflow. Role-specific information filtering may happen at both ends: either the creator of information directs the information towards the desired roles or the person acting in a specific role expresses interest towards specific information (or filters out unwanted information). One idea and aim may be to provide role-specific views to the development information.

Implications for this work. As product development is activity that incorporates several stakeholders, an information support system for product development should provide a structure for recognising the existence of different stakeholders and provide proper support for each of them. Each stakeholder has a "role" in the development project. Each role can be and is often assigned to be responsible in providing a specific point of view to the application area. These different point-of-views should complement the concept "role". In a general level, the point-of-views may be topics like business issues, customer requirements, market demands, competitors, technical possibilities and user needs. This issue provides a partial definition of the key concept "what is product development".

3.3 User-Centred Design: Processes and Methods

This section provides basics to the underlying key concepts about "what is user-oriented product development" and "what does systematic development mean" in the context of user-centred development work. The methods that are presented in this chapter represent a very basic level of user-centred development work and is utilised later in this thesis as background information about the *information structures* that user-centred development requires. This very basic level is selected because of the widespread use of the methods in practical situations. Basically, the aim of this work is not to improve the methods themselves but to study the means of managing the information provided by them.

There are a few process models that provide basics for user-centred development work. These models provide background about the systematic way to conduct user-centred development.

A rather generic and rough model has been presented by Booth (1989, p.118). His model divides product *usability testing* to three phases. A rather similar division of *usability design and evaluation* phases can be made to

- requirement specification phase
- development/implementation phase
- testing and validation phase.

The methods in the *requirement specification phase* gather information about the usage and problems of existing products. Some methods try to explore the users' intentions how to improve current tasks (e.g. observation, Contextual Inquiry) and some figure out possible task scenarios with completely new devices (visioning, scenario creation). Critical and other measurable parts of requirement specifications may be put into the usability goals, or usability requirements, of the product. These can be verified latest in the testing and validation phase but they will also direct the development work in development phase.

When the actual product is being developed in the *development/implementation phase*, usability gets its concrete form in the user interface. Developers use their own expertise to create the best possible solutions. In addition to their own knowledge and experience, developers may depend on different usability documents like standards, guidelines and style guides. Descriptions of practical usability problems made in the requirement specification phase serve for focused usability design. The usability goals set in the requirement specification phase can be used to verify the development results already during the design and implementation.

The last phase that takes place chewing the actual development process is *testing and validation*. By testing the products, usability can be verified and proved. Testing will be done with specific test tasks and against specific test criteria. If there is development work still to be done, practical results can be gained to direct the following development work. The results of a usability test, however, need to be documented and delivered to the developers who implement the improved features.

Systematic development work requires that process instructions and procedure descriptions are available in the development organisation and that those instructions followed in the development work. From the perspective of quality systems and requirements set by ISO 9000 we know what sufficient documentation all the development processes and activities is essential. Systematics for user-centred development are offered by two main sources: ISO 13407 and a SPICE-structured (ISO 15504) evaluation model for human-centred development: the Usability Maturity Model (Earthy 1999).

Implication for this work. The process models that have been considered to be the most important for this work are the one presented in ISO 13407 and the one proposed by the Usability Maturity Model (UMM; Earthy 1999). These processes provide contents for the processes that were described in general level in the overall description of product development. Both process models leave open the detailed practical user-centred design methods that can be used to fulfil the requirements of the models. Therefore, more detailed usability design methods are required to complement the generic process models. The usability design methods provide the detailed level information and structures that a user-centred information support system should contain.

3.3.1 Reference Process for User Centred Development

ISO 13407 is a standard that defines the human-centred design processes for interactive systems (see figure 9). In more general level, reference software development processes are defined in other standards and instructions like ISO 15504 (SPICE), CMM (Capability Maturity Model; Masters & Bothwell 1995), ISO 12207, IEEE 1220, IEEE 1074, J-STD-016 (MIL-STD-498) (Nyström 1997). ISO 13407 aims at covering the whole development life cycle of an interactive product from the standpoint of human-centred and usability issues. In the definition of usability this standard relies on the definition given in ISO 9241.

Reference process. ISO (1996, p.1) characterises a reference model as follows: "A reference model defines, at a high level, the fundamental objectives that are essential to good software engineering. The high-level objectives describe what is to be achieved, not how to achieve them." If we apply this definition to user-centred development processes, we may conclude, "a reference process for user-centred development defines, at a high level, the fundamental objectives that are essential to good human-centred development." The definition continues that "The high-level objectives describe what is to be achieved, not how to achieve them." This statement makes it evident that the models are not applicable straight away. Instead, the application of the model requires organisation specific adjustments.

Reference processes can be used for various purposes. Curtis & al. (1992) present, that goals and objectives for software process modelling are:

1. facilitate human understanding and communication ("represent process in form understandable by humans")
2. support process improvement ("identify all the necessary components of a high-yield software development process")
3. support process management ("develop a process-specific software process to accommodate the attributes of a particular project such as its product, or organisational environment")
4. automated guidance in performing process ("define an effective software development environment")
5. automated execution support ("automate portions of the process").

Process oriented standards for user-centred development. ISO 13407 mentions that ISO 6385 (Ergonomic principles in the design of work systems) constitutes provisions for it. ISO 13407 is based on the principles presented in ISO 6385. Additional standards that have an effect on user-centred development are listed on the EUSC web site at http://www.lboro.ac.uk/eusc/index_r_standards.html[19]. The list contains process and product oriented standards. In addition to ISO 6385, 13407 and 9241, process oriented standards mentioned at the web site are ISO 10075-1 (Ergonomic principles related to mental workload - General terms and definitions) and ISO/IEC CD 14598-1 (Information Technology - Evaluation of Software Products - General guide). The product-oriented standards cover mainly the development work of pure software systems (VDT, visual display terminals).

ISO 13407 describes user-centred design as multidisciplinary activity, which includes human factors, ergonomics knowledge, and techniques that aim to improve human working conditions and minimising the adverse effects of use on human health, safety and performance (Bevan 1997).

[19] Checked 19-Feb-2004.

According to ISO 13407, a development organisation should implement four user-centred design processes during each development project. The processes are (see figure 9):

1. Understand and Specify The Context Of Use
2. Specify User and Organisational Requirements
3. Produce Design Solutions and Prototypes
4. Carry Out User Based Assessment

Understand and Specify the Context Of Use. In the first human-centred process, the context of use is defined. According to ISO 9241 that is in the background of 13407, the context of use consists of user characteristics, user's tasks, user's tools, and user's environment. The definition of the context of use is done by finding out and documenting the users' characteristics, tasks and the social, organisational and physical environment. The relevant user characteristics may include topics like knowledge, skill, education, experience, training, physical characteristics, habits, and capabilities. As it is a rare condition that all users fall into the same user category, or segment, it is usually necessary that user characteristics are defined separately to different user groups. The categorisation can be done based on the characteristics that have greatest influence on the use of the product. It may be, for example, the user's technical experience.

The description of tasks should include the users' overall goals for using of the system, allocation of functions between the system and the human, and the operational steps between the human and the tool. The user's environment description should include the

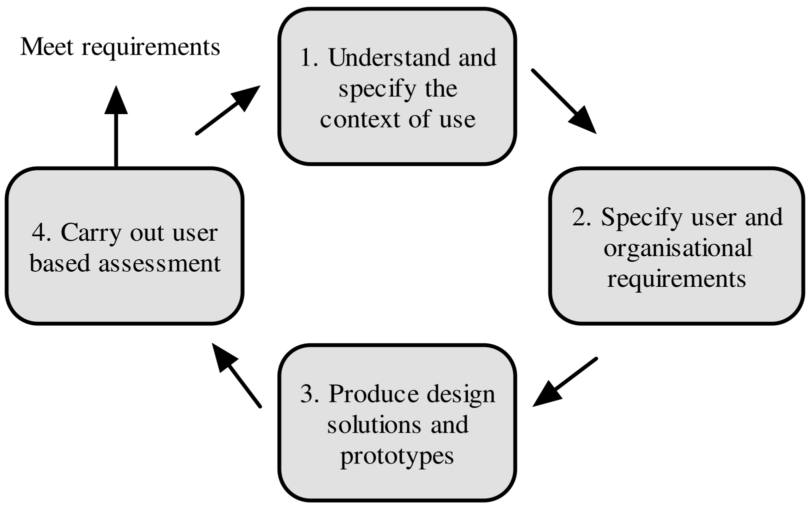

Meet requirements
1. Understand and specify the context of use
2. Specify user and organisational requirements
3. Produce design solutions and prototypes
4. Carry out user based assessment

Figure 9. The human-centred design process according to ISO 13407 (1997).

description of hardware, software and materials and also relevant characteristics of the physical and social environment need to be described. (Bevan 1997, ISO 13407 1997)

Specify User and Organisational Requirements. Before creating functional specifications for a system, we need to know what the system intends to support. These intentions that lay in the background of the functionality of the system remain often hidden. In order to direct the following design activities to correct direction throughout the development project, we need to make these background issues visible. The user and organisational requirements can include, for example,

- required performance of new system against operational and financial objectives
- co-operation and communication between users and a other relevant parties
- work design organisation
- the human-computer interface and workstation design

The specification of user and organisational requirements must be adequately documented, it must identify the range of relevant users and other personnel in the design, it must be confirmed by the users or those representing their interests in the process, and it must provide a clear statement of the human-centred design goals. (ISO 13407)

Produce Design Solutions and Prototypes. Any design work aims that construction of a product. It is, however, important that before starting the final production, the product has been appropriately tested. The testing can be done by constructing and evaluating prototypes in the middle of the design work. Based on the results from testing, iterative design loops should occur until the prototype fulfils the requirements. The design solutions should be done according to multidisciplinary approach so that all relevant standpoints are taken into account.

Carry out User Based Assessment. User based assessment is a key design method in human-centred development. User based assessment should appear in all appropriate stages of the development cycle but it is especially important at the end to verify that the design has reached the required level of usability. The assessment must be designed carefully to cover all aspects of design. These include the human-centred goals, responsibility of evaluation, how the assessment is to be conducted, how much resources can be put on assessment, when will the assessment take place. Based on the results from assessment, feedback needs to be provided back to the development and new design iteration may take place.

3.3.2 Usability Maturity: Assessing Conformance to ISO 13407

An attempt to look at the organisation's ability to conform to the human-centred development processes and activities it the Usability Maturity Model ("UMM", Earthy 1999). In his thesis work, Jokela (2001) presents experiences from five assessments that were conducted in four industrial companies with the KESSU UPA model that has been developed from the basis of the Usability Maturity Model.

The Usability Maturity Model is a model that can be used to "assess the degree of capability reached by an organisation in its ability to perform human-centred design activities". It does not directly point out the assessment against ISO 13407 but the processes that UMM presents contain the processes presented by ISO 13407. One main difference between the processes presented by ISO 13407 and UMM are in the beginning and at the end of the development cycle. In the beginning of the development cycle, UMM presents two processes that address the inclusion of user-centred issues in the strategic planning of the development organization (HCD.1). Similarly, UMM presents a process for planning the human-centred activities in the design cycle (HCD.2). At the end of the development cycle is also an additional process (HCD.7) that takes into account the implementation and introduction of the system in the customer organization. The processes in the middle (HCD.3-6) represent the core processes defined by ISO 13407.

UMM presents a model that can be used "to measure how well organisations do the human-centred part of system development and support projects. The model is also intended to provide a basis for those planning what human-centred activities to perform on a project and to assist those who wish to improve how well their organisations perform human-centred activities." (Earthy 1999) The informative part of UMM is based on ISO 13407 and the assessment model is based on ISO 15504 (SPICE). The model can either be used to assess the current capability of an organisation to produce human-centred applications or it can be used to create a framework of human-centred activities that need to be performed in different stages of the development process.

ISO 13407 and UMM provide a process model for the management of user-centred information.

3.3.3 Basic Methods for User-centred and Usability Analysis and Design

In order to define what kind of activities may take place in user-centred development and in user-oriented development organisation, related development methods are presented here. The description of these methods provides an in-depth viewpoint to the detailed-level components that are supposed to be part of the information support environment for user-centred development work. The details in the methods reveal *information structures* and expected *information content* that a user-oriented development organisation is supposed to manage with or without a supporting information system.

Even though ISO 13407 and the Usability Maturity Model provide a basic framework for user-centred development processes, they do not point out the practical methods and tools that can be used to implement and realize the “reference process” in real settings. The framework needs to be filled with appropriate methods and activities that operationalise the process model.

The following chapters present a short overview of such methods (see figure 10). The method descriptions also point out indirectly working procedures and expected stakeholders. These data is used in the definition of concept and in the construction of the information support system.

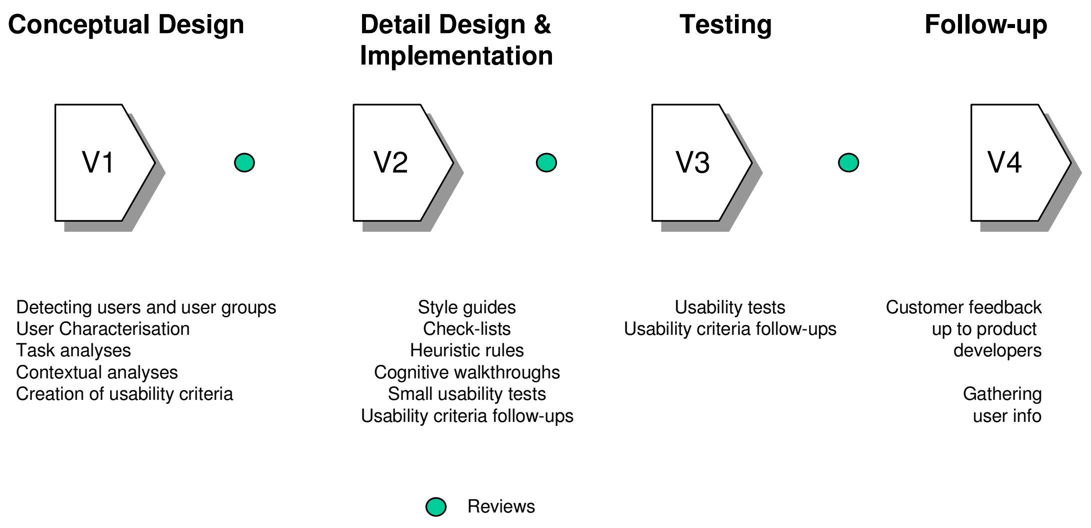


Figure 10. A selection of usability engineering methods aligned with typical phases of development process. Intermediate evaluations in review meetings provide "gates" or "exits" to next phase as e.g. suggested by the model presented by Valjus (1994).

The information that is gathered with the methods for user-centred and usability design needs to get documented and communicated with the "design rationale" mechanism that is applied in the organisation.

Usability and user-centred issues can be accessed with a variety of methods that can be applied in different phases of a development project. Benefits in applying explicit methods include that they make user-centred development work more systematic and predictable and that they have the possibility to make user-centred issues visible to all development stakeholders. By making usability and human-centred issues visible in the development process, there is the possibility to provide even measurable review criteria for it ("exits" or "gates" in the development process). This way the user-centred issues can managed in a similar way as other product attributes.

One way to group different development methods is to do it according to the expected use (and phase) of the method in the development process. In the following figure, there is a short overview of some usability methods attached to phases of a development process. Muller & al. (1997) give a similar but a more comprehensive list of methods.

In conceptual design, the focus is in the description of user groups, use context (environment), and characteristics of the users (Booth 1989). The aim is to provide developers with sufficient practical and easily applicable information about the user and context of use. Based on this description, usability goals (usability criteria) can be set to reflect the demands of the task and use context. This provides an evaluation base for all upcoming usability tests and reviews.

In detail design and implementation, the focus is in more detailed topics that are of great importance; details matter. When detailed product features are created, user descriptions provide a base for quick desk tests. The implementation can be quickly evaluated against heuristic rules (Nielsen 1993), or if more detailed evaluations are required, against usability guidelines or user interface checklists that may contain several hundreds of statements. Cognitive walkthrough (Polson & al. 1992) is a powerful tool for detailed interaction analyses, but its use requires basic knowledge about cognitive psychology.

Usability testing is perhaps the most often used method for usability design and evaluation. In usability tests, real users or good representatives of the user group use the product in real or real like situations performing typical tasks. The situation is usually videotaped and the outcome of the task is evaluated against usability criteria. This way usability can be measured and affected systematically during development.

After the product has been released and delivered to market i.e. users, its usage must be followed. To support the further development of the next versions of the product and the development of other subsequent products, information about practical use of the product needs to be gathered from various sources. The methods for doing this follow-up may include user observation, use of log-files, support and maintenance call analyses, and complaint and claim analyses. The information that is gathered here needs to be stored for later access.

Usability can be addressed in product development with different kinds of usability methods and tools. Obviously, different methods focus on different types of problems and can be used in different phases of a product developent process. Some usability methods and their corresponding places in a product development process are shown in figure 11.

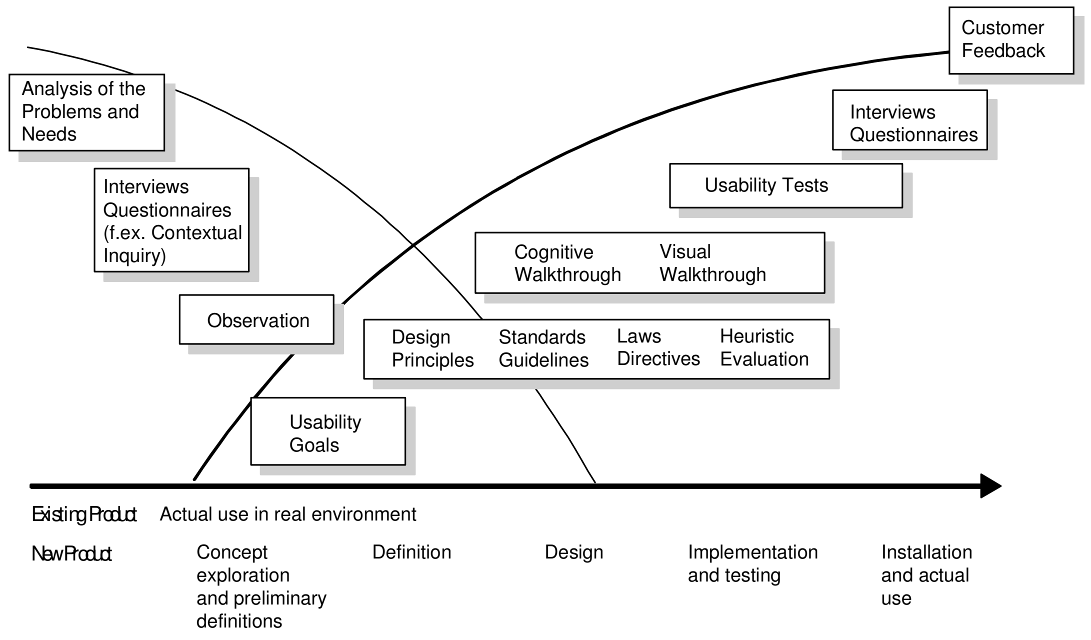

Customer Feedback
Analysis of the Problems and Needs
Interviews Questionnaires
Interviews Questionnaires (f.ex. Contextual Inquiry)
Usability Tests
Cognitive Walkthrough
Visual Walkthrough
Observation
Design Principles
Standards Guidelines
Laws Directives
Heuristic Evaluation
Usability Goals
Existing Product
Actual use in real environment
New Product
Concept exploration and preliminary definitions
Definition
Design
Implementation and testing
Installation and actual use

Figure 11. Some usability methods attached to different stages of product development (see Whitten 1990). The curves behind the method boxes represent starting and ending development projects (life-cycles).

3.3.3.1 End-User Detection, Division and Definition

The basis for all usability discussion is the end-user. All usability actions, the use of different methods and tools, approach to better understanding and support of the end-user while she is performing her tasks in the working environment. This is why the detection, division and definition of the end-user and user groups is essential. All usability results relate to the attributes of the end-user.

The first thing to do is to find out what are the target user groups of the product. There may be just a single user group but it is likely that people with different tasks, background knowledge and education will use the product. The detection of users and user groups can be made by surveying a customer database, doing research on the users of an existing similar system, or discussing with people from other affecting groups to product development.

The expected users of a system should be characterised. This can be done through a description of a stereotype individual that belongs to the defined group. The explicit description makes it easier and more concrete to evaluate different solution possibilities in latter phases of the development. Issues that can be described from the example user are: age, sex, profession, education, training to this task, task expertise, device expertise (previous, similar devices), technological profile, intensity of work, frequency of device usage.

The detection and division of end-users helps the development people to focus and restrict the aimed situations to which the device is developed.

3.3.3.2 Questionnaires and Interviews

Questionnaires and interviews can be used to evaluate the subjective satisfaction and attitude of a user towards a device or a system. Suggestions for improvements can be discussed through these, too.

Questionnaires can provide measurable results from a large group of users. If an overview of usability topics is wanted, these can be useful. If there are free form answers to questions, they can reveal additional problems that have not been addressed in other parts of the questionnaire. Examples of usability questionnaires are QUIS (Chin & al. 1988) and Ravden & Johnson's (1989) usability questionnaire.

Interviews and questionnaires can be used attached to a usability test or separately. Pre-test interviews and questionnaires can be used to find out the preliminary assumptions that users have towards the system. After the test these topics can be covered again and changes in the assumptions can be evaluated. Background information about test users can be collected with standard questionnaires.

Questions used in conjunction with a usability test can be (Whitten 1990):
- How satisfied were you with performing this task?
- What would you have made this task easier?
- How satisfied were you that the tutorial prepared you for performing this task?
- How satisfied were you with the terms used?
- List any terms you did not understand.
- What did you like most about the product?
- What did you like least about the product?
- What improvements overall would you suggest?

The answers can be free form or given with the scale from 1 (very satisfied) to 5 (very dissatisfied).

The answers to questions can be given using different scales. QUIS uses a scale of 1...9. In both ends there are opposite adjectives that describe users' attitudes. Ravden & Johnson (1989) uses scales *always, most of the time, sometimes, never* and *no problems, minor problems, major problems*. An often-used scale in conjunction with statements is the Likert scale (Nielsen 1993). In Likert scale there are values from 1 to 5. At the ends of the Likert scale are opinions *agree* and *disagree*.

The topics of an interview can be used as the measurement criteria for the usability goals presented in the usability matrix. A goal might be "90 % of users must give a rating higher than 2 to satisfaction" or "Number of assists per task must be lower than .33".

Interviews can be used in early phases of development. Doing interviews requires more work than doing questionnaires to same amount of people, but in interviews deeper knowledge about the topics of the discussion can be gained.

An example of a specialised interview technique is *Contextual Inquiry* (Duncan & Beabes 1995; Beyer & Holzblatt 1998). If the developed product is being used in an existing environment and the tasks are closely related to existing tasks, contextual inquiry can be used to find out detailed information about the social and technical environment of the product.

Contextual inquiry is done in a real working environment. The aim of the technique is to get realistic data about the characteristics of the users, their tasks, environment and terminology. The situation of the inquiry is supposed to be as normal as possible without any uncomfortable obstacles. The observer should let the user tell about his work and lead the conversation. Important topics that should be discussed are:

- user's tasks
- goals of the tasks
- work place
- tools
- user's terminology (words)
- communication with other people
- organisational structure and culture
- goals of the organisation.

3.3.3.3 Task Descriptions

Descriptions of the users' tasks construct the basis for the functionality of a device or a system. They are also used as background information in almost all usability design and evaluation situations. Thus, this importance of task descriptions shall not be underestimated.

Task descriptions are gathered from the real end users. Task descriptions tell how users of the device walk through their typical procedures. They contain detailed information about different stages of the process and describe the user's goals.

If the device or system under development provides a completely new way of action, task descriptions are one way describe these. In this case, we can talk about *task scenarios*. Task scenarios can be written out similarly to descriptions of existing work procedures. As with realised user interface prototypes, scenarios are also subject to evaluations as well. Methods for doing this include heuristic rules and guidelines. Due to the “activity” nature of scenarios, organisational and social evaluations may be required as well. Evaluation techniques for this have not been presented much yet, and there seems to be a “hole in research” for this.

Hierarchical task analysis is one method for the analysis of the tasks (Johnson 1992). In this method, the task is divided into tasks and subtasks all having corresponding goals and subgoals. To all tasks, there are certain procedures by which the goals can be achieved. The description of HTA is a hierarchical tree that describes all the steps required to achieve the goals of the task. Johnson (1992) has developed a method called Knowledge Analysis of Tasks that bases on the theory of “task knowledge structures”. Knowledge analysis of tasks is concerned with identifying a person's task knowledge in terms of actions, objects, procedures and goals (Johnson 1992, p. 170).

The data is collected from people that are directly or indirectly affected by the task. Several methods can be used to gather the data needed. These include protocol analysis, observations, questionnaires and interviews.

Tasks are seldom separated from influences of other people. At work, supervisors set up the goals for different tasks and interaction with colleagues may change the preferences of important issues. These all have an effect on the goals individual users' form. Thus, an organisational point of view can be included in task descriptions.

As tasks are typically carried out with the aid of technological devices, it is important to describe these tools. In the description of the task, phases that are and that are not dependent on existing technology should be identified.

3.3.3.4 Usability goals

Usability can be defined as a measurable factor in the development process. In order to be considered a "real design criteria" this may even be mandatory in the context of technical development. Measurability helps to treat usability as a serious topic and practical exit criteria for the review meetings during the development process.

Using the generic usability factors as a source one can create meaningful usability topics. However, they are not usually applicable as themselves but need to be translated into more practical and application related topics. This can be done during the requirements analysis phase by looking at the tasks and goals of the real end users through the generic usability factors. Thus, a basic task can be looked at from different perspectives: is the operation effective enough and is it easy enough to learn? The goals may even look like opposite to each other in the beginning, but this requires decisions that put more weight to some aspects or approach to other type of solution.

Usability aspect	Measurement tool	Measurement unit	Worst level	Planned level	Best level	Notes
Subjective attitude	How many users dislike the system?	Amount ot users	2 / 5	1 / 5	0 / 5	
Few errors	How many errors happen in the log-on procedure?	Amount of errors	20	10	0	
Ease of learning	How much training does a user need in order to do the basic functions without help?	Time spent at training	3 days at a course	1 hour with manual	15 minutes without manual	
Ease of use / effectiveness	How fast will an experienced user load the basic parameters?	Time elapsed	20 min	5 min	2 min	
Easy to remember	How much training does one need after 6 months disuse of the system?	Amount of training	Half a day	1 hour	15 min	

Figure 12. Usability criteria matrix (see Whiteside & al. 1988). This type of a matrix may be used to set explicit and measurable criteria about usability related issues to the product under development.

One way to deal with usability issues in a measurable way is to put usability criteria into a "Usability Goal Matrix" (Whiteside & al. 1988, see figure 12). The rows of the matrix contain information about usability topics relevant to the system. The columns contain information about the measured topic, its measurement unit, worst level values, planned level values and best level values that can be reached at least some day in the future. This goal matrix needs to go through a wide review discussion so that correct topics are measured and realistic goals are set up. All development groups also need to commit to the goals and accept them as review meeting exit criteria throughout the process.

3.3.3.5 Heuristic Evaluation, Guidelines and Standards

One type of design support method (or tool) are the various types of guidelines and “design instructions”. Heuristic evaluation is an example of this type of method.

Heuristic evaluation means the evaluation of a product against general principles and "rules of thumb". The evaluation can be done basically by anyone but knowledge and experience about usability and usability related problems makes heuristic evaluation more effective. Heuristic evaluation is an effective method specially when evaluating the very first prototypes of a product or products that have not yet been designed with usability in mind. In these cases, heuristic evaluation can reveal fundamental and robust problems even without actual user based testing. According to Nielsen, heuristic evaluation is a good way to start applying usability approach in design and development.

Nielsen (1993) has presented a 10-topic list of heuristic usability rules (sort of rules of thumb). They are focused on software evaluation but may apply to other industries, too. Nielsen’s original heuristic rules are presented in table 1 (they have been refined since then). These kinds of checklists and rules need to be accessible through the quality system of the development organisation.

Heuristic rules are general by nature. More detailed instructions can be found from different *guidelines* and *standards*. Examples of guidelines are Smith and Mosier's (1986) guideline that has over 900 topics and guidelines for different computer user interface environments like Motif, Microsoft Windows and Macintosh.

Table 1. Heuristic rules presented by Nielsen (1993).

1. **Simple and Natural Dialogue**. The structure of the "conversation" between the device and the user should be as simple as possible. This means that within the dialogue all the desired information must be presented but nothing more.
2. **Speak the user's language**. In addition to simple dialogue structure, the terminology must also be suitable to the context and fit to the user's background knowledge. The user is normally considered to be interested in the results of the interaction, not the technical details of the device.
3. **Minimise User Memory Load**. If the device shows some results in the middle of the dialogue, the user should no be forced to remember these things later. Instead, the device should do all the "remembering" for the user.
4. **Consistency**. Consistency means that similar kind of functionality must work similarly in different parts of the system. That way the requirements for additional and unnecessary learning can be reduced.
5. **Feedback**. A user wants to know, if his operations have had effect on the device. Thus, the device must provide proper feedback to the user. Feedback can be textual, visual or auditive but it has to be clearly understandable. Special consideration to feedbak is required in situations in which the system has long response times.
6. **Clearly marked exits**. If users enter a part of a system incorrectly, they want to go back easily. To support this, the escape procedures must be clearly expressed.
7. **Shortcuts**. As users learn to use a device, they may want to speed up the operation with it. This can be supported by providing shortcuts to commonly used functions.
8. **Good error messages.** User will do errors when using a device. In many cases, devices can have defects, too. These error situations must be handled with clear and understandable error messages. User must be able to know, how to proceed or make a correction.
9. **Prevent errors**. Even though it has to be accepted that errors can happen, they should be avoided in the first place. In many cases, there are many "trappable errors" that happen because of user input. User input can be restricted to a selection of correct answers and the performing of critical incidents can be verified with additional questions as "Do you really want to do this?".
10. **Help and documentation**. Many contemporary devices and systems must be used with additional support information. This can be provided as on-line help or proper manual documentation. Help should be really help, not just a list of features of the system. As well as to the product itself, usability issues cover these components of the product, too. Despite the importance of help and documentation, manuals are rarely used in real environments, so the product itself should provide enough support for using it.

3.3.3.6 Usability test

A usability test is a formal situation in which a test user is performing real or real-like tasks. Usability tests can be used to reveal *usability defects* of a product. Usability defects can be considered as analogous to reliability and performance problems to products (Booth 1989, p. 118). A usability test can be arranged in a real working environment or in a special usability laboratory.

A usability test consists of real users who test a device as if they were in a real use situation. Thus, descriptions of typical tasks are required as a prerequisite in a usability test. The tasks must be described to the user in such a style as if they were given or formed in a real task situation. References to detailed system specific functions should be avoided.

During the usability test, the user is asked to think aloud. This will hopefully reveal the thoughts of the user and give information about the user's terminology and mental structures. As the user interacts with the device situation specific detailed information comes up.

A usability test can be conducted according to instructions given by Gomoll & Nicol (1990). Before running the tests, objectives for the test must be created, tasks to test must be selected and described, settings for the test situation need to be prepared and representative users must be found. The test situation itself can be conducted through the next ten steps presented in table 2.

Table 2. The usability testing procedure (Gomoll & Nicol (1990).

1. Introduce yourself
2. Describe the purpose of the observation (in general terms)
3. Tell the participant that it's okay to quit at any time.
4. Talk about the equipment used to observe the user.
5. Explain how to "think aloud."
6. Explain that you cannot provide help.
7. Describe the tasks and introduce the product.
8. Ask if there are any questions before you start; then begin the observation.
9. Conclude the observation.
10. Use the results.

One method that can be used to complement a usability test is Visual Walkthrough (Nieminen & Koivunen 1995). With visual walkthrough some cognitive overload problems that emerge in a usability test can be avoided. The problem of having to do too much new things and think-aloud at the same time tend to lead into silence during the think-aloud sessions. Unfortunately, thinking aloud is the main method for finding out what the user thinks about the interface and the functionality of the product.

In visual walkthrough, test users are not allowed to do anything with the product or the system. They must just tell what they think about the user interface elements and the functionality they attach to these elements. During a visual walkthrough session the test users of a product are asked to do the following:

1. The user is asked to look at the system and tell what he or she is looking at in general (an overview)
2. After the overview has been done, the user is asked to tell what kind of elements and units he or she recognises in the user interface
3. When recognising the elements or after it, the user is being asked to tell how he or she understands the symbols and terminology used in the user interface

4. After the recognition of the user interface elements, the user is asked to describe what they expect that will be found behind the functional elements (e.g. buttons, switches, menu items)

The results of visual walkthrough can be used to evaluate the visibility and understandability of the user interface elements.

3.3.4 Usability Embedded in Product Development Organisation

Knowledge about possible user-centred development methods is not sufficient when thinking of implementation of user-centred activities in a development organisation. Various organisational characteristics, restrictions and hindrances, existing working methods, the composition of different stakeholders – all these affect the real implementation of user-centred activities.

Usability can be looked at in three levels in the context of product development. A *single designer* affects the detailed features of the product using the framework and infrastructure of the project. *Projects* in the development company are focused to a specific application area and they are given instructions about quality factors and measurement principles by the management that directs and focuses the *whole development work*. The three organisational levels that need to be taken into account are designer level, project level, and organisational level.

These three levels are reflected in the various process maturity models that have been applied in development organisations recently. In the field of human-centred development, the Usability Maturity Model (UMM, Earthy 1999) utilises the levels and structure of the SPICE model. At maturity level 1, individual developers carry out the processes (the Personal Software Processes, PSP, indicate this level as well, see e.g. Humphrey 1996). At maturity level 3, there is organisational support for the enactment of the processes. At level 2, the processes are “managed” but not throughout the organisation i.e. individual project groups are responsible of the conformance to the processes. By integrating the user-centred activities to other already existing efforts in these different levels in a development organisation we may reduce the hindrances in the introduction of the human-centred development processes.

3.3.4.1 Designer level

Individual designer is the one that utilises the results that usability evaluations and user-centred design activities offer. He is also the main user of the usability design methods. Thus, the knowledge and expertise of a product designer in usability is essential for good results in usability. When usability and user-centredness is introduced to the organisation, training, education and other supporting usability material must be reachable by the designer.

To start with, heuristic rules are a good and easy starting point for the designer. Nielsen's (1993) list provides a way to get in touch with the "usability thinking". In the heuristic rules, several background theories are encapsulated to simple statements. These statements can be used to evaluate detailed design solutions at the time of their creation. Other sources for everyday design support are application area specific guidelines and style guides that exist for different technical and operational surroundings (platforms and application areas).

“Embedded usability testing” is one way to have outside opinions to the design solutions. If real end users are not available, it is at least good to get “another opinion”: other designers can give their opinion about the solutions. This can be supported by quality systems by implying peer reviews, for instance. Design reviews are already part of modern quality assurance procedures and these can be expanded to cover usability issues as well.

In addition to detailed information about usability methods and tools, designers should be supported by action descriptions that offer them suitable methods and tools for use in different phases of product development. These “process descriptions” are project level documents that describe the entire flow of the development work for one product.

Due to the multi-disciplinary nature of the human-centred design approach, the utilisation of the human-centred, or user-oriented, design artefacts requires often interaction between the participating stakeholders. The designer’s ability to participate in e.g. usability reviews affects the outcome of design. It may be useful that the supporting methods, tools, and practices in the future take into account the discussion-like style of development. This may indicate, for example, the provision of intermediate designs for the evaluation of the whole team thus reducing the “it’s now finalized” approach in the early stages of the development process.

3.3.4.2 Project level

As the contents of individual usability methods and tools are of great importance in the designer level, the project level descriptions should focus on the use of correct methods and tools in correct phases of the development process. At this level, the structure of an individual project must be defined with all milestones and quality reviews. At this level, usability goals are set for each phase and the achievement of those is followed.

A concrete way to provide the designers with proper information is the creation and use of a usability guide. This guide describes shortly the overall structure of a product development project that designers can follow. It contains also short and practical descriptions of usability design and evaluation methods that can be used in everyday design situations. Using a usability guide, it is easier to ascertain that these issues will be taken into account through the organisation.

3.3.4.3 Organisational level

At the organisational level the spectrum of projects, focus areas, result assessment figures and quality guidelines are handled. If usability is wanted to be a key issue in the individual development projects, metrics for its evaluation need to be created. It can be attached to quality requirements, exit criteria for the development phases, user feedback or support level. As the construction of individual quality manuals is defined in the organisational level, the inclusion of usability here will spread the approach rapidly through all projects.

3.3.4.4 Introduction of Usability Approach in a Product Development Organisation

Even though methods for usability exist and there are appropriate places for them in the development process, it is not easy to introduce them in a real development organisation. Before an organisation or its individuals will start using the new approach and the corresponding new methods and procedures, there must be a need for change. Suggested new approaches must be accepted. This requires internal or external tensions towards current practices. This issue affects the practical process and method implementation work. However, this issue will not be dealt with in detail in this study. But for completeness purposes, it is addressed shortly.

The need for change tends to be market driven (changing circumstances; see e.g. Ward & al. 2001) or intra-organisational. Market driven needs may rise from customer demands or from competitive situation. As an example of these drivers, O'Hara (2000) reports that "*Meeting real customer needs and improving project estimates and visibility with the customer were the key drivers for the software process improvements*" in a company called NewWorld Commerce. From the usability standpoint, customers stress for easier-to-use products. Similarly, competitors may release new products with improved design and interface that needs to be responded to. Occasionally, a single company wants to be the first in the industry to release usable products and gain market share by that.

From the usability standpoint, intra-organisational needs may rise from frustration to huge amount user support that is considered trivial. Another reason may be the continuous rejection of delivery tests. These events may point out usability related problems and the improvement work may start from rather small actions. If a company has "usability enthusiasts" in the organisation, one way to introduce usability is to let them try the easiest and lightest methods in a restricted pilot case. If the method turns out to be non-profitable, it will not be used in the next project; otherwise its use may spread. This will set up requirements for new kinds of result measurements, too. For example end-user reactions and support costs should be included within the evaluated figures of the development organisation.

Nielsen (1993, p. 20) has presented a "recipe for action" to introduce usability in a development organisation. He suggests a comprehensive five-step action procedure but notes that a lighter one-step start is also possible. This lighter version includes a short usability evaluation to a product (or a user interface).

In this lightweight action procedure, a usability test with just a few real end-users is conducted. The end users are supposed to perform typical tasks of theirs without *any* help or interference from the designers. If this is the first time ever a usability test has been conducted to the product, it is more than likely that some problems appear. These results should be used in the development of the next version of the product.

As with all actively used issues, knowledge about usability will enhance during time. This knowledge is, however, tacit for the development organisation unless designers document it somehow. Possibilities for this exist in the context of quality systems and design rationale. A quality system may require that in certain phases of the development work, usability reviews have to be done. In these reviews, the arguments for different design solutions can be discussed and documented. This way not only the individuals but also the whole organisation can develop in the area of usability.

Some experience about the introduction of usability to real development organisations was gained during the *Usability programme* (1994-1997) that was coordinated by the Federation of Finnish Electrical and Electronics Industry[20]. At the beginning of the project, most of the participating companies wished to get external help and support for their usability efforts from the participating research organisations. During the project, most of them started to put internal effort on it, too. External support only does not have enough influence on most organisations; company internal interest and dedication to this issue is needed, too.

In the Usability programme, external support varied from training to conduction of usability tests. Especially small companies had restricted resources for dedicated usability design. In large organisations the reasons for slow ramp-up on new things rely on the budgeting cycle. Getting more resources to take care of new topics can take years.

Some obstacles for the introduction of new issues, work styles, and tools are presented in the following list. The list is put together mainly based on the discussions with industrial representatives during the Usability programme.

- **Intensity of work**. People have no additional time for anything beyond their existing tasks. This means that usability cannot be expected to put effort to without additional or redirected resources. This sets up serious requirements to the form of usability documentation offered to development staff: it must not be long and it needs to be practical, fitted to meet the needs of the particular organisations and individuals.
- **Resistance to change**. Related to previous topic, if something new is added to the tasks of people that are overcrowded with their jobs already, they may not be willing to easily accept anything that seems to add their work. More than that, changes in general may be felt uncomfortable at least in cases where the use new methods may reveal negative things about the work that has been done earlier by an individual developer. Especially usability evaluation reveals issues that are not always pleasant for the developers. This does not make it easier to implement usability activities.

[20] http://www.electroind.fi

- **Results in the long run**. Usability considerations can decrease the amount of work through the increased focus to important topics. Even though the verification for this may partially take place already during the development project, the real results will be available only after the product launch and the delivery to end customers. These results will be available only in the long run, which makes it more difficult to accept the new approach rapidly.
- **Lack of Management Support**. Management should stand behind the new approach and ideas and provide support for individuals willing to utilise the new tools or approaches. Managers need to be in the first people that should understand the benefits of the new ideas. Without management support the work vanishes easily.

3.3.4.5 How Much Consideration Does Usability Need?

Since its industrial expansion in the mid-1990's in Finland, usability is an issue that has been thought to be in order in many companies (Toivonen, Vuori & Pekkarinen 1995). This is based on the assumptions that if the products include all the required functionality, it will be usable as well. In cases where explicit effort to usability design and evaluation has been put, this may even be true, but then again, it is almost completely a question of individual designer's usability skills. Reasons for ignoring usability in development can be found from the current practices within each industry and lack of proper information and knowledge, too.

In most cases, usability methods provide means for getting validated results abut the usability of a product without having to rely on implicit assumptions on usability. Of course, if no usability defects can be found during usability evaluations, quitting using them may be considered. Nielsen (1993) presents that this is hardly likely to happen, if no explicit usability approach has been introduced earlier within the development of the product. Later, as designers learn the basic principles of the users' behaviour in the selected environment, the nature of usability inspection may change towards "implicit integration of usability"[21].

But the question of the designer's usability skills raises a question about the need for using the usability methods. In a sense, usability is something a designer puts into the

[21] It has been discussed that there are two different schools that weigh differently the importance of processes and developers' skills: The "process school" aims at securing the quality of design with pre-defined "standard operating procedures". The "skill school" emphasizes the importance of the individual designers' skills as means for reaching high-quality output.

product as he is doing his daily job implementing parts of the device or system. If a designer is experienced in usability, he is probably able to think about usability factors already since the very first prototypes. Thus, he is also able to detect usability defects in the later phases of the development. In this kind of a situation, the nature of usability inspections may change from tests to assurances. Still, a verification of usability goal achievement needs to be done as a part of the quality assurance process.

3.3.5 Implications for this Research

As a summary of the previous review, following implications[22] can be presented to constitute partially to the architecture for the user-oriented development environment and organisation.

Process. In a multi-stakeholder environment, product development work is often arranged in a process-like way. Alike technical development, there are reference process models for user-centred development as well. The user-oriented development organisation should be able to conduct development work by applying these process models either separately or attached (or embedded) to the overall development processes in the organisation. The development environment should enable a foundation for process modelling and appropriate environment-specific tailoring, when required.

Methods. Being often rather general by nature, process models do not normally point out specific methods that can be utilised in different development environments. In practice, design, analysis, and testing methods are required to complement the process model in order for it to be practically realizable. There are no single generic methods that could be used in whatsoever case and the development environment should be able to provide guidance in selecting appropriate development methods and support for the utilisation of the method. These include storing, sharing, and analysing the information gathered with the method.

The methods, obviously, vary based on several reasons like the phase in the development process and the application area of the product under development. From

[22] As the aim of this part is to provide basic construction elements for the architecture of user-oriented development and its organisation, the following statements are presented in a normative form and manner. It is obvious that this approach has lots of underlying, both evident and hidden normative assumptions about desirable and “good” ways of doing design that may be argued. But for serving the modelling purposes in the later parts of this thesis, this approach is selected.

the standpoint of user-centred development, key focus is in describing various user-related issues including goal and task descriptions, use scenarios, user's education, training, and experience as well as product related issues like usability goals and appropriate user interface design guidelines. As each method typically contributes to a restricted subset of design, for instance in terms of process stage and product or application area, it can well be expected that additional viewpoints will emerge in the future. This issue points out the demand for flexibility for both in the selection of methodology as well as with the structuring of information managed with the methods.

Organisation and stakeholders. An increasingly important factor affecting the result of development in user-oriented development work is the constellation of stakeholders that take part in the development work. There are indications that this issue is getting greater importance all the time (e.g. participatory design during the last two decades, and more recent attempts like the UMM/Attitude model). It does, therefore, provide one element for the architecture of the user-oriented development work. User-orientation can be looked at with different scopes in the development organisation: in the individual designer level, in a project group level and in the organisational level.

Development approach / Values. User-orientation can be seen to be a design approach arising from the values and ideals within the development organisation. In order to get more widespread acceptance in the development organisation, these values and ideals require advocates, internal spokesmen, who point out the positive impact and results that the selected approach may provide. An often made remark is that management support for all important issues is required. There are no reasons to believe why this would not apply to user-oriented development approach as well.

Tools. The organisation of the user-oriented development environment is not, however, only about the people and human resources. The practitioners apply different tools in order to achieve the desired results of the work. In design work, tools that support the multi-sided analysis of the information around the development work, are of central importance. It can be seen that the quality of the design concept (and implementation as well) is dependent on this. Tools with organisational dimension may provide a platform for information gathering and processing (i.e. discussions, analyses, decisions) in a multi-stakeholder organisational environment.

3.4 Empirical Background: Organisation of Product Development Activities

In order to be able to construct a conceptual model, architecture, about user-oriented development organisation, and an information support mechanism that can support systematic user-centred development work, information about current real working practices in which the support mechanism will be applied is required.

In the following chapters, this is done with three empirical studies. In the first study, an overview of working practices in several product development organisations is presented. The study makes use of an interview that consisted of a rather broad range of issues that have been addressed earlier in this work. The second study addresses the topics in a more detailed way. It is a case description of one development project of an embedded product that consisted of physical, electric and electronic parts, and software that provided the functional logic for the user interface.

The third study presents an in-depth description from an organisation that has utilised and applied the principles of Contextual Design in their development activities. In this thesis, this part is supposed to provide a type of state-of-the-practice description of real-life user-oriented development organisation and its activities. The third study is also used as the requirements specification work for the prototype implementation of the information support environment.

3.4.1 Empirical Study I: Overview of Current Practices in Product Development[23]

The goal of this research section is to cover two issues. Initially, the aim is to find out what is the current state-of-the-art of the "usability approach" in product development. Usability, being a concept that is not clearly defined widely in development organisations (at the time of making this part of the study, at least), issues that relate to usability may appear in different forms in different organisations. Therefore, it is also important to address product development, its structure and content, at a more general level. By using this approach it is possible to figure out what kind of possibilities, help and support usability can offer for different stages and different participants of product

[23] The results presented in this part of the thesis have been presented in a conference paper by Nieminen & Parkkinen (1998) in ODAM98. The text here contains, however, some complementary results.

development. The combination of these issues aim at finding practical means and methods by which usability issues could be examined already in the early stages of product development process.

Important questions in this part of the study were how usability had been taken into account in the development practices in the past and how it was intended to be taken into account in the future. One way to incorporate new topics into existing practices is to include it as a small additional side issue among other important topics, not changing the whole system at once. We wanted to find out whether usability had been addressed with some other concepts in the existing development work and therefore we could attach the issue to the existing practices with minimal overhead.

3.4.1.1 Subjects and the Interview Method

In order to find out how product development is currently done, a total of 17 Finnish product developers working in five different commercial product development companies were interviewed. The interviewees were development engineers (N=6), project managers (N=6), and persons having broader quality responsibility in the organisation ("quality managers", N=5). The companies were working in the field of software development (N=2), mainly hardware (N=1) and embedded electronics (N=2).

The interview (see appendix D) consisted of about 70 questions. There were slight differences in the emphasis of the questions depending on the interviewed person's position in the organisation. Quality managers were asked questions that covered broader topics and development engineers were asked about details. The structure of the interview, however, was similar in all cases.

Even though the questions were defined in a detailed level, the actual interviews were conducted in the interview meetings like normal discussions. In some meetings, the interviewees held the list of questions and went through the list according to their own pace.

The interview addressed following topics:

- Current task and project of the interviewee, characteristics of the project, structure of the project and other participants
- Working methods, working instructions, quality system, metrics for the work
- Staff organisation, communication and reporting responsibilities

- Working conditions, education and training
- Development tools
- Customer participation, product specification and documentation, process documentation, prototyping, usability

Three different versions of the list of questions were applied in the interview study. All interviewees were presented questions that addressed same topics. According to their background and position in the development organisation, emphasis was put on slightly different topics. The main difference in the questions and answers was in the organisational coverage: the questions that where to be answered by the quality manager concerned the overall activities in the development organisation. The questions that were assigned to a project manager concerned issues about project level. And the questions for other members of the development team covered issues concerning their own design and implementation work within one project. The questions that were targeted to product developers in a development project are in appendix D.

Related research methodology. The topics that we addressed in this study have been addressed earlier in other studies. An early study in the HCI field called "Designing for designers: An analysis of Design practice in the Real World" addressed the design of interactive systems (Rosson & al. 1987). The results of the study (N=22 designers) were presented in the CHI'87 conference. This study was addressing issues about iteration in design, user testing, design of the user interface, and generating ideas in design. Issues were addressed from the standpoint of a recent development project that included a significant user interface component.

Other results about studies addressing similar topics have been published by Lubars, Potts & Richter (1992) and Curtis, Krasner & Iscoe (1988). Both studies have been addressing the working practices in software development projects. Especially near the question setting and method of this study is the study conducted by Lubars & al. (see table 3). The emphasis in that study is the creation and management of requirement specifications but it addresses several issues that are present in our study as well. The questions used by Lubars & al. are in table 3. From the standpoint of user-centred design practice, more recent studies about its status are presented by e.g. Vredenburg & al. (2002).

The emphasis on requirement specification as a framework can be argumented with the fact that major decisions concerning the product are done at the stage. Information about users and usability should be available at that time if we expect that it will be visible in the resulting products as a designed feature. The decisions that will be made in the early stages of the development project will affect usability. By conducting user-centred design actions in the early stages of the development process, we can affect the resulting features in the product in the right time without major redesign requirements that might be the case if we put emphasis mainly on testing and evaluation.

Table 3. Question list used by Lubars & al. (1992).

(Lubars & al 1992:)
General Can you describe your work? What projects are you working on currently?
Requirements origination How do system requirements come to you? In what form? From whom? Who interprets for you, if anyone? What decisions are you permitted to make about requirements? Who is the customer?
Requirements elicitation and validation Who in the customer organisation do you interact with? How do you check that you understand the requirements? How do you document requirements? What types of questions do you ask? Do you perform walkthroughs?
Requirements modeling Do you / how do you specify system behavior? Responsibilities? States / modes? Effects of actions? Triggering conditions? Timing? Statistical performance requirements? Do you / how do you specify system data? Objects / data? Relationships / attributes?
Evolution Do the requirements change? If so, how often and radically? How formal is the change process? Do implementation constraints impinge on requirements? Customer-specified platform? Pre-existing system (if the proposed system is a modification)? Other systems (integration / interoperability)? Desired components (3rd party reuse)? Your knowledge? How is the system delivered? (Big bang, increments, etc.) Do you prototype? How? How do you assess feasibility?
Methodology Any experience of methods (esp. OO)? Results? Any tool use? Results? What connection is there between requirements analysis and project planning?

3.4.1.2 Results of the Interview

Project / product type. All companies developed products that can be characterised as mass products i.e. the products will be sold to currently unknown user population (unknown in detailed level). One of the companies was in the field of software development and it developed also customer specific products in which the customer and end-users population is more restricted.

Usability. Usability was to some extent a familiar concept to 13 people (N=17). Typical answers defined it as "that the end user can use the product easily"[24]. According to 7 answers usability had already been included somehow in the process documentation.

User involvement. According to 10 answers (N=17), contacts with users and development staff has been minor (nothing or "contacts at the fairs"). According to 5 answers, some contacts have taken place during development time (installation visits, pilot use, and usability tests). Mainly, users are expected to be contacted indirectly via sales or marketing. In some cases, developers are regarded as users as they use the products that they have developed. Customers and users are sometimes separated from each other but there appeared to be a preference for "*customer oriented*" products.

Communication. Product developers communicate daily with their colleagues and project managers in their own or neighbour projects. Representatives of support groups are also contacted frequently in review meetings and as the project is about to end, communication intensity with these groups increases. Communication with colleagues happens informally, managers are met mainly in weekly meetings and reviews.

Quality system. The existence of a quality system in the own organisation was known by all quality managers (N=5/5). One of the interviewed developers (N=1/6) was able to describe the elements of quality system without supporting material. All project managers (N=6/6) knew about the quality system but not all of them were able to describe its structure. Quality systems were referred to as process documentation, ISO 9000, checklist, and quality handbook. Quality documentation consisted of information about compulsory events during a project, reporting responsibilities, documentation requirements, and guidelines and checklists for development work.

Specification and documentation. Product development documentation contains following documents: requirement specification, product specification, and functional specification. Competitor analyses are done to provide background information and complement these documents. Requirement specification is expected to be provided by

[24] Some answers to the question “How does usability appear in the development documentation?” were as follows: “easy to use, ergonomics”, “the device is useful to the matter that one needs it for”, “functionality, functioning of the product”, “ergonomics, terminology, visibility check, localisation”, “easy to use by a normal consumer, the product fits the specific uses that it is meant for”, “easy to use, self-clarifying, intuitive”, “usability check, consistency”, “usability test, pilot test”, “how to use the device”, “customer satisfaction”, “the product fulfils the needs of all stakeholders in a cost-efficient way”, “user can reach the desired outcome with minimum effort and displeasure”.

marketing staff. Project and product managers create product specification. Functional specification is created mainly by the project team. Process documentation that is created during the project consists mainly of development staff meeting memos and minutes.

Review practices. Reviews were common in all organisations. Only one interviewee mentioned that reviews have not been formally conducted during current development project. In addition to this, three (N=3/17) interviewees mentioned that practices for reviews were not precise. Even in these cases, however, it was mentioned that there were informal checkpoints or milestones during the project. Review meetings are gathered round various topics like schedules, problems, production issues, product specifications (functionality), development time documents, technology, testing and conformance to specifications, visual design, costs, subcontracting, and sales and marketing issues. Usability and localisation reviews were mentioned, too.

Subcontracting. Internal subcontracting is realised through "neighbour projects" and these projects are attached together via management-defined schedules and deadlines. External subcontracting has been used mainly in activities that are outside the "core competence" of the organisation. These have included translation services, industrial design, and usability evaluation. The use of company internal support services is quite similar to external subcontracting from the standpoint of the development project. External subcontracting was, however, experienced difficult from the standpoint of quality and schedules.

Measurement of development work. The measurement of development work was not established in practice in any of the companies. Measuring development work was considered rather difficult and in some cases even not desirable. Some developers even had the impression that rewards from "measured" good work were not tied to the results of the work but to the manager's activity and bothering. There have been some attempts to measure, or evaluate the conformance to deadlines ("time") and budget ("costs"). "Time to market" was mentioned in one interview. Reports from product testing ("errors") have been used, too. A suggestion was posed that one could evaluate the conformance of the product against requirements. It was also mentioned that a correct metric for development success would be customer feedback.

Schedule and deadlines. 15 interviewees mentioned that project schedules do not normally conform to the planned deadlines. Two persons answered, however, that the project has fulfilled schedule requirements. In these two cases, the schedule was created together with all stakeholders of the project. Problems were seen to arise from vague milestones and long periods with no reviews. Some schedules have been known to be unrealistic already since the beginning. In some projects, however, no strict and explicit deadlines were present at all.

Development tools. The most common development tools were different kinds of computer systems that are obviously industry specific. Companies developing mechanical products used CAD software and software development companies used various programming tools. For documentation, standard office software was used. In one hardware development company, even paper and pencil got one mention as important tools.

3.4.1.3 Conclusions based on Current Practices

According to the findings in the interviews, users are not directly involved in the development process. The presence of the requirements that they have, come from various sources that is not always clear. In the case of customer-specific applications, the user population is obviously clearer. In both cases, however, means for gathering, storing, and communicating user information and user requirements are needed.

User-centred information is, however, not in a coherent and structured form. As user-centred issues are new to the practitioners and the organisation, they do not have a view of systematic procedures to address user characteristics, users' tasks and use context information. Process guidance and simple documentation templates about user-centred issues are required. The requirement for process guidance is evident based on the existence of quality systems and regular development reviews. If possibilities for document templates are provided in the information support mechanism, there is also the possibility for the management of the various specification and verifications documents that are in use in product development.

Subcontracting is a way to address issues that do not fall into the core development areas. A user-centred information support system should provide a communication platform even for extra-organisational teams that can manage parts of development information together without the need to be organisationally related or physically near to each other. In practice, this implies for wide area networking (WAN) solutions of which Internet technology is perhaps the most feasible.

The measurement of development work was considered difficult in the answers. As information is the basic construction element in product development, one possibility is to measure the amount of information that has been gathered during a development project. An information support system may also keep track of information usage during the project. This way it may become possible to provide "content specific" metrics for a development project.

Computer based tools are the most important ones to product developers. Therefore, a computer application can be regarded as a feasible solution to be the information support mechanism.

3.4.2 Empirical Study II: Case “MyPhone”

In order to understand the internal dynamics of product development in a bit more detailed level, a retrospective case description is provided here. From the standpoint of this study (the creation of a knowledge sharing mechanism for user-centred information within product development organisation), deeper understanding about design events, communication structures and all various stakeholders is required. This case description puts the elements that were detected with the general interview in the previous section into real context.

The research method in this case study was a retrospective interview together with document reviews. The interviewees were a project manager and a developer in a company that designed and produced embedded devices consisting of mechanical, electrical, and logical i.e. software elements.

3.4.2.1 Starting point for MyPhone

MyPhone[25] 1200 is a new generation phone that is developed by MyTel, Inc. *MyPhone* has been developed in a project that results in two new product models: “1200” and “1500”. In addition to this, all applicable results will be used in the following and already ongoing concurrent development projects.

[25] The name of the products and the company has been changed.

The "MyPhone" project started from the basis of another project called "Wide" that had already been running for a couple of years. In a way, the "MyPhone" project continued the work that had already started earlier even though the resulting product was to be an independent product. The reasons for the development of the new product were similar as in that previous project: basically, the reasons were market related but some internal drive for new products existed, too. The existing product models were not regarded as bad but they were thought to become ordinary and somewhat outdated during their rather long lifetime. Especially a new, fresh, appearance was wanted. Competitors had started to have 3 or 4 different models in the market. At that stage there was only one model available from MyTel. At the same time, market needs for enhanced functionality and components (like displays) were detected.

The initial aims were to create a product family of 2 to 3 devices that were ergonomic by their user interface. Usability and user interface was, thus, considered important in the development.

3.4.2.2 Organisation of "MyPhone" Product Development

Events of the project. The project started at fall 1994 with the name "MyPhone". At that stage, the trade names of the resulting products were not known yet. At the end of the project, the names of the products were decided to be "1200" and "1500" to fit to the names of the existing product lines. The marketing department did this naming. The development of "1200" and "1500" were separated to own projects in the very late stages of the project. The development of "1200" was to be carried out first, and the development of "1500" would wait until the user interface and functionality of the "1200" would have been freezed.

Even though the project started officially at fall 1994, preliminary analysis about competitors' products had already been done before. That information served as background information for the development group. As a prerequisite for the project, two physical models of the possible product designs existed, too. The models were made of wood and they were made at summer 1994.

In the beginning, the project was evaluated to be one year in length. The resulting product was supposed to be released in fall 1995.

The main parts of the “MyPhone” project were: 1) user interface development, 2) mechanical design and 3) production arrangements. User interface design was the first task for the development group. Mechanical design was supposed to start right after the basic structure for the user interface (form of the device, basic operational logic) had been freezed.

The last part of the project consisted of arranging the production; getting plastic moulds and product covers as well as organising the assembly of the product. However, at the same time some detailed component design was still on the run.

User interface design. Here, user interface design can be characterised as the design of “functional logic that is visible to the user through the embedded display”. Even though the design of the physical user interface, i.e. t the cover of the device, buttons etc., is part of the user interface, it is partly considered under “mechanical design”. This includes tasks like doing the CAD construction drawings.

It was stated that the effect of user interface design is of major influence to mechanical design. Thus, the major decisions about the user interface were supposed to be freezed early in the project. The design of the mechanics was supposed to start right after the freezed user interface definitions.

In the beginning, there was a time of 2,5 months for the design of the principles and basic structures of new user interface. Already in the first review meeting, a mock-up model of the user interface was introduced. The user interface designer who presented the initial models was responsible for the design of the functional logic that appears to the user through the embedded display of the device.

For the design of the user interface principles, a user interface workgroup was established at this stage. The design principles were then defined within two months. Already at the very first reviews in the beginning of the project, two computer simulations of the user interface were created and one older one was reviewed. After freezing the basic structures of the user interface, additional four months were available for the refinement of the user interface.

During the project, usability issues strengthened their position in the design. At the end of the project, an in-house usability evaluation as well as a usability test were realised. The in-house evaluation was done by several employees of the company with the aid of an evaluation form. The evaluation form consisted of instructions to go through various functions of the system. The user was then asked to evaluate whether it was difficult (1) or easy (5). The usability test was conducted at an outside organisation and consisted of the evaluation of typical tasks that a user is supposed to do with the device.

Mechanical design started right after the basic definitions about the user interface (covering mainly physical components) had been freezed. This happened 3 months after the project had been initiated.

3.4.2.3 Participants of "MyPhone" Product Development Project.

Following staff was involved in the "MyPhone" development project:

- Steering group (technology manager, project manager)
- User interface design workgroup
- Industrial designers (form and shape of the physical device, audio design, screen icons: all outsourced)
- User interface designer (functional logic, in-house staff)
- Software engineer (outsourced)
- Cognitive scientist (in the design of the logic of the internal phone book, outsourced)
- Production subcontractor
- Marketing

Steering group. As the steering group consisted of the managers of the company, Its primary mission is to evaluate the market possibilities in the beginning of the development project as well as to follow how the project proceeds. The issues handled in the steering group should cover "broad lines" instead of technical details. It appeared that it is easy to get deep down into details even in the meetings of the steering group.

User interface design workgroup. The mission of this temporary user interface design workgroup was to create the initial definitions for the basic user interface. The participants of this group were the chairman (project manager), three in-house product designers including the user interface designer and an external consultant. The lifetime of the workgroup was two months.

User interface designer. One person inside MyTel was assigned to the design of the user interface of the new product. In this case, user interface design was primarily concerned about the logical functionality that appears to the user through the display of the device.

Industrial designers. In the "MyPhone" project there had been three contracts with three external consultants in the area of industrial design. These consultants (A, B, and C) were responsible of following issues

- design of the form and shape of the physical device (A)
- physical model making (A)
- computer aided design of the covers of the device (A)
- participation to functional design of the user interface (A)
- audio feedback design (B)
- design of display icons (C)

Software engineer. Software engineering was also outsourced. Even though the new products were of new "look and feel", a great amount of the electronic components and software were passed from the previous products. As an external software engineer was already familiar with the development of this, he was hired to continue with his work.

Marketing. The role of marketing was twofold. Their impact existed implicitly in a way throughout the development process. The user interface designer had earlier been working at the marketing department. In addition to this, people from marketing participated in some product reviews. In the latter stages of the project, marketing people got prototype products in order to review and comment them. At the early stages of the project, a marketing workgroup was also established to gather market information for the designers' use.

Production subcontractor. Discussions with the production subcontractor (plastic covers) were started after the external design was done.

Cognitive scientist. As a phone book was included into the phone itself, a cognitive scientist shortly consulted about the operating logic. The commenting of the design was carried out within two months (calendar time) and the contract was closed with a review meeting.

3.4.2.4 Information Base for Product Development

The basis for new product development was mentioned to be twofold: market demands and internal drive. In addition to the initial ideas about the need of new products, several other activities, surveys and documents provide the "information base" for the product development. Initially, market analyses ("customer needs") and development of technology ("technological possibilities") provide the starting impulse for new development work. At design time, main information is delivered at the review meetings and through design documentation.

3.4.2.4.1 Initial Information

The project "New Phone" started from both internal and external demands for new products as mentioned earlier. However, preliminary studies supported the decision about starting a new product development project.

Competitor analysis

The preliminary analysis made about competitors' products consisted mainly of technical functions and visible surface elements. Topics covered in the analysis included issues like

- type of screen (no screen, amount of characters and icons)
- the amount of buttons in the devices (hardcoded and soft-keys)
- overview of functionality (amount of functions, function name)

The devices were divided into three categories: basic, standard and executive. The "Basic" model included only the minimum amount of elements and functionality. The "Executive" model consisted of a large display and a great amount of buttons and functions. The "Standard" model remained in the middle of these two.

A more detailed report about all major competitors and market areas was also done. This consisted of topics like external design, connections to other devices and components, (production) technological solutions and the display.

The user interface workgroup presented a wish about an analysis of functions in the competitors' products to be conducted by marketing organisation. It was initiated as a check for crucial functionality to be embedded in the designed product. This happened a while after the main functionality and user interface had been defined and just before the coding for the user interface was about to start.

Marketing feedback

The marketing organisation had gathered customer information about a previous product called 1000SX. That information covered topics concerning the physical components and the overall design of the product. A general note about the memorability of the additional functionality of the product was also made.

In the beginning of the development project, the marketing workgroup was assigned to gather market information for the product definition. In February 1995, feedback was gathered from resellers at a common internal meeting. That information covered physical characteristics of the device.

3.4.2.4.2 Information at Development Time

The development work at "New Phone Project" was done concurrently in several organisations. In a way, the project can be characterised as "subcontractor-intensive". The project manager had to put lots of effort to organise and follow the work done. The main communication style in this environment was keeping reviews and other kind of meetings.

Meeting Memos

From the meetings, a sequence of memos is created. In the case of "MyPhone", memos were created from each part of the development project, i.e. development of the user interface, mechanics and arrangements of the production. The memos contained issues like

- components, connectors, chips and their locations, cost effect of different solutions (mechanics)
- display details, dialogue structures, text expressions, prioritisation of functionality; what is most important and what is easiest to implement (user interface)
- production organisation

In addition to the above topics, all memos contained administrative information about the progress of the project, forthcoming events and responsibilities for different people.

Technical Documentation

During the development of the mechanics, a checklist with key issues was created. It contained 44 questions or statements that needed to be answered and checked.

During the "MyPhone" project, several design documents were created. For documentation purposes, a list of all documents was kept up-to-date. During the design of the user interface, more than 50 documents were listed. This list included following information: topic of the document, document version, document numbering, date, author, file name, file format and attachment information.

The documents covered preliminary thoughts about the functionality and guidelines for the user interface design as well as more detailed definitions of the different functions. "Preliminary thoughts" served as a ground for discussion in the first meetings. Guidelines and function definitions were used to transfer ideas, wishes and requirements to in-house development staff as well as to sub-contractors that created the actual features.

3.4.2.5 Development Tools and Techniques

Product development is done with a variety of methods and development tools. In addition to a typical product development engineer working style, individual planning and implementation work, following methods, tools and techniques were utilised in the "MyPhone" project:

Group work with wallpapers. In the early stages of the project, product ideation was realised through group meetings. The results of the meetings were documented with post-it notes put to wallpapers. The wallpapers were stored in the room of one developer.

Reviews. The reviews were normal meeting that addressed just some specific pre-defined topics. There had been form and shape reviews and user interface reviews.

Computer simulations and emulations. Three simulations about the user interface had been used in the beginning of the project. One simulation was five years old and two new ones had been created. One technical emulator had also been created that was to be attached to the designed products.

Mock-ups. Initial user interface ideas were presented in overhead slides to the steering group in the beginning of the project. Right after this, a paper-based mock-up about the display in real size was created. This way it was possible to evaluate the size of the characters and icons that appear on the display.

CAD drawings. CAD drawings were created about mechanics during the project. The pictures were wire-pictures and more fully visualisations.

Focus Group Survey. The preferences of the potential customers were planned to be evaluated in a focus group survey. The initial project plan included this, but this was not realised.

Usability inspections. In the late stages of the project, the evaluation of usability was thought to be an important part of the product design and testing of the initial prototypes. Usability was then inspected through internal usability evaluations (in-house staff) and with outsourced usability tests.

3.4.2.6 Restrictions for Product Development

Product development has not been free of restrictions in the case of the “MyPhone” development project. Major restrictions for the development were reported to appear from

- previous products (may be divided into current, existing technology and suitability of new products to existing devices)
- emerging technologies and
- component and production costs.

As the development work at “MyPhone” project started from the idea of renewing the existing products, the technology used in the development work implicitly relied heavily on these. The initial ideas for the renewal were twofold. In the first stage, the visual appearance was to be changed. The operating user interface was supposed to be derived and glued together from two existing products “1000” and “LightCom”. In the second stage of the development project, the functionality and internal technology was supposed to be changed. Here, major effort was put to getting the device to operate in a new kind of technological environment.

In the selection of the components, price was regarded as a major selection criterion. In the case of the display to appear in the device, a four-row alphanumeric display was selected instead of a more flexible graphics display. Reasons for the selection of the display were not only at the price but also in the existence of display control electronics. A readily available controller existed for the alphanumeric display, as the controller for the graphic display should have been created within the project. That would have meant additional costs to the project as well as additional risks to the time schedule. Even though graphics did not truly appear in the display, it was still possible to add a few graphical icons to the tailored display.

3.4.2.7 Conclusions from Case "MyPhone"

Product development is conducted both by company internal stakeholders and some external consultants. Different stakeholders are assigned different responsibilities concerning specific development topics. Topics may deal with issues like user interface development, mechanical design or production arrangements. The stakeholders meet in meetings that deal with topics that each participant has on ones responsibility.

The results of these meetings are recorded in meeting memos that are stored in the project folder. The project folder is managed by the product or project manager. Information in these memos is somewhat scattered and should someone wish to get an overview of a specific issue, he needs to leaf through all the memos and create a summary of the topic. This requires a lot of effort. An information support mechanism should be able to provide summaries and overviews by just gathering together all the small pieces of information concerning a specific development issue.

Group work methods with wall techniques are rather well known among developers. They provide good means for collaboration in a shared physical workspace. The results of these sessions are described in big sheets of paper that are hung on the wall. After the design sessions, these wallpapers are stored in a room of one individual developer. This storing mechanism may lead to a situation in which no one really uses this information afterwards. And, perhaps even more important, the information is not available in any other location that the physical office where it is located. The wallpapers could be transformed into digital form and they could be delivered via the network to all requesters.

As some restrictions for the “MyPhone” development work came from earlier products, it would have been helpful if the newly assigned developers that had not participated in the original project could have had access to development information from the previously conducted development project. Even though the new products were somewhat considered as independent from other products, there was the technological path from the previous products to the new ones. But the new product was targeted almost to the same user group as the existing product. Careful documentation about user characteristics, users’ tasks, tools and environment would have helped the initiation of the project and could have provided means for prioritising the possible new features that where to be included in the new product. In the current situation this work would have had to be done from the scratch.

3.4.3 Empirical Study III: User-Centred Development Activities in Contextual Design

The third empirical source of information that contributes both to the design of the architecture of user-oriented organisation and to the development of the supporting information management tool is an in-depth description of the utilisation of a contemporary user-centred development methodology called *Contextual Design* (Beyer & Holzblatt 1998).

This part of research presents activities that may take place in a user-oriented organisation, practical arrangement of the activities, resulting artefacts that are created by the user-oriented activities, and participating stakeholders. Even though this empirical part of this thesis contributes somewhat to the overall architecture of the user-oriented organisation, these data is utilised mainly as requirement specification for the implementation of the supporting prototype application, Knowledge Storage, which is described in the last part of this thesis.

This part of the description of this empirical case is focused on the description of the Contextual Design related activities that point out one possible way to do user-oriented development work[26]. The activity itself that is described here had its focus (and result expectations) in the design of the information sharing concept within the multi-stakeholder software development organisation that is organised in a distributed / decentralised way.

Contextual Design was selected in the case company to be the design methodology for the new development practices that should improve the management of user centred information. In addition to designing the new practices with Contextual Design, a secondary aim was to get first-hand experience in applying the methodology and see how the artefacts that are created during a Contextual Design session could be shared effectively in electronic form.

Contextual Design is a combination of different methods that aim at providing a solid base for the specifications of (software) product from the standpoint of end-users. The methodology relies heavily on the observation of users' work in the real workplace. This information will then be refined during the special Contextual Design process.

Contextual Inquiry. Contextual Design starts with Contextual Inquiry. This is an interview method that emphasises the importance of real events in the user's workplace. The interview is conducted at the user's workplace and the situation proceeds as if a "master and apprentice" were discussing. There is a pre-defined focus in all contextual inquiries that obviously restricts the observations to a limited subset of actions in the user's environment. The interviewer asks questions from the interviewee according to this pre-defined focus and makes notes of it. Occasionally, the person being interviewed is asked to show what he or she really does instead of just telling it. This way the interviewer can ascertain that things really happen according to the story. The aim is to overcome the problem that people tend to tell things like they should happen, not the way they happen in reality. And, as user-centred development is especially interested in these "breakdowns", it is important to get a realistic picture of the situation.

[26] This statement reflects the underlying assumption that Contextual Design (in a rather thoroughly completed form) can be regarded as a methodology that puts user-orientation into practice in development work. This issue was discussed much by the researchers that conducted an experimental UMM/HCD process assessment in a closely related research project. It must be noted here that the participants of the UMM/HCD assessment group were not unanimous of this kind of an interpretation. Therefore, this issue seems to require additional consideration and academic discussion that is, however, considered to be outside the scope of this study.

For the design of information support environment for user-oriented development organisation, seven persons (U1-U7) were interviewed in December 1998. Two interviewers conducted the interviews within one week. Because of practical resource limits, the amount of interviewers was limited to two. As the amount of interviewed persons should not be decreased (no less than six), the interview period got a bit lengthier than normally in a CD process. The amount of interviewers is normally in the Contextual Inquiry more than what was available in this case. Usually, all participating developers conduct these interviews. In a typical Contextual Design situation, the interviews are conducted by several interviewers (the participating developers) within one day.

The interviewees represented product marketing, development and support. All these are important stakeholder groups: they all are in contact with users or require information from users. In addition to questions about contacts with users, the information about users that they deal with was outlined. The interviewees were asked to introduce and demonstrate their information management tools and systems and provide examples about their activities and working styles with the tools that contained the information. The answers were recorded by taking notes of their answers and actions.

Sharing. Right after the interview (within 48 hours; so that the events are still in memory of the interviewer), the results are shared with all CD participants. Usually, there are 5-10 participants in a sharing session. Sometimes the amount of persons in the sharing sessions may even be higher than that. In these sessions, all interviewers tell others what happened in the interview meeting and what they discovered. Each sharing session must not take longer than the actual interview. Normally this is 1-2 hours. In sharing sessions, the results are recorded by writing *affinity notes,* creating *flow* and *sequence models* and by drawing *physical layout models*. These models provide basic data for subsequent CD stages. In sharing sessions, the participants have different roles like the storyteller (interviewer), affinity recorder (written notes from the interview, the story), modeller (flow model: stakeholders, information, documents, artefacts).

As the CD process was conducted with a limited amount of people in the case of the development of Knowledge Storage, a sharing session was conducted right after each interview. The interviews (contextual inquiries) were not conducted parallel but in a serial way. In order for the interviewer to be able to recall the events and findings from the inquiry session, this quick enough sharing of the results of the inquiry were arranged. In the sharing sessions the two interviewers and a third person were present. The roles in the sharing session were the storyteller, the affinity recorder and the modeller.

Consolidation. As the amount of interviewed people in Contextual Design is around six, the data needs to be consolidated into one covering model. After all the interviews are analysed together, the resulting individual models are combined into a single and unite model. The affinity diagrams are used to create chunks of issues that reflect important topics from the standpoint of the end users. The flow models with corresponding information flows and artefacts in the users' environment are combined together by harmonising the concepts and expressions that have emerged in the different interviews meaning the same thing. All consolidated models are then used for reviewing them with a larger group of people.

The creation of the affinity models and the consolidation of the flow models from each interview sharing session was done within one months period after the sharing sessions (January 1999). The results of the consolidated flow model and the affinity diagram were presented in the beginning of February 1999 to the persons who had participated in the interviews. This was done to provide feedback to all interviewees about their contribution and also for validating the results. The interviewed persons found the consolidated results and the affinity diagram to reflect the results of the interviews.

Visioning. After consolidation starts the visioning of improved ways to work. At this stage it is important to leave out all restrictions and figure out "high fly visions" about all kinds of better ways to organise the work in the target environment. The result of a visioning session is often some kind of a flow model. It is usually necessary to create 2-3 visions in order to get new ideas into the design but also to get finally back to realistic level.

In the development of Knowledge Storage, we created three versions of the vision: one "high fly", one realistic and one in between these. The last one was selected for further consideration. Visioning was conducted by having all previously created material available. The visions were created at the end of February 1999.

Creation of the initial User Environment Model. The concrete vision that results from visioning will be developed further. The model that results from this design work is the *User Environment model*. The user environment model creates a view of the system/product that will result from the development project. The user environment model represents the system from the standpoint of the user and user interface structure. It defines the dialog structures of the system and enables thus the creation of the prototypes that can be used for the evaluation of the concept in a very early phase of the development process.

The initial user environment model for initial prototyping with users was created in the beginning of March 1999.

Prototyping. Prototyping is required mainly for getting early feedback from the users. By applying paper prototypes (sequences of user interface layout) this can be done with reasonable effort and in a limited amount of time. The prototypes are created on the basis of the initial user environment model.

The paper prototypes of Knowledge Storage were created on the basis of the initial user environment model. The rather simple hand-drawn screen layouts were presented to four development representatives together with the storyboards that showed the expected typical tasks that users would do. The users had the possibility to comment both the paper prototypes and the storyboards.

Finalising the User Environment Model. Based on the results gained in the testing of the paper prototypes with users, the user environment model will be refined. The refined user environment mode is the outcome of the process and it provides basics for the writing of the user interface specification and implementation.

During the contextual design process, we created the artefacts that the methodology addresses: affinity diagram (important issues in observed work), flow models (key roles and information flows between them), visions of new way of working, storyboards (user scenarios) and a user environment model (structure and functionality of the system).

For testing and demonstration purposes, all these artefacts were stored by applying the Knowledge Storage itself. This material also acted as the initial introduction and training material in the Knowledge Storage to the participants of the pilot project in which we applied the concept and the prototype.

The contextual design work was supported with existing material from the current state analysis that had been conducted earlier. This material included an analysis about instructions in quality system, internal research reports about existing user centred activities, and an analysis about existing development information systems. Additional interviews of 12 development representatives were used as background information as well.

Implications from the utilisation of the Contextual Design methodology point out the need for product developers to manage different types of intermediate design artefacts, structured or free-form, that provide means for articulating the requirements in the developer community and for pointing out the evolving product features that have a solid argumentation from the real-life findings. A key issue is that the participants can create a shared view on the important characteristics on the users, their surroundings and the expected features of the product. In a decentralised development organisation it is not always possible to gather all stakeholders together but there is still the need for creating the shared mutually agreed view in order to proceed with the development to the desired direction.

In addition to this general level view on the development work, designers come up with design ideas and insights about the design during the various design sessions and even outside them. These issues contain important input for later development work (e.g. processing a functional priority list, implementation of the product) and therefore are important to record.

One implication that this case data points out is also the requirement for providing process guidance, or template procedures. Due to different reasons, product development needs to be performed in a traceable manner. Reasons for this may be, for instance, regulatory or based on competence (new issues require instructional support). Contextual Design itself is an indication that even in the field of usability engineering, explicit development procedures and design models & artefacts are emerging and require constraints and boundaries for their effective utilisation. This issue needs to be reflected in the organisation and arrangement of the design environment as well.

3.4.4 Implications for this Research from the Empirical Studies

The empirical studies complement the literature-based implications and provide practical considerations about the demands for user-orientation and the management of user-related information in real development organisations. Implications for this research can be seen to be as follows.

Product development work is getting more systematic. Demands about shorter development cycles and more customer-oriented product features are common[27]. Development work is not only freely fluctuating creative work that is conducted free of boundaries. Systematic does not, though, mean "uncreative" as product development is highly demanding creative work in the end; instead there is a demand for systematic means for supporting creativity. On the other hand, the level of systematic working (use of process models and procedures, quality system and corresponding documentation) is not always at a high maturity level[28] that is explicitly defined and measured. Knowledge about the existence and the contents of the quality systems that hold information about the systematic procedures is somewhat vague in practical daily work. According to the interviews, the quality and process guidance/control systems and activities are not yet working as an integrated support mechanism for the practitioners even though the practitioners do recognize the existence of such systems.

Based on the findings from the studies it can be argued that the quality systems and process/procedure instructions are not yet considered to be of great *support* for the daily development practices. Quality systems are, however, for different reasons considered to be important by the practitioners, and their importance is from that standpoint certainly growing. They can be considered to be emerging and improving ones that do exist and remain in the future development organisations. This finding suggests that if we can find means to utilise these currently important-but-not-so-practical elements in a more integrated, supportive, and practical way, there may be possibilities for quicker introduction and more integral inclusion of new and emerging important development

[27] An example of these demands in Finland is e.g. the RAPID technology programme ("Tuotekehityksen tehostaminen") at the end of 1990's that had its focus on speeding up the lengthy development processes and cycles in product development companies. Examples of required shortenings were "from 3 years to nine months". This overall demand, of course, was split up to different sections/parts that contribute to the shortening of the process. User-centred development was one of the issues that were expected to provide more focused possibilities to provide "right on target" style requirements for the products under development. Being able to focus to the development of the customer-desired functions only, the development cycle can be seen to shorten. Additional utility can be seen to be gained from the "better fit to customers' demands".

[28] Maturity levels for software engineering are presented in more detail in models like SPICE and CMM. The Human-Centred Development category is a model of human-centred (or, user-centred) development activities that is compatible with the SPICE maturity model.

standpoints. This issue is, however, heavily related to organisational issues that cannot be neglected in the analysis.

When the first studies that are presented in this thesis were conducted in mid-1990's, company wide intranets had not yet been introduced. From this standpoint, development work may be considered to have been rather local by nature even though some shared development (with clearly identified interfaces and working procedures) between international development groups was running. In the last case study conducted about three years later, the company internal (and internationally operating) information portal was running. Despite the fact that the utility of the portal was considered to require improvement, the portal was shared between developers from different locations thus providing an initial "shared information space" for them as the portal included also some interactive sections.

In addition to the emergence of the technical standpoint to the networking in the globally functioning product development, the emergence of shared activities between the stakeholders in the different regions can be seen to have improved. This affects also user-centred development activities: In the last development case, the observed end-users (with Contextual Inquiry) were in another country. The gathering of data may well have been conducted by local representatives. In such a case, there is a need to do the "sharing of data" that the Contextual Design methodology requires with other distant developers.

This "distributed (or decentralised) development" issue points out the emergence and improvement of the virtual side of development organisation and its activities. The most recent developments in the field of decentralised/distributed product development (see e.g. Tiako & al. 2001: "Process support for distributed team-based software development workshop") also point out that this activity is improving and also crossing the boundaries of the companies. Support for the distributed activity can be expected to be required from the technical direction as well as reported in Tiako & al. (2001; "Web-based Framework").

One of the most important implications from the second study is the need to provide support for inter-stakeholder communication. There is a growing amount of interdisciplinary communication within the development organisation – or development community whenever it is a question of loosely-connected developers. Additionally, based on the studies, user interface design is an emerging part of development work. In the second study, there was an external “user interface design consultant” taking part in the development work. The industrial design of the product and its user interface was also conducted by an external organisation. Based on the hindrances reported by the interviewed persons, these kinds of development arrangements require additional means for communicating the designs and corresponding decisions.

3.5 Some Implications Based on Background Research

So far in this thesis, we have been putting together the “construction elements” for user-oriented product development both from the theoretic-conceptual direction as well as from the empirical and practical direction. From now on, these elements need to be linked together and arranged in a meaningful way. This summary aims at starting the bridging of these topics that continues in the following chapters.

The initial (and normative) assumption behind the whole work is that doing user-centred development work is important. By focusing on users, we can improve the working or living conditions of the users and customers. Similarly, by focusing to the users, the resulting products will be of greater utility to them (and customers) as they understand the functioning of the product easier and are capable of achieving their goals with it more effectively and efficiently.

A continuum in this line of (positive) analysis is that as the customers experience benefit in using the product, there are better possibilities for the product to succeed also commercially[29]. The development of commercially successful products requires, evidently, other considerations as well that user-oriented approach needs to be aligned with. Therefore, there is a need to look at the surroundings of the development work with a somewhat larger perspective[30].

[29] It is noteworthy to emphasize the importance of the word “possibilities”. So far, there has been little scientific research on the linking of economic and commercial issues (not to mention the commercial success) and user-centred design. Links are rather easy to point via analyses out but solid scientific proof for this seems to be lacking. Thus, this is one possible research area in the field of user-centred design.
[30] From the standpoint of a focused research question, this statement is difficult and to some extent even “dangerous” as it broadens expands the scope of the work that is often a difficult thing in scientific work. Despite this, the approach of “looking around” in the development environment from user-centred perspective has been selected.

Systematic working requires that there is appropriate instrumentation available for doing the required important things during development. Product development is getting systemised with the introduction of specific methods for analyses and ideation. The HCI research field has also an offering of different methods and process guidelines to be used by the product developers. Based on empirical findings there is, however, a danger to over-systemise the development work so that developers themselves start experiencing that all the different important issues become a burden for them – not a support for the creation of more qualified products.

Another challenge in the current development environment is the multi-disciplinary nature of the work. This extends from thematic issues to practical distributed and even networked work arrangements. All these pose major challenges for fluent communication and weighing of different standpoints. This is the main issue that will be dealt with in the forthcoming chapters of this thesis.

4 Architecture of User-Oriented Development Organisation[31]

How does user-orientation show up in a product development organisation? What kind of pre-requisite is required for the inclusion of user-orientation in development organisation? We can identify different types of methods that unveil important issues about users and usability for the developers, but how can we ascertain that these methods can really be used in the development organisation? The studies presented in chapter 3 pointed out that there is a multitude of applicable methods but that the surroundings of the development work restrict (or sometimes supports) the possibilities for the application of these methods.

One challenging problem is also the multitude of different design approaches that emerge now and then in the trends around product development[32]. The user-centred approach can be seen to belong to these as well. All in all, the introduction of the methods (and tools) themselves does not seem to be sufficient. It seems to be so that the organisation needs to be able to utilise them in their particular technical as well as "mental" situation.

[31] The core of this chapter has been published in Nieminen (1999; SAICSIT'99). Changes and supplements have been included in this text, though.

[32] One interviewee in the initial state-of-the-practice interviews pointed out that "We [the practitioners in product development] are nowadays somewhat tired to all these *tiny-so-tiny-development-pilots* focusing to all kinds of issues. There are QFD, EMC, DFA and so on… And, yes, we have also participated in the 'usability' programme."

One way to look at the enablers of user-orientation in product development is following: to put user-orientation into practice, both interest towards this issue is required and capability to perform the activities with appropriate skills: this can be reflected with concepts attitude and ability[33]. In this chapter, a surrounding framework into which the user-oriented methods and respective supporting tools can be embedded into is outlined. This is done in order to position the collaborative information support environment in the overall development environment. The elements of the resulting model are to be used to construct or improve mechanisms that support the inclusion of user-related development activities in product development.

An early study from the standpoint of interactive systems and HCI practices in a real development organisation was reported in CHI'87 by Rosson & al. (1987) in a paper with the title "Designing for Designers: An Analysis of Design Practice in the Real World". In their paper, Rosson & al. divide the research on interactive systems to two perspectives: "the process of design, and tools to support the process". More recent results about similar topic have been reported by Vredenburg & al. (2002) in a CHI'2002 paper called "A Survey of User-Centered Design Practice". An important difference between these papers is the addressed target group of the study: Rosson & al. deal with software developers in general whereas Vredenburg & al. have already identified the group of "user-centred design (UCD) practitioners". Rosenbaum & al. (2000) go even further, they discuss about "usability organisations". Based on this, one can see that during the 15 years from 1987 to 2002, user-centred development is beginning to become an integrated part of the development organisations.

This integrated status raises also challenges: Vredenburg & al. (2002) refer to Gould & al. (1991) in the statement "user-centred design process was still not often used in practice due to both organisational and technical reasons even though the approach had been in existence for over a decade". Gould & al. have pointed out that organisational and technical reasons hinder the utilisation of the user-centred – or user-oriented – approach in development work. Vredenburg & al. mention that "It is of critical importance to the UCD community to determine whether the situation has changed over the past few years".

[33] In the development of the maturity models for Human-Centred Development, there has been an attempt to divide the topic into two pieces. There is a measurement model for developer attitudes (Earthy 1998: UMM / Human-Centredness Scale, INUSE Deliverable D5.1.4s) and for performing certain processes and base practices (Earthy 1999). Both these affect the ability of the organisation to perform user-oriented development.

The organisational aspect of user-centred development work has gotten more emphasis in the HCI field during the last half of 1990's. Rosenbaum & al. have arranged a series of workshops under the topic "strategic usability" (CHI'96, CHI'98, CHI'99 and UPA'99). They define strategic usability as "*embedding usability engineering in the organizational processes, culture, and product roadmaps. In strategic usability, usability data contributes to corporate-wide decision-making, such as product priorities and make vs. buy decisions*." This approach can be seen to indicate the transition from the tool approach towards more holistic and integrated approach between the user-centred design practices and other parts of product development work – even taking into account the other business units and business aspects as well.

The definition of strategic usability points out the requirement about the tight and close co-existence, or even integration, of user-centred development practices with other parts of product development and business functions. This kind of "gluing and merging" requires a framework that takes all these different business functions into account both from the procedural and organisational directions.

The integration of user-centred design in development organisation has been under discussion recently in CHI 2002 "Usability in Practice Session". Janice Rohn presented an example of a centralised organisation of user-centred development. According to Rohn, Siebel Systems has a centralised user-experience (UE) team that is responsible not only about the "piecemeal design and evaluation" of products but on the overall design (Rosenbaum & al 2002). According to Rohn, the central position of the UE team "requires strong support from senior management, plus team members who possess both design and evaluation skills." Following remarks can be highlighted from the description of the Siebel's organisation of the UE team:

- Siebel has a single, centralized UE group responsible for the entire product line
- The UE Senior Director reports to the Senior VP of Product Marketing and meets regularly with the President and the CEO of Siebel
- the UE team creates style guides that are followed by the development organisation; the style guides are evolving towards integrated design tools that support design
- the UE group is truly responsible for the product design and it both creates and "owns" portions of the product
- cross-training: for example designers are trained in user research
- customer information is available across the company via tailored customer relationship management software – access to users is therefore easy

- customer site visits per product type (results in individual site visit reports and a field summary report)
- data from user research studies are shared with other development representatives and introduced into the products via marketing requirements documents, the defect tracking system, and release plans
- user data drives the design of next generation products; this data are a combination from product marketing, sales, and other sources

In the same paper (Rosenbaum & al. 2002), Jokela, Jämsä & Iivari present how to improve and the role and impact of UCD in development projects and how to introduce UCD into development organisations. They present results from the empirical studies conducted in three Finnish companies: Buscom, Nokia, and Teamware. The approach of Jokela & al. in developing UCD practices is through capability assessments. Their experience is that the traditional assessment process[34] does not contribute enough to the overall improvement and introduction of UCD[35]. The main findings of the resulting UCD assessment and improvement method that is presented more thoroughly in Jokela's doctor's thesis (Jokela 2001) form the important characteristics of the assessment and improvement method called "KESSU UPA":

- strong integration of forthcoming UCD improvement activities to ongoing development projects/activities (e.g. a "pilot project")
- commitment of the key persons to UCD
- a management team that can make decisions about UCD
- an implementation team that carries out the UCD activities
- a series of workshops (min. 2 sessions) that have a strong emphasis on training aspects in addition to evaluation; the evaluation parts are focused more on the outcomes of the processes, not the activities themselves
- planning of UCD activities and supporting management activities
- implementation of the UCD action plan under the control and decisions of the management team.

The findings of Jokela & al. strongly emphasise the importance of commitment on all organisational levels, perhaps even more on the management level. The "committed

[34] With traditional assessment Jokela & al. refer to the SPICE/CMM-structured assessments that are performed by interviewing the practitioners about selected processes and corresponding activities.

[35] In Rosenbaum & al. (2002), Jokela & al. present that "The software developers found the models and the results difficult to understand, and the recommended usability programs were perceived as delaying the development schedule."

practitioners and managers" must be aware of the underlying UCD approach and the activities that it points out. These people act as the advocates of UCD that are needed in the introduction stage of the new approach for design. Of course, adequate knowledge about the UCD methods and tools is required as well. Another important point in the previous list is also the broad sharing of UCD information and knowledge within the development organisation since the very beginning of the introduction work.

Vredenburg presents remarks about the introduction of "full-scale UCD" at IBM during the 1990's (Rosenbaum & al. 2002) that are consistent with the findings presented by Jokela & al. Vredenburg writes about "major cultural transformation for IBM and a paradigm shift for its practitioners". Support is required not only from the senior management but also from individual developers and mid-management. According to Vredenburg, the key elements towards successful introduction of UCD included "identifying core principles, carrying out education, and integrating UCD into the company's business and development process". One problem that Vredenburg reports is the lack of "agreed-to elements that can be measured and thus managed". Their solution to this was the development of "core set of UCD metrics that summarize the key elements that are important in the management of UCD on projects at the project, division, and corporate level".

User-centredness in a development organisation can also be characterised as "design for usability". Whenever special focus is required about an important topic, one may use the "Design for X" (DFX) approach to address that important topic. One "Design for X" approach is "Design for Manufacture" (DFM). Herbertsson (1999) has dealt with this approach in his thesis entitled "Enterprise Oriented Design for Manufacture - On the adaptation and application of DFM in an enterprise". The underlying situation is well aligned with the "Design for Usability" approach: Herbertsson mentions that "*the concept of Design For Manufacturing (DFM) is being recognised as a critical issue in new product development*" as the performance of manufacturing is not only dependent on the manufacturing process itself but heavily on the product design as well. Herbertsson continues that "*In today's highly competitive environment, DFM should be regarded as a strategically important factor for the performance of both product development and manufacturing.*" Again, this reflects the situation with the design for usability approach. From this standpoint, the framework that Herbertsson outlines may well be feasible to apply in this case also – from the organisational standpoint in product development.

Herbertsson's thesis "analyses and describes how DFM theory and methodology can be adopted and applied on the basis of a wide perspective of a manufacturing enterprise". The overall architecture he suggests as a result of his work consists of three main activities: preparatory, supporting, and operational DFM.

Deriving from the activities identified by Herbertsson and coupling them with the models presented previously here in the area of human-centred development (HCD/UMM: processes and the "attitude model"), a framework with five elements is suggested here to constitute an embedding architecture or "platform" for the user-centred topics (see figure 13):

1. values and philosophy (attitude) of the development organization
2. development process (or stages in the development work / product life-cycle) that the organisation follows[36]

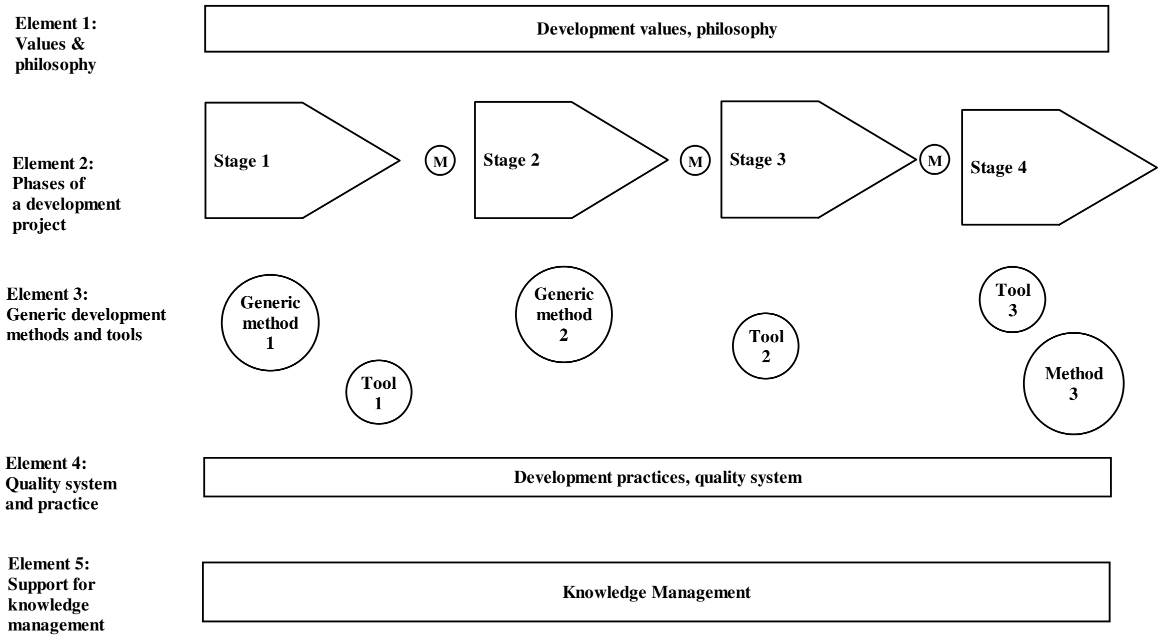

Element 1: Values & philosophy
Development values, philosophy
Element 2: Phases of a development project
Stage 1
M
Stage 2
M
Stage 3
M
Stage 4
Element 3: Generic development methods and tools
Generic method 1
Tool 1
Generic method 2
Tool 2
Tool 3
Method 3
Element 4: Quality system and practice
Development practices, quality system
Element 5: Support for knowledge management
Knowledge Management

Figure 13. The elements for implementing "Design for X" in product development are: 1) underlying philosophy of development work (attitude) in the organization, 2) generic process description 3) generic development methods and tool and 4) organization specific development practices (derived from 1, 2, and 3). Knowledge management (5) supports and enables the organization to conduct and improve the activities.

[36] Another view of the process/stage level is from the (technology adoption) life-cycle approach: products ("ramp-up", "mature"), businesses (like Geoffrey Moore's "Chasm" model; Crossing the Chasm, Moore

3. generic methods for conducting development work (the collection of applicable methods in the organisation)
4. organisation and industry specific development instructions and practices (quality systems)
5. tool support for the shared management of these issues – including user-related activities and information.

In the first part of this chapter, the elements are defined and described. In the second part, the elements are adjusted to fit the “Design for Usability” framework. In the third part, an example on how the elements can be used in real settings with the support of the collaborative information support environment is presented.

4.1 Description of the Elements

4.1.1 Element 1: Organisational Attitude / Values and Philosophy in Development

Even though the concept "philosophy" may be somewhat rarely used in the context of very technical product development, each development organization implements some kind of philosophy in its actions. Encyclopaedia Britannica (http://search.eb.com/) defines philosophy as “the critical examination of the grounds for fundamental beliefs and an analysis of the basic concepts employed in the expression of such beliefs.” The leaders and practitioners of a development organization valuate things according to their reasoned beliefs of what is important and desirable. The approach provided by the “Design for X” directs these thoughts.

Before an organisation can start user-oriented activities, its practitioners need to consider user-orientation as an important factor. Based on the reports and findings from the organisational development and management literature, support from management is especially crucial. Like any other activity, user-centred development requires resources, skilled developers, and appropriate time for the work to be useful and profitable.

1991/1999), and even companies themselves go through different stages in their life-cycle. Different stages in the life-cycle require different considerations and actions.

Illustrating explanatory examples may clarify this. In a software development organization, developers may be mainly interested in technical aspects like the efficiency of the algorithm, the novelty of the components, or the "beauty of the code". Here, the analysis of the utilisation of the outcome may be underweighted. In another development organization, designers and developers may mainly be interested in the outcome and utility of the developed tool while it is in use. They may discard the issue of appropriate documentation and it may result in non-readable, non-commented, and non-maintainable code. In a third organization, documentation is considered to be of top priority and it may become a barrier for fluent, innovative and quickly-turning improvement of the product. Optimal situation requires that all elements are in balance. If something is missing, it may well be detected as a hole in the quality system.

"Philosophy" is not, however, something that affects only issues that are only technical and practical by nature. "Philosophy" has an effect to the overall attitude and thinking that the organization and its participants have towards different issues. On the other hand, the attitudes and opinions of all the participants have an effect to the philosophy of the organization. The affect is thus bi-directional.

From the standpoint of user-oriented development work, the attitude towards users and their requirements is of central importance. There are even attempts towards the measurement of this attitude that are tied to the development of the Usability Maturity Model (“Human Centredness Scale”, Earthy 1998).

Philosophy may appear in a development organization in the topics that are required to be considered in each project. These topics may be "in-house generated" but they may well be based on external requirements. External requirements are primarily customer needs, government requirements, and legislation concerning the customer's environment. The attitude towards the importance of these issues reflects the philosophy of the organisation and its representatives.

4.1.2 Element 2: Life-cycle / Generic Development Stages

Development work is usually conducted in stages. The general model by Ulrich & Eppinger (1995) consists of five stages that are presented in chapter 3: 1) Concept development, 2) System level design, 3) Detail design, 4) Testing and refinement, and 5) Production ramp-up. In addition to the stages that Ulrich and Eppinger present, there are often stages like product launch and delivery, even support in the process models of software organizations.

A well-defined development process has several benefits for the organization (Ulrich & Eppinger 1995): A clear process structure defines the roles of all participants and instructs them to contribute in a proper stage. Quality assurance is employed through the checkpoints along the way. Milestones with exit criteria are easy to detect between the stages. Managers can follow the progress and point out corrective actions if required. At the end of the project it is possible to create a summary of experiences that can be used to improve the process in the future.

Even though the generic development process stages are normally in a sequential order, also iterative development processes should incorporate all these stages. We must do planning before we can proceed to more detailed design, implementation, and testing. All development work includes these elements and they form the basis for the development organization specific process descriptions.

The existence of element 2 in this model is supported also by our own findings: In our study of six development companies, we found out that all companies had a description of development process stages for development projects. The contents of the stages were not completely memorable but in case of need, the documentation would have been easily accessible. The development process stages were described in the quality documentation of each development unit.

4.1.3 Element 3: Generic Development Methods and Tools

If an organization aims at acting in a systematic way, it needs to develop systematic practices that produce relevant information according to the organization's (pre-defined) information needs. These kinds of methods are hard to develop from scratch in the organization itself. Methods and corresponding tools are often "imported" from outside the organization. There is a wide range of methods and tools available for various "Design for X" needs including user-centred development.

The generic methods and tools provide a library that can be user to build organization specific practices. These methods in the library need to be accompanied with appropriate supporting instructional material to enable widespread familiarisation with them. These practices appear finally under element 4 (organization specific development practices). Each method encapsulates a proportion of the philosophy behind the development practices. The most structured methods may be implemented as tools (even integrated ones) that are provided for the practitioners. Additionally, the tools used in the development organization contain elements that have been considered important for its activities.

Generic methods and tools may also be quality assurance instruments that can e.g. ascertain that required documentation and actions (like a project plan, a quality & test plan, a build plan, or a usability test) have been created or conducted and approved.

4.1.4 Element 4: Organization Specific Development Practices

If an organization wants to follow the generic process stages, it still needs to tailor them to fit its current situation. If a company does not have a certified quality system like ISO 9000, the development practices may be non-documented and they may be based on individual practitioners' skills. In such a case, development is easily conducted in an *ad hoc* way without good possibilities to follow and reconstruct what has been done. Even an organization at this level has its goals and intents: The goal may be the utilization of state-of-the-art technology that interests all practitioners. This may drive over all other development aspects.

Organization specific development practices are often documented in the form of a quality system. Quality documents provide instructions on how to perform specific activities in a systematic and repeatable way. Quality systems may contain instructions for methods that address specific issues that have been evaluated to be important for the development of the particular product or for the development organization as a whole.

The methods and tools may well be generally available commercial ones but they are always applied according to case specific circumstances[37]. The organization specific development practices are often instances of the generic methods and tools just the way every ISO 9000 solution is different from each other even though ISO 9000 itself is a very formal generic construction. Case specific circumstances relate e.g. to the current customer base, application area, and experience and skills of the practitioners.

[37] For instance, the Janice Rohn's presentation about the UCD practices at Siebel Systems (Rosenbaum & al. 2002) pointed out the use of tailored customer relationship management software that hosts customer and user information gathered by the different stakeholders in the company.

A quality system should support activities throughout the whole life cycle of a product. An organization specific quality manual may contain descriptions of procedures, more detailed work instructions, and document templates. Procedures may cover contract review, project planning and scheduling, project requirement specification, and delivery and installation. Work instructions provide guidance on storing documents to shared document storage, using standard source code templates, and using a specific development tool. Templates may contain document skeletons for interface control description, software test plan, and software version description.

4.1.5 Element 5: Support for Knowledge Management

The fifth element that deals with support for knowledge management appears to be somewhat deviated with the four first ones. It does not contribute directly to the issue on how to describe and construct a specific "Design for X" approach in a product development organization. But when it comes to implementing a "Design for X" approach, we need systematic methodology and supporting systems that can be used to introduce and sustain the conceptual ideas expressed by the selected approach.

A support mechanism for knowledge management provides a platform for presenting and putting into practice the approach with corresponding values, aims, generic methods, working practices, and instructions on tool/method usage. But more than just presenting these issues to the practitioners, a comprehensive collaborative information support mechanism can also gather feedback and experiences of the rationale of each element. A knowledge management mechanism can thus support learning and incremental improvement of working practices.

A key concept here is, of course, "knowledge management". Knowledge management is a rather new research area; e.g. the 1st European Conference on Knowledge Management was held in year 2000. Knowledge management is closely related to organisational memory that has been defined to mean a "*comprehensive computer system which captures a company's accumulated know-how, business activities, the related core competencies, and other forms of knowledge assets, and makes them available to enhance the efficiency of knowledge-intensive work processes*" (Vasconcelos & al. 2000). This definition emphasizes the role of the computer system in the management of important information that is also reflected in the works of Abecker & al. (1998) who, however, "*find that effective knowledge management requires a hybrid solution, one that involves both people and technology*". They define that the main function of organisational memory is to enhance the organisation's competitiveness by improving the way it manages its knowledge. An organisational memory should provide functions for knowledge preservation, knowledge capitalisation, and knowledge creation, and organisational learning (Abecker & al. 1998).

In many ways, a bi-directional (reading from the information support source and writing to it) collaborative information support environment may be considered as one kind of a knowledge management tool. With the concept "knowledge management" it is referred to the "knowledge creation" theory presented by Nonaka & Takeuchi (1995). According to their theory, knowledge is explicit or tacit and knowledge gets converted between these two types. From our perspective, the most interesting thing is when tacit knowledge gets converted to explicit knowledge. This conversion is called "externalisation". The other three modes of knowledge conversion are socialisation (from tacit knowledge to tacit knowledge), internalisation (from explicit knowledge to tacit knowledge), and combination (from explicit knowledge to explicit knowledge). These conversions are of central importance with collaborative information support applications. Input for knowledge creation process comes from collaborating organizations and, in user-centred development work, from users in the market.

4.2 Implementing "Design for Usability" with the Elements

We can use the elements described above to put together important issues when creating a case specific framework for "Design for Usability" approach. An example of this is presented next. Figure 14 presents an outline of the explanation as well.

4.2.1 Element 1: User-Centred Development Philosophy

If a development organization has expressed its intent in producing usable products, it already has selected a focus for its activities. Using other words we can say that the organization wants to develop tools that fulfil the requirements of an end user. This statement positions the organization towards its clients and end-users and reflects a portion of its development philosophy.

The philosophy may take a more formal form if the organization starts applying "user-centred" standards or development processes like ISO 9241-11 (1996) or ISO 13407 (1999). To assess how well the organization has positioned itself in the usability

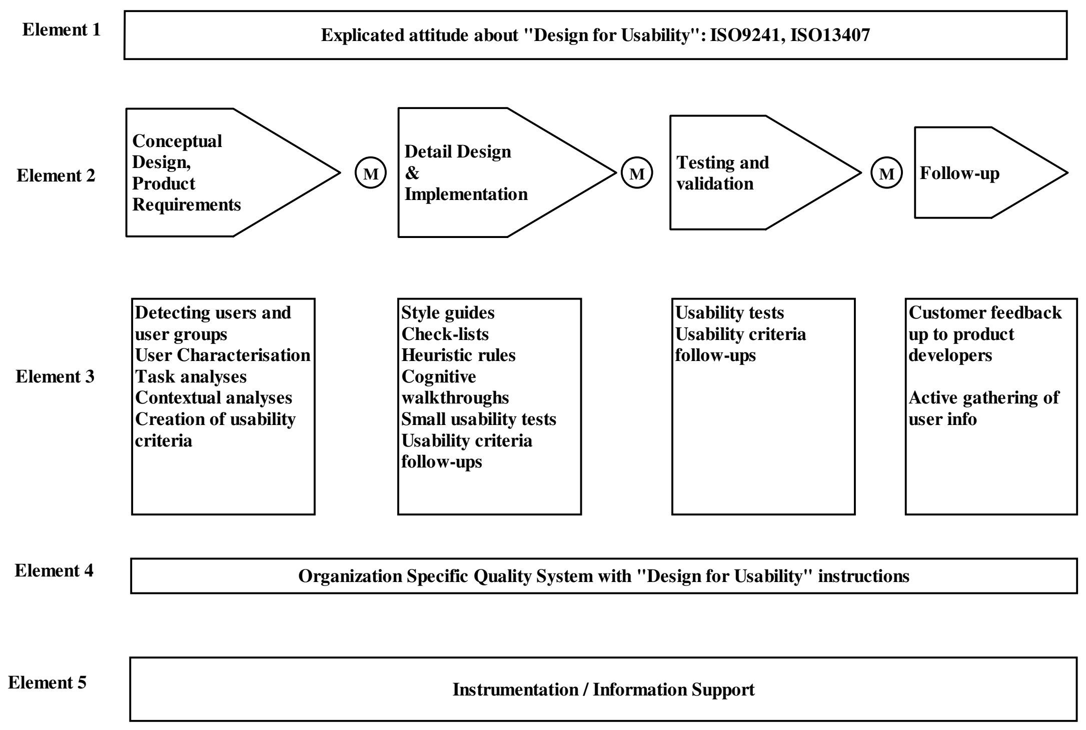

Element 1
Explicated attitude about "Design for Usability": ISO9241, ISO13407
Element 2
Conceptual Design, Product Requirements
M
Detail Design & Implementation
M
Testing and validation
M
Follow-up
Element 3
Detecting users and user groups
User Characterisation
Task analyses
Contextual analyses
Creation of usability criteria
Style guides
Check-lists
Heuristic rules
Cognitive walkthroughs
Small usability tests
Usability criteria follow-ups
Usability tests
Usability criteria follow-ups
Customer feedback up to product developers
Active gathering of user info
Element 4
Organization Specific Quality System with "Design for Usability" instructions
Element 5
Instrumentation / Information Support

Figure 14. An example of the framework with example “Design for Usability” methods filled in. The explication of user-centredness in an organisation is realized through the acceptance and use of user centred processes, standards, and maturity models (element 1). These models define an ideal development framework that can be adjusted to fit in the generic product development phases (element 2). Each development phase will then be supported with user centred tools and methods (element 3) that are put into practice in the organization specific quality system (element 4). A knowledge management mechanism for user related issues can be used to support the daily work and organisational learning (element 5).

framework, it is possible to conduct a usability assessment by applying Usability Maturity Model (Earthy 1999; for an application of this model, see Kuutti & al. 1998), Quality In Use (QIU, Earthy and Sherwood-Jones 2000; cf. Jokela 2001) or KESSU UPA (Jokela 2001).

4.2.2 Element 2: Organization Specific Stages for Development

Before we proceed to the most detailed level (element 4) in which we define the daily practices that finally enable the creation of usable products in the particular organization, we need to take a look at the proper placement of the methods in the life cycle of the product.

First, we must ascertain that we have reasonable stages for systematic and controlled management of user-centred issues. We need to be able to follow the progress and improvement of usability during the process. To do this, we must define development stages with appropriate development activities and milestones for usability inspections. The milestones may be attached to other development reviews. In those review meetings, we can, for example, apply usability guidelines and see whether the conducted development effort has resulted in improved usability. The improvement can be detected by comparing the current state of usability to the previously observed one from the standpoint of predefined usability criteria.

4.2.3 Element 3: Generic Development Tools and Methods

Usability needs to be addressed from several directions. Dix & al. (1998) present that learnability, flexibility, and robustness are the basic principles that need to be followed while aiming at usable products. Each of these principles consists of sub-elements. Learnability consists of predictability, synthesizability, familiarity, generisability, and consistency. Flexibility consists of dialog initiative, multi-threading, task migratability, substitutivity, and customizability. Robustness encapsulates observability, recoverability, responsiveness, and task conformance.

When we aim at introducing “Design for Usability”, we must provide designers with methods and tools that can address the various aspects of usability. Using just one method we cannot cover all the aspects. We need a collection of appropriate methods and tools.

Lots of methods and tools have been developed in the academia and in industry that address usability from different perspectives. Examples of such methods are interviewing with paper prototypes, usability testing, heuristic evaluation, cognitive walkthrough, contextual inquiry, and usability questionnaires like QUIS and SUMI. The selection of the tools should be done according to case specific circumstances. If the company is just about to begin to apply user-centred development methods, it may not be willing to select a very complex methodology to start with. A simple usability test may provide enough insights to the utility and productivity of the approach. With productivity we mean the ability of the methodology to produce practical improvement suggestions to the product with reasonable development effort.

The utility of the selected methods and tools needs to be evaluated in combination with the stages of development. Not all methods are useful for all stages. For instance, if we only apply usability testing at the end of a development project, it may not anymore be possible to put enough resources to the improvement of the product. Similarly, it may turn out that the required changes would be too big that at the end affect the very grounds of the system. An appropriate selection of different methods should be thus available and used throughout a project. A single step usability intervention in a development project may not provide sufficient results and desired effect.

4.2.4 Element 4: Organization specific development practices

All previously presented elements (1, 2, and 3) are somewhat general by nature. All of them require case specific adjustment and application. The case specific applications are usually quality instructions or quality systems that reflect issues presented in elements 4, 3, and 1 and they may be certified according to e.g. ISO9000 specifications.

Organization specific development practices exist, however, even without formal documented instructions. Despite the explicit definition of development values, process stages, and applicable tools and methods, the organization has practices for conducting development work. And even a development organization working in a non-systematic way applies values and development philosophy. These elements remain, however, as tacit knowledge that appears in the culture of the organization.

In this element, "Design for Usability" appears in the daily development activities. Developers have a certain attitude towards usability, users and user-centred issues. This attitude directs their work to take into account user originated features and improvement suggestions. Hard evidence about "Design for Usability" in a specific development organization can be detected e.g. in the amount of user descriptions created during a development project, analysis of users' tasks, amount and contents of usability tests, developer contacts with users and customers, and user feedback processes. These actions may be conducted according to individual developers' skills or according to detailed organization wide instructions about methods and usage of corresponding tools.

Even though the systematics differ a lot from each other in these cases, both ways to organize activities appear in element 2. If we need to evaluate the "quality of actions", we may use special assessment methods like CMM (Capability Maturity Model) or

SPICE (Software Process Improvement and Capability Determination) or apply their capability scale. For usability capability evaluation, the assessment methods mentioned under “element 1” (UMM, QIU, KESSU UPA) are available.

4.2.5 Element 5: Information Support

To support the enactment and improvement of the user centred development practices, we need mechanisms to manage information that have been gathered with the methods and experiences that have been gained in applying them. In a small organization this may happen just along normal working; people discuss things with each other, receive feedback and give advice to each other. But in a larger organization with functional lines and organizational barriers, information and knowledge does not get transferred as smoothly. We need supporting practices and infrastructure to promote this.

We may outline a concept called *Knowledge Storage* that is an approach towards this direction. It may be utilised as an organization wide carriage of user-centred information. Practitioners have the possibility to submit user-related information to the knowledge base of the system with categorizing information. Some amount of that categorizing meta-information will get saved without additional effort (practitioner's current role, name of the project, and phase of the project). Additional categorizing information can be attached to the submitted information manually. It is possible to direct the information to specific development persons with specific roles so that the information appears for those people in a specific phase of the project. Additionally, the information can be categorized with specific knowledge categories that represent topics that have been selected to be important in the development organization and in the particular project. All information is then later searchable from the system using the categorising information.

From the "Design for Usability" standpoint we can use the Knowledge Storage in following ways:

- *Knowledge categories that focus on users and usability*. We can include in the Knowledge Storage knowledge categories that address users and usability directly. These kind of category can be “user characteristics” consisting of information about user demographics (education, skills in technology, cultural environment, values), tasks, work environment and working conditions, and social environment.
- *Roles with responsibility about user centered issues*. As one basic concept in the Knowledge Storage is “role”, we can include in the system such development roles

that have direct responsibilities in doing work that addresses users and usability. Such a role may be a usability tester, a requirement analyst, or an application developer. While having an overall responsibility of the product under development, product and project managers also have user centered issues in their responsibility.

- *Phases with focus on user centered issues*. Development work consists of different stages. Some of them focus on gathering and creating requirements, others focus on validating and testing and verifying that the outcome is in conformance with the requirements and specifications. Usability testing is an example of an element within other development stages that have its focus on verifying the product's conformance to user requirements and usability criteria. Earlier in a development process may appear reviews that ascertain that the requirements really originate to customer and user requirements, not solely to in-house generated ideas.
- *Document Templates that address user-centred issues*. We can include templates in the Knowledge Storage that provide a platform for gathering and documenting user-related issues. A template can be e.g. a template for gathering important user related information when in a meeting with a customer or a user. It can also be a skeleton of an artifact that will be filled during a Contextual Design session.

As developers and other stakeholders proceed using the Knowledge Storage, they gather together information about all topics. As all categorizing information is visible to all stakeholders every time they submit a *knowledge entry* to the system, they become better aware of the multi-disciplinary nature of the development work. This can have the effect of expanding the range of issues that developers and stakeholders deal with.

4.3 Applying the Elements

To describe the practical aspects of the five elements, an example case is put here into the framework. Our case organization "A" is a middle-sized software development company that has development activities in several locations. It has a broad range of products that are developed in independent project groups.

Size has advantages and disadvantages: on the other hand there is the possibility to establish shared resources that can be used to support all development projects. With shared resources, it is possible to e.g. create a common quality system that may be introduced to all projects. A disadvantage of large size is that information and communication management is not as easy as in a small company of a few people. On the other hand, information and experiences can get gathered with a lot higher bandwidth than in a small one. The organization has possibilities to learn fast but it requires appropriate settings in its infrastructure and quality instructions.

Usability and user-centred development was selected to be a central issue in the development and forthcoming product releases. Previous in-house work in usability testing that had already been conducted for several years gave a good ground for this. Topics "user-centred" and "usable" started gaining visibility in the headlines of marketing material. This promoted the wider inclusion of user-centred activities in the organization. The user-centred development activities were identified and documented for utilisation in specific stages of the development process. Usability started gaining strategic importance among other development paradigms.

At this stage, we can see that development "philosophy" (element 1) began really include user-centeredness (strategic level). This inclusion provided ground for wider application of user-centred development methods (element 3) in specific stages of the development process (element 2) that can be included in the organization specific development practices (element 4).

When the approach gained support from management it became possible to assign resources organization wide to usability and user-centred activities (element 3). According to a closer analysis of user-centred development processes (UMM & ISO13407) it was detected that usability activities should be conducted already in the very first stages of the development process (element 1). Usability testing is important but it reveals problems only in the late stages of a development project. It is therefore a reactive method. To reach good results also in the late stages of a development project, we need to design and develop usability in the early stages of a development project. We must construct usability the same way we construct other product features. We need a shift from reactive to proactive usability practices.

An analysis of appropriate proactive user-centred design practices lead in this case to "Contextual Design" (Beyer & Holtzblatt 1998) that encapsulates user-centred requirement gathering and analysis, translation of the requirement into prototypes and into initial product specifications in the form of user environment models. The methodology was selected to be a central development method and it had an effect to the instructions according to which development work is being done (element 4). All forthcoming development projects should conduct a user-centred requirement analysis as one the very first activities in a project.

4.4 Analysis of the Model

How can different organizations be described with the elements presented in this chapter? What do the elements contribute to different organizations? If we first take a look at an organization that does not have a quality system or documented work instructions, we can see that its activities appear on element 4 (organization specific development practices). Even though the organization does not have documented work instructions, it works with some specific work practices. This can be described with element 4.

From the standpoint of improvement of working practices, the elements can point out that “as the organization is lacking a systematic description of its development stages, it should probably start thinking about how to define and introduce them”.

Even though not explicitly defined, the organization has at least a tacit philosophy that provides goals for its activity. The tacit philosophy is presented in actions and activities on element 2. But because the philosophy is tacit, the organization may be lacking systematic methods and supporting tools (element 3). This may guide the organization and its leaders to think about important issues that the organization should focus to. This work may lead towards several “Design for X” concepts that may be used to put more weight to important topics. This work may result in improved development practices.

To some extent, the elements can be used as construction pieces for a reference model for the improvement of development work. However, it does not provide instructions on detailed level activities and tasks because it does not define what development approach (or philosophy) should be selected. The selection of important topics needs to be done using other means, perhaps by applying process models.

5 Information Support for User-Oriented Development Organisation: The Knowledge Storage

In order to facilitate the activities of a user-oriented development organisation, supporting tools and methods are required. As mentioned in the beginning of chapter 4, an organisation needs to be able to perform the activities that it considers important, the ability to perform the activities.

This chapter has its focus on the support for the ability of the development organisation to perform the important activities: the Knowledge Storage. In many ways, this chapter presents the results to the original research question about the “information support system/application for user-oriented development work”.

Knowledge Storage aims at consolidating the conceptual ideas presented in the previous chapters. The Knowledge Storage concept and implementation is constructed in order to create a facilitating prototype for the management of user-oriented design activities and user-related information. Additionally, this construction provides a platform for the proof-of-concept that can be used to evaluate the ideas in a real environment.

Before the presentation of the development of the construction of the information support instrument that aims at providing support for the user-oriented development organisation, background about information support is outlined.

5.1 Information Support

In order to understand what kind of issues in user-centred development environment require information support, we first need to look at the concept “information support”. After having defined this concept, we can take a look at those more detailed “information items of user-centred development” that constitute the contents for the information support.

The amount of information has increased dramatically, especially in knowledge-intensive working environments. Product development is a typical such an environment. Individuals in this environment cannot easily cope with all the information required and available at the development time. Different areas of development require special expertise and knowledge that is often realised through multi-disciplinary teams. These multi-disciplinary teams that conduct product development are often located in different geographical locations. This multidisciplinary and multi-site activity requires information support.

Product development work can be characterised as knowledge work. Kappes & Thomas (1993) define *knowledge workers* as “people who produce not tangible products, but some form of processed or enhanced information”. Information support plays a major role in this context: “Because the primary input to knowledge work is information, work quality depends on access to high-quality sources of relevant information”. Decisions that knowledge workers make require information from various sources: regulations, databases, guidance letters and the expertise of a co-worker. “The knowledge worker's ability to access this information directly affects the quality of his or her work.” (Kappes & Thomas 1993)

Durstewitz (2001) addresses knowledge work in the context of product development from the standpoint of problem solving and decision making. Durstewitz (p. 42) mentions that “Problem solving in design is basically an interactive multi- level analysis-synthesis process. It leads to a description of an artefact (design solution) in terms of energy, material and signals (information content)”. The result of product design work, design solution, is constructed with the aid of applicable and relevant information content available. Designers can evaluate the relevance of information content through their activity environment consisting of personal expertise and experience (own memory), colleagues’ expertise (social environment), design tools, reference books, and existing documentation (information environment).

Information support that is required by knowledge workers can partially be derived from the concept of design rationale (Moran & Carroll 1996). One part of information support to the practitioners of product development is the possibility to review ideas, solutions, decisions and events that have been presented earlier in the process.

In many ways, information support can be seen as communication of right information to right persons at right time. Support for knowledge work may be retrieved by

- supporting the storing and management of data (e.g. databases)
- supporting work processes (e.g. workflow)
- supporting the organisation (arrangement) of work (e.g. roles in the workflow).

In addition to the technical approach in information management, we can take into account the concepts of "action regulation" and "action facilitation". These concepts bring in the context of work in general, the workplace, and they may be applied in the context of design work.

5.1.1 Activity Environment: the Source of Information Support

In traditional work environments and industrial tasks, support for an individual or a workgroup has been extracted from three different sources: the physical environment, the social environment and one's own memory (Vartiainen & al. 1996). In the physical environment, familiar tools, parts, and surroundings provide support. In shop-floor work, for example, components at hand suggest the right assembly procedures and tools hint the correct work methods. The physical environment restricts the employee's *action field*, the area of possible actions. Additional support has been received from supervisors and peers, who advice and train if necessary. Additionally, workers can rely on their own mental models that have formed during years through experience.

The physical workspace may be revised to provide better support. Social support is permanently useful but may not be available when needed. Therefore, the support provided by the physical environment (tools) can be enhanced by expanding the concept of activity environment to include the information environment.

The action regulation theory (Hacker, 1973) provides one model to describe person's interactions with the activity environment. Work behaviour is composed of separate actions directed at certain goals. An action is a whole of sensory, cognitive and motor processes. Actions are executed according to action programs or mental models at intellectual, perceptual-conceptual, and sensorimotoric levels. Actions have certain phases: orientation, planning, execution, and control.

We may argue that Hacker's action regulation theory has been designed from the standpoint of "shop-floor work". We may, however, expand the model to cover design and information work. With this expansion, we must broaden the resulting actions from short-length physical events to longer activity sequences, processes. In development work, action regulation can be translated in the following way.

In order to reach the goal of the design work (or part of it), designers need to find the reasons and arguments for specific features (orientation). To work in a systematic way, they must specify the functionality of the resulting system and its implementation process (planning). Then, they proceed according to the plan (execution) and the result will be evaluated (control). In order to accomplish this fully, the designers need *information support* in all stages. Information support may cover issues like

- What do we aim at with this design assignment? (requirement)
- What is the work product / deliverable? (goal of the development work)
- How can we achieve the result in an efficient way? (process)
- What are good working practices to accomplish the task? (methods)
- How can we evaluate the quality of outcome? (evaluation)

The action facilitation model (Roe & Meijer 1990) proposes even more specific principles for task support. Action facilitation means that the individual's work performance is improved or at least maintained, while his or her mental or physical efforts decrease. The action facilitation model looks for task support from the "activity environment", i.e. workplace, in every phase of a task: Orientation, design of action programs, decision making, execution of action, supervisory activities and use of feedback. For example, the orientation on the tasks and the design of action programs should be facilitated by presenting relevant information and offering help in analysis, problem solving and decision making.

To address this process from the standpoint of an individual designer interacting with an information support system, we can use the seven-stage model of interaction presented by Norman (1986). Norman presents that the interaction between the system and the human consists of an interaction cycle presented in figure 15.

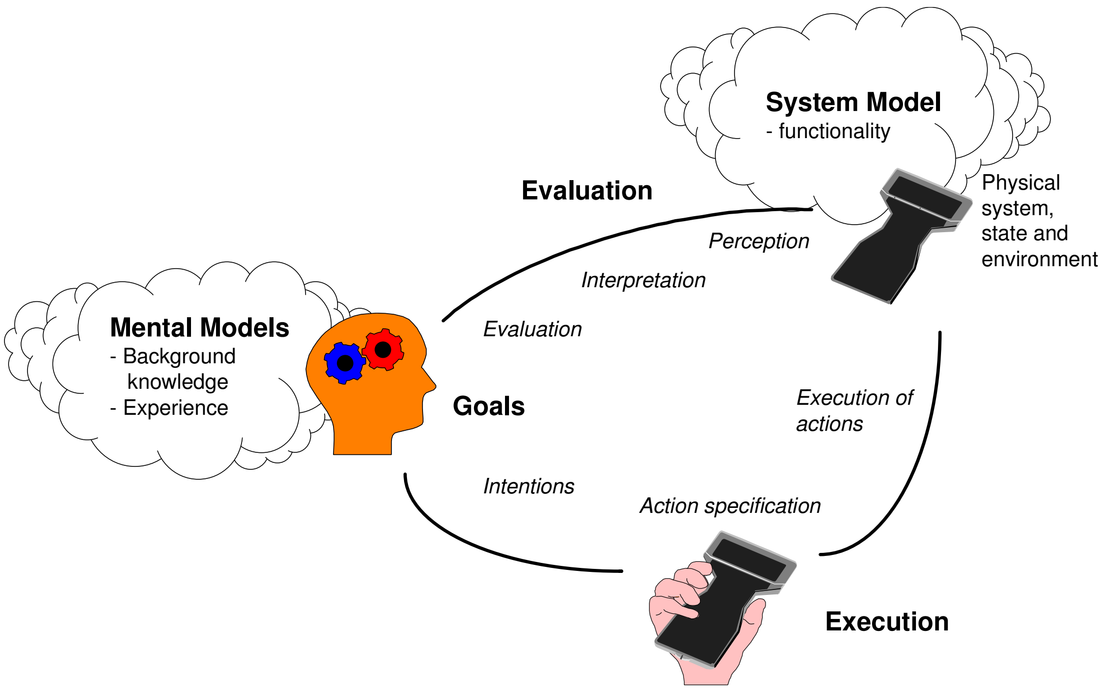


Figure 15. The cycle of interaction between the user and the system as presented by Norman (1986).

The user of the interactive device or system perceives the (usually mainly) visual, audible, and haptic elements that the device "communicates". Based on the perception, the user interprets and evaluates the status and state of the system. According to his or hers goals, background knowledge and experience, the user forms intentions about how to proceed towards the aimed goal. Then, the user specifies practical actions that, according to his or hers experience or expectations, lead to the goal or an intermediate goal. The last stage in this interaction is the execution of the specified actions. Then the interaction cycle starts from the beginning again as the device provides appropriate feedback to the user.

In the context of knowledge work, this model implies that both the medium that is used to provide the user with appropriate information and the information itself needs to be in a form that is easily perceivable and can be easily interpreted and evaluated. Thus, the information should conform somehow to the existing mental models that the developer has. The use of familiar information, searching and presentation methods can facilitate the developer's work.

The action facilitation approach (Roe & Meijer, 1990) suggests that in order to know, what kind of information support is needed, the related tasks and actions, the structure of the employees' mental regulation and the context they work in have to be analysed. This way the support provided will lead to a better understanding of the requirements for the product and to better functional principles and technical properties of the product under development. The resulting qualified cognitive working style appears as a critical individual performance plan, answering many a "why" question. In the context of performing shop-floor tasks, this kind of a working style has been found to lead to an effective high quality performance (Teikari & al., 1985).

In order to develop a qualified cognitive working style, employees are presumed to have a qualified mental knowledge structure: inner models of the task and flexible mental strategies to use them. It is the information environment of the workplace that is responsible for maintaining this knowledge model.

Multidisciplinary organisation of work has several problems. Information accumulates to individual practitioners' experience. This experience is often tacit. When a person leaves the development organisation, a great amount of knowledge is lost. In specific cases, the knowledge may be crucial for the organisation, and it may take a long time to get back to the original level.

Additionally, project based multidisciplinary development way to organise development work builds up barriers between project groups and functional divisions of the organisation. Developers may "invent the wheel" again and again because of lack of interaction between other project groups.

As with many other development topics, the management of user-centred information suffers from the lack of interaction between project groups. This issue is one of the most important underlying findings that promote the importance of shared information support. Increase of information exchange in user-related issues can be seen to improve the efficiency of development work: Even though development groups tend to develop products that are separate from each other from technical standpoint, they may well be developing products for the same audience. Same users may use different products in the same environment. If one project group has gathered information about its users, this information could well serve other development projects. They could utilise user descriptions, task descriptions, environment descriptions, results of conducted usability tests, user interviews, and user observations that already exist in the organisation.

We have a need for concepts and tools that can address these issues. We need tools that can be used organisation wide to share and make explicit the information that has been created in a single project.

Implication for this work. For an information support system, the “activity environment” implies that the system must provide information that is of relevance to the current tasks of the practitioners. To prevent information overflow, such a system or mechanism needs to filter out unwanted and non-relevant information. The system may provide either automatic means for intelligent filtering, or it may provide possibilities for all practitioners to provide sort of filtering rules that manage the visibility of the information during the process for different stakeholders. The goal is to provide all stakeholders with appropriate context dependent information throughout the development cycle. In the design of such features, one has to take into account the requirements set by the cycle of interaction. This issue provides partial answer to research question “what is information support”.

5.1.2 Design Rationale: Memory of Design History

Partially, information support can be seen to get realised through “memories”: In the individual level, developers remember well-working solutions and good ideas. From the organisational perspective this remembering requires sharing of important information and experiences. Additionally, organisational remembering requires documentation.

Important issues in managing development information in general are tracking of product features, development decisions, and the different solutions that have been addressed during the development work. The different solutions may have been implemented in the product or they may have been discarded for some reasons. *Design rationale* encapsulates these issues. The characteristics of design rationale need to be taken into account when designing an information support system for product developers’ use.

Design rationale has been defined as (Moran & Carroll 1996):

1. An expression of the relationships between a designed artefact, its purpose, the designer's conceptualisation, and the contextual constraints on realising the purpose
2. The logical reasons given to justify a designed artefact
3. A notation for the logical reasons for a designed artefact

4. A method of designing an artefact whereby the reasons for it are made explicit
5. Documentation of (a) the reasons for the design of an artefact, (b) the stages or steps of the design process, (c) the history of the design and its context
6. An explanation of why a designed artefact (or some feature of an artefact) is the way it is.

A design rationale system may provide support for design, maintenance, learning, and documentation (Lee 1997). Additionally, a design rationale system provides support for project management in the form of collaboration support, requirement engineering support, and reuse support.

Design support helps designers track the issues and alternatives that are being explored. *Dependency management* can display issues that depend on the current issue or issues that are related to the current issue. More complex systems can actually detect conflicts among various design constraints. *Collaboration and project management* provides the participants of the project with a common and shared workspace. Within this workspace designers can interact with each other. By interacting with each other, designers can form common vocabulary and project memories. This way it is easier to negotiate and reach consensus. *Reuse support* offers designers an index to past knowledge. There may be similar design problems that have been encountered earlier.

Maintenance support is gained through the explanations that have been stored in the previous design decisions.

Learning support. In most advanced systems, both technical systems and humans can learn. A system may be able to suggest more optimal solutions when it detects a non-optimal structure. The most common way to support learning is, however, through knowledge that has been saved in the past to the knowledge base. For instance, new designers can scan through this information and learn from it. A design rationale system supports learning by presenting the designer all design decisions to that are stored in the system. Each piece of information builds up the knowledge base and provides designers with more comprehensive information.

Documentation support. A design rationale system can be used to automatically generate documentation. As documentation is something that reflects the actual events of the development project, a design rationale system gathers documentation in real-time. It has been mentioned that the documentation perspective has been the most successful in the implementation of design rationale systems.

Lee (1997) presents that following approaches can be used in the creation and implementation of design rationale systems:

1. **Reconstruction**: introspection, post-interviews, capturing rationales in raw form.
2. **Record-and-replay**: rationales are captured as they unfold, e.g. as designers use a shared database to raise issues, propose alternatives and criteria, and enter evaluations.
3. **Methodological by-product**: designers follow a certain method and the system can capture and generate rationale from the refinements and reformulation.
4. **Apprentice**. System "looks over designer's shoulder" and asks questions when it disagrees or does not understand the actions. Rationale is then generated by the system based on its domain knowledge and designer-supplied justifications.
5. **Automatic**. A system generates design rationales from an execution history according to its knowledge base and rules.

Knowledge Management. The ultimate result in applying a design rationale mechanism or system provides means towards knowledge management. Nonaka & Takeuchi (1995) present a theory about knowledge creation and knowledge management. This theory provides an overall framework in which a design rationale system should operate.

According to the theory, knowledge is explicit or tacit. Knowledge gets converted between these two types. When tacit knowledge gets converted to explicit knowledge, externalisation happens. The other three modes of knowledge conversion are socialisation (from tacit knowledge to tacit knowledge), internalisation (from explicit knowledge to tacit knowledge), and combination (from explicit knowledge to explicit knowledge). With a design rationale system that records the designers’ experience, we can support the conversion from tacit knowledge to explicit knowledge (externalisation) and the conversion from explicit knowledge to explicit knowledge (combination).

According to Nonaka & Takeuchi, organisational knowledge creation happens in five phases: I) sharing tacit knowledge (socialisation), II) creating concepts (externalisation), III) justifying concepts (internalisation), IV) building an archetype, and V) cross-levelling knowledge (combination). This process is interactive and the initial information for this process comes from collaborating organisations and from users from the market. The process results in explicit knowledge in the form of advertisements, patents, products and services.

Implication for this work. Design rationale provides concepts that can be used to implement traceability and history of design decisions. From the standpoint of practical implementation and introduction of design rationale features, the approaches “record-and-replay” and “methodological by-product” seem promising. If we can capture the design ideas and “rationales as they unfold” (“record-and-replay”), no additional overhead work is required to store that information for later use. A properly designed network-enabled application that provides a shared database to raise issues, propose alternatives and criteria, and enter evaluations may implement this approach. In a strictly guided development environment where the quality system provides detailed instructions, the “methodological by-product” approach can also be used. If designers follow a certain method and document issues with a specific documentation procedure, the system can capture and support the generation of rationale. This issue provides partially the definition of the key issue “what is information support”.

5.2 Collaborative Information Support Tools for Usability Engineering

In this chapter, related research about collaborative applications for engineering are presented with the focus on the applicability of these applications and concepts for user-centred development activities, usability engineering.

As more and more of product development is nowadays organised in a decentralised and distributed way[38], different types of tools for collaboration play an important role in supporting the development. The exchange of product, production, design and other engineering information is of central importance in multi-site organisation operating on an international basis.

This decentralised and distributed way of working may well be reflected in the organisation of usability engineering activities as well. This can be seen to be especially important when the product is targeted to an international audience with cultural

[38] This is reflected in the topics of several international academic forums and conferences such as CIRP (International Institution for Production Engineering Research), TMCE (Tools and Methods of Competitive Engineering). There are also EU projects (e.g. WISE) addressing this topic as well. As an example, Takalo & al. (2000) refer to this phenomenon.

differences. The construction of the product should reflect this characteristic. An international team of developers may be able to gather user-related and user-centred requirements for the product. Similarly, they may be able to conduct location-specific usability evaluation to the product or even to product requirements.

Examples of multi-site working in usability engineering are the gathering of user requirements for the product (in the beginning a development project) and the exploration of user experiences for the evaluation of the product and the assessment of its usability, applicability, and utility (at the later stages of the development project). According to usability principles[39], these activities should happen locally at each market site even though the development work may be conducted centralised in one location.

Geographical decentralisation of development work is, however, just one aspect of distribution of development work. A categorisation of computer-supported co-operative work illustrates also other dimensions for this.

5.2.1 Categories for Computer-Supported Co-operative Work

CSCW as Research Field. Bannon (1993) has presented "An Initial Exploration" about CSCW. About the origins of the CSCW research field, Bannon notes the invited workshop arranged in 1984[40] and the first open CSCW conference was held in 1986 in Austin, Texas. Since then, CSCW has been addressed in several other forums (conferences and workshops) such as "collaboration technology, group decision support systems and multi-user systems". Bannon also mentions "many journals in the areas of office systems, human-computer interaction, decision support, and software engineering now include CSCW in their list of topics"[41]. He concludes that CSCW can be considered as a legitimate academic research area.

As the broad range of different academic forums where CSCW appears point out, it cannot be considered as one united research field. In many ways, it is a dispersed one having a great amount of multi-disciplinary approaches within. In his article, Bannon (1993) divides the research field into "Human and Social Factors" and "Technology Factors". The human and social factors raise questions about the organisational

[39] Real users need to be contacted and involved e.g in contextual inquiry sessions and in usability tests.

[40] Arranged by Irene Greif and Paul Cashman (Bannon 1993).

[41] Additionally, Bannon mentions journals that are focusing directly to CSCW: "Organizational Computing" (Ablex), "CSCW: An International Journal" (Kluwer), and "Collaborative Technology".

environment (ways of organizing and coordinating work activities, better integration, up-to-date information, easy access to information), people's expectations (individual level; appropriate user interfaces; HCI), and the social aspects of computer use (the situated nature of everyday activity, aspects beyond the human-computer dyad). The technology factors are "the search for new software markets" (groupware) and "technological developments" (networking; greater connectivity between people, locally, regionally, and globally).

Drawing on these basics, Bannon continues and defines five ways to look at CSCW:

1. CSCW as simply a loose agglomeration of cooperating and at times competing communities
2. CSCW as a paradigm shift
3. CSCW as software for groups
4. CSCW as technological support of cooperative work forms
5. CSCW as Participative Design

Using these categories, approach 4 is perhaps the nearest one for the development of a collaborative computer system that takes into account the organisational and social aspects. In this category "the emphasis is on understanding cooperative work as a distinctive form of work (Schmidt 1991[42]), and on supporting these cooperative work forms with appropriate technology" (Bannon 1993). However, from the standpoint of the participatory design approach of the collaborative system, category 5 is of importance as well. Even though Bannon (1993) criticizes that equating the CSCW area with participative design practices is "a clear mistake that can only add to confusion surrounding both fields". It can be easily seen that, yes, participatory design is not equal to CSCW, but the design of CSCW applications certainly requires the use of multi-disciplinary participatory design practices. And even if we cannot make the one-to-one correspondence with these two fields, we can see and understand the very close relationship with these two areas.

CSCW as Application Framework. Another way to look at CSCW is from the direction of the applications and their types. Bannon (1993) divides CSCW applications to three groups that contribute to the functions of CSCW: communication, coordination/procedure, and meeting support. Communication function enables people to exchange information with each other about topics of mutual interest (working, hobbies, discussion). Coordination/procedure provides means for controlling the

[42] Cf. Bannon (1993).

information flow (e.g. workflow) in an appropriate way. According to Bannon (1993), meeting support may contain applications that allow shared calendaring & meeting scheduling systems as well as applications that allow users to jointly work on documents and share the same views on these documents (WYSIWIS: What You See Is What I See). One may argue that shared calendaring applications belong to coordination/procedure whereas meeting support aims at supporting simultaneous on-line collaboration between collaboration participants. Especially this last characteristic can be analysed and discussed via time-place dimensions.

Another approach to model the functions of CSCW is via time and place[43]. In the time-place matrix, the CSCW applications are divided into four groups:

- *Same place – same time*. These applications provide support for collaborative working sessions in which all participants are present. Examples of such situations are workshops and face-to-face meetings. Applications that fall into this category may be computer-facilitated meeting rooms with networked computers providing a shared discussion/innovation forum or electronic walls
- *Same place – different time*. These applications provide information that is related to working in a specific place. Such a place may be an operator room in a power plant, a telephone exchange desk, or a customer support centre. These kinds of applications may provide information about events and activities that have taken place during the time when an individual person has not been present but needs to know or get up-to-date what has happened during his or hers absence. Logbooks and diaries are typical examples of such applications.
- *Different place – same time.* This category outlines a system that provides synchronous collaboration at separate geographical locations. People can work together and share some work artefacts with each other without the need to be in the same place. On-line video conferencing is a typical example of this; even telephone calls may be categorised to fall into this. On-line collaboration with shared whiteboards and application sharing are typical examples of applications in this category.
- *Different place – different time*. Applications that enable people to collaborate without the need to be in the same place and without the need to be present at the same time are identified as asynchronous collaboration tools. Electronic mail is perhaps the best known application in this category, but applications like newsgroups and shared file storages belong to this category as well.

[43] E,g, in Applegate, L.M. (1991).

Contemporary and future CSCW applications ("CSCW Environments") will provide functionality that combines the characteristics presented above. On the other hand, many of the CSCW-related functions will be embedded into traditional applications like word processors and graphics packages.

Davis & al. present three major issues in the area of building CSCW applications: 1) workplace studies employing ethnographic methodologies, 2) workflow systems (either script type or map type[44]), and 3) the actual building of the systems that support collaborative work[45].

5.2.2 Web-Based Asynchronous Collaboration

When drawing from the technological origin of CSCW, the networking infrastructure and the groupware applications provide one starting point. In addition to the general conceptual analysis of the CSCW field, Bannon (1993) also presents early experiences with CSCW systems and presents several applications and application types that either originate from the CSCW field itself or are by their nature considered as contributing ones[46]. Since 1993, however, the technological infrastructure that can be used for the construction of CSCW applications, especially applications that can be used widely in the industry in addition to academic community, has changed tremendously.

---

[44] Davis et al. (1999) presents that the script type of workflow is "*automated workflow systems with the ability to deal with contingencies that script a set of sequential and parallel activities"* and the map type of workflow is "*shared information spaces that provide contextual guidance and support*" (Schmidt, 1997; cf Davis et al. 1999).

[45] Davis (1999) lists Examples of such system prototypes: "*gIBIS (Conklin, 1989) and Answer Garden (Ackerman and Malone, 1990) for capturing design rationale and maintaining organizational memory, Collaborative Work Environment (CVE) to facilitate real time group work with shared objects and artifacts (Hindmarsh et al., 1998), Oval, a tailorable system for cooperative work (Malone et al., 1992) and EGRET to assist in exploratory settings (Johnson, 1992).*"

[46] Bannon (1993) lists following application categories and applications: applications focused on communication (Electronic Mail, The MIT Information Lens Project, The Coordinator System), meeting support (Shared Calendaring & Meeting Scheduling Systems, Xerox PARC CoLab, Ventana GroupSystems,), coordination /procedure Systems (DOMINO Office Procedure System, XCP). As an example of the forthcoming applications ("Towards CSCW Environments"), Bannon presents Lotus Notes. Bentley et al. (1997b) present that Lotus Notes provide a wide variety of CSCW services but mainly within one organisation only. There are also many issues with integration and interoperability with different platforms that must be addressed.

Internet is perhaps the most influencing technological leap that has enabled the development of more versatile and innovative CSCW applications. Since 1993 the emphasis in corporate networking has shifted in many ways from LAN (Local Area Networking) towards WAN (Wide Area Networking). In practice, this means the dominance of IP-based (TCP/IP) networking especially in the global corporate networks but is very often used also as a one-site-specific solution as well (intranets). From the technological standpoint, most recent CSCW applications make use of this technology.

Bentley et al. (1997a) have analysed the possibilities of the world-wide-web as an enabling technology for CSCW in the context of their application called Basic Support for Co-operative Work (BSCW) (presented also in Bentley & al. 1997b). One of the most important issues that they present (1997a) is that world-wide-web makes it really possible for CSCW applications to step out of the laboratory settings into real industrial use. The networking infrastructure introduced by the Internet and world-wide-web provided an integrated platform that was not dependent on the manufacturer-specific solutions as had been the case very much earlier. The client-server, or browser-server architecture provided means for creating applications to a large user population without overwhelming needs to manage the client side application.

Despite the intriguing possibilities that the world-wide-web offer, constraints for the applications built on top of world-wide-web were detected: “We suggest that the Web is limited in the range of CSCW systems that can be developed on the basic architecture and, in its current form, is most suited for asynchronous, centralised CSCW applications with no strong requirements for notification, disconnected working and rich user interfaces” (Bentley et al. 1997b). This finding points out that the pure browser-based utilisation of web for CSCW applications is best suited for asynchronous (“off-line”) collaboration instead of synchronous (“on-line”) collaboration[47].

Being one of the first web-browser-accessible information sharing applications, BSCW offers an interesting starting point (Bentley et al. 1997b). The basic services that BSCW offers are storing of documents, pictures, URL links (to other web pages or FTP sites). Additionally it provides a space for threaded discussions and storing of member contact

[47] Here, of great importance is the definition of “pure browser-based”. We are all familiar with the current possibility to create e.g. notification-enabled applets and on-line collaboration messengers on top of the IP-networking. These applications, however, require the use of (platform) specific clients (applications, applets, or plug-ins that may well be downloadable with a browser through internet; for instance peer-to-peer networking and messaging applications).

information. A description of the functionality of BSCW is presented as follows: "*The contents of each workspace are represented as information* ***objects*** *arranged in a folder hierarchy. Members can transfer (upload) information from their machines to the workspace and set access rights to control the visibility of this information or the operations which can be performed for others. In addition, members can download, modify and request more details on the information objects by clicking on buttons–HTML link that request workspace operations from the BSCW server, which then returns a modified HTML page to the browser showing the new state of the workspace.*" (Bentley & al. 1997b; p. 831).

The previous description captures together many aspects of computer-supported co-operative work. Basically, it presents BCSW to be web-accessible shared document storage with a web-based newsgroup built-in. This is, however, somewhat generic by nature and leaves open questions that relate to specific working situations, activities and application areas. Is document sharing and discussion a sufficient solution for a wide range of work? Is there a need for enhanced information structures that address the communication, coordination/procedure, and meeting support? If so, what would they be? In the context of this thesis, enhanced information structures that provide support for usability engineering activities are considered important.

5.3 The Knowledge Storage Concept

According to the assumption on which this thesis is very much based on, appropriate tools are required to support the utilisation of user-centred development methods in a development organisation. Additionally, in order to gain early insights about the possibilities of working procedures and real-life activities that can be evaluated; a technical system that realises the concepts presented in the previous chapters is needed – at least for research purposes. This chapter presents the development and evaluation of the technical system called Knowledge Storage.

5.3.1 Development of the Knowledge Storage

Due to the reason that no readymade application has been available that could have been used to present a proof-of-concept for the ideas presented in the previous parts of the thesis, one part of this work has been the development of such a system. The system implements the basic concepts of the framework. This part of the thesis presents the development of the Knowledge Storage information support application.

5.3.1.1 Methodological Considerations

The Knowledge Storage was created by applying Contextual Design as the design method. Contextual design is a user-centred development method for analysing the end-user needs in the early stages of a development project and for creating the basic structure of the user interface to the product. The basics of Contextual Design have been presented already earlier in this thesis and it is not reproduced here. The Contextual Design stage was followed by the implementation of a working prototype of Knowledge Storage. Finally, the prototype was tested in a real pilot case for validating the concept and the implementation. The following chapters describe these stages.

In their book "Knowledge-based decision support systems"[48], Klein & Methlie (1995) propose that the implementation of knowledge-based decision support systems (KB-DSS) consists of 12 stages (Klein & Methlie 1995 p. 355). Even though the construction of Knowledge Storage did not follow slavishly the methodology proposed by them, we can find similarities in the process that was used in the creation of Knowledge Storage.

The very first two stages of the KB-DSS implementation process aim at (1) understanding the users' goals and understanding and (2) defining the problem boundary. In the construction of Knowledge Storage, these stages were addressed with the discussions, interviews and document analyses in the target organisation. The (3) descriptive modelling decision process, (4) definition of a normative decision process for the problem, (5) specification of changes in decision making, (6) selecting which part of the decision process to support, and the (7) functional analysis of the KB-DSS (Knowledge Storage) was covered by Contextual Design. The (9) design and implementation of the KB-DSS software that takes input from (8) selection of development environment was done right after the concept of the Knowledge Storage had been created. These stages are followed by (10) testing with user-designers (in our case "usability testing") that precedes stage (11) education and generalisation to other users. The last two stages are (12) evaluated and controlled. In the case of Knowledge Storage, evaluation and control happened through piloting the prototype construction.

A good previous example of the constructive approach used in the development of Knowledge Storage is presented by Davis & al. (1999). They present the creation of a shared information space that supports collaborative design work. The environment in

[48] The research field "decision support systems (DSS)" is a related research area to CSCW.

which Davis & al. have done their work can be seen to be comparative to the environment and research approach that is selected to provide the construction section in this thesis as well.

The study by Davis & al. was done in a constructive manner: Based on the requirements gathered by the study, the LIRÉ implementation was created. In practice, LIRÉ ended up being a software application that runs on top of a web server making use of Oracle relational database management system.

In their report they position their work in the fields of group support systems (GSS) and CSCW. They also present that "a growing stream of research has targeted the problem of providing computer support for both co-located and distributed work". Their work describes one case in the field of design, implementation, and testing of computer systems that support asynchronous, distributed, collaborative work. With their work, they aim at exploring and extending the notion of shared information spaces.

In addition to the implementation, an important issue was also the realistic applicability of the system. Davis & al. conducted an initial evaluation of LIRÉ was done with 20 students using the system using a usability survey instrument. A more formal assessment was done together with the industrial partner "Delta". This arrangement of the study points out the researchers' desire towards practically applicable results.

According to the evaluation of the outcome "the baseline features of the system have met with ready acceptance", but Davis & al. point out that "several criticisms, concerns, and managerial issues emerged from the interview responses and ongoing discussions". From these remarks, it can be seen that creating a (prototype) system that aims at supporting product developers' activities will provide partial improvements to their work but certainly not solve all the problems in the selected field. In that sense, a realised prototype tool itself is capable of uncovering requirements for not only for forthcoming implementations but also for restructuring and refining the concepts behind the prototype. With this kind of research-oriented instrumentation and real-life exploration it may well be possible to dig out practical considerations and hindrances that may be overcome in the future research projects. This type of research approach can be seen to provide means to overcome the problem of organisational changes vs. tool/methodology improvements mentioned earlier.

The research presented by Davis & al. also points out the fact that doing research and studies about CSCW is multi-disciplinary by nature: it incorporates ethnographic workplace studies combined together with the aim towards technical implementation that is often approached with constructive and methods making use of some kind of intervention studies. The combination of different research traditions and approaches make it somewhat difficult to position these kinds of studies. This is a true challenge for researchers in this field and deserves further discussion.

What is different in this thesis when compared to the study of Davis & al.? The first one is in the constraints and focus of the targeted and supported design activities. From the standpoint of the implemented prototype, the thematic focus in this work is in the support for user-centred development activities. In this section, there is a strong basis from the tools research in the HCI research field. Another difference is the consideration about the organisational environment that aims at outlining the appropriate "hosting" organisation that is capable of utilising such an information sharing system. This section comes from the more recent discussion in the HCI area.

5.3.1.2 Results from Initial Questionnaire

In order to identify, ensure, and get a grasp of the main challenges in the development organisation of the case company (empirical study III), a questionnaire survey (see appendix A) was conducted[49]. The questionnaire was supposed to verify the requirements that were found out during the analysis of the requirements for Knowledge Storage as well as provide the basis for the introduction of the Knowledge Storage. We received 13 answers to the questionnaire. All answers were from participants from the selected distributed development project who were expected to be the users of the pilot system. The participants represented following roles that are also presented in the communication analysis matrix (figure 16): competitor analyst, consultant (N=2), developers in different roles (N=6), and managers (N=4).

The answers to the questionnaire strengthened the initial impression about the main problems. The main results based on the answers to the questionnaire are presented

[49] As a comparison, Davis & al. (1999) gathered the requirement data for the development of the LIRÉ system with information flow analysis from the developers. The gathering of data and analysis was done with a questionnaire and an interview. The questionnaire consisted of questions about tasks, documentation/information needs (from whom, to whom), format of the information, storing of information, tools used, and inefficient tasks.

below. The results about the communication between stakeholders are presented in figure 16.

User information does not reach developers well enough. Even though information from the customers and users is considered to be an important issue, this information does not fluctuate to all required development stakeholders. Customer and user information was considered to be scattered around in the distributed and decentralised organisation. The organisation as a whole may even have the construction pieces of this information but there are no well-defined practices that could bring this information to the reach of the appropriate developer or group in such a way that it could to be included easily in the design of the product under development.

Information is not shared between projects well enough. This issue is related to the previous one but looking from another perspective. Sharing of product and process information between project groups and organisational divisions does not happen as smoothly as it should even when information sharing is excellent inside a single project (this is illustrated in the next figure that presents the results of an additional information flow study that was conducted to support the preparation of the pilot case). The organisation doing the development work was considered to be, according to some answers, "a bunch of garage shops" that are not tightly interconnected. Individual project groups appeared to work rather independently from each other. Experience about product information, development issues and working practices does not always accumulate in a desired way in this environment. In addition to the separate organisational units, there are also separate function[50]-specific information systems that contain information that could contribute to the overall development work. This *information sharing* issue has, therefore, both technical and organisational aspects.

Information sharing is not only a question about knowing what others know. Sharing is considered to be also about analysing the information and findings together. With that activity, improved and even completely new ideas about product solutions may emerge more efficiently than with separate and diverse function-internal analyses. Different types of group work and brainstorming activities that are conducted overall in development organisations may provide support for this interpretation. Similarly, there is a specific sharing activity in the early stages of Contextual Design as well.

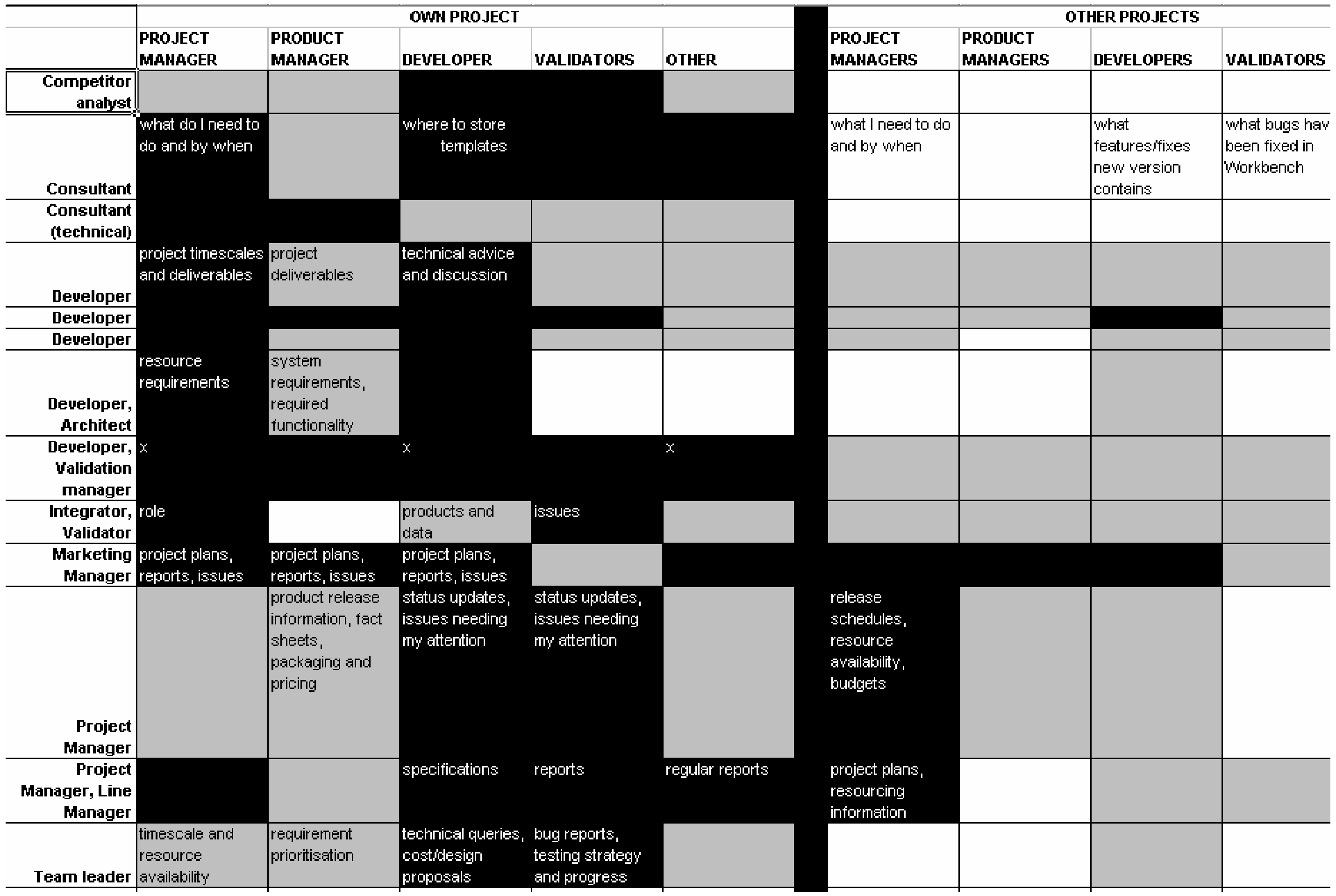
[50] Here, the term "function" refers to an organisational activity that is conducted by the assigned persons (i.e. an organisational unit with its responsibilities).

Traceability. It is not easy to go back to history and reconstruct a decision situation: Why did we select this feature? What kind of analysis was in the background of this decision? What was the rationale behind this decision? These kinds of issues are not always documented systematically and therefore the information will remain tacit.

Despite the fact that the focus of our research project was in the improvement of usability design activities, we decided to include the topics mentioned above in our study and in the improvement work that we will conduct in the last stage (piloting) of the project. Concepts of *design rationale* and *knowledge management* stayed among throughout the project as the problems were detected.

Communication. In addition to the answers about the most important challenges that may be addressed with the Knowledge Storage, an analysis of communication flows and communication frequency was done with the questionnaire. The results of this part are presented in figure 16. The aim of this analysis was to gather information about the "baseline communication structure" in the organisation that might be then evaluated against the communication that takes place through the Knowledge Storage. In figure 16, different roles are in the columns and rows of the matrix and the information that is being transferred between the two roles is presented in the crossing cell. The colour of the cell indicates the frequency of interaction between the two stakeholders (black=daily, grey=weekly, white=seldom).

The results show that, obviously, most active communication takes place within a single development project. Communication outside own project is more infrequent and not so specific. The contents of the interaction relate to control of development work ("what do I need to do and by when", "status updates"), the outcome and results of the project ("deliverables", "specifications", "requirement prioritizations"), and planning ("project plans", "release schedules").



	OWN PROJECT					OTHER PROJECTS			
	PROJECT MANAGER	PRODUCT MANAGER	DEVELOPER	VALIDATORS	OTHER	PROJECT MANAGERS	PRODUCT MANAGERS	DEVELOPERS	VALIDATORS
Competitor analyst	▨	▨	■	■	▨	□	□	□	□
Consultant	■ what do I need to do and by when	▨	■ where to store templates	■	■	□ what I need to do and by when	□	□ what features/fixes new version contains	□ what bugs hav been fixed in Workbench
Consultant (technical)	■	■	▨	▨	▨	□	□	□	□
Developer	■ project timescales and deliverables	▨ project deliverables	■ technical advice and discussion	▨	▨	▨	▨	▨	▨
Developer	■	■	■	■	▨	▨	▨	■	▨
Developer	■	▨	■	▨	▨	▨	□	▨	▨
Developer, Architect	■ resource requirements	▨ system requirements, required functionality	■	□	□	□	□	▨	□
Developer, Validation manager	■ x	■	■ x	■	■ x	▨	▨	▨	▨
Integrator, Validator	■ role	□	▨ products and data	■ issues	▨	▨	▨	▨	▨
Marketing Manager	■ project plans, reports, issues	■ project plans, reports, issues	■ project plans, reports, issues	▨	■	■	■	■	▨
Project Manager	▨	▨ product release information, fact sheets, packaging and pricing	■ status updates, issues needing my attention	■ status updates, issues needing my attention	▨	■ release schedules, resource availability, budgets	▨	▨	□
Project Manager, Line Manager	■	▨	■ specifications	■ reports	■ regular reports	■ project plans, resourcing information	□	▨	▨
Team leader	▨ timescale and resource availability	▨ requirement prioritisation	■ technical queries, cost/design proposals	■ bug reports, testing strategy and progress	▨	□	□	▨	□

Figure 16. Results from the information flow analysis that were gathered with the questionnaire. The colour of the cell (rectangle) indicates the intensity of interaction between the two stakeholders (black=daily; gray=weekly; white=more infrequent interaction). The text in the cell shows the content of the interaction. This result indicates that there is intensive interaction within a project but project-crossing interaction takes place more infrequently and via specific participants. It also points out topics that are discussed with the project external representatives are not strictly specified.

5.3.1.3 Results from Contextual Design

The basic structure for the information sharing system for user-centred information was created by following the Contextual Design process. The results from that process are described here.

During the Contextual Design phase, the CD development group created design artefacts ("work products") that are mentioned in table 4. Figure 17 shows the arrangement of two artefacts (12/Storyboards and 9/Vision) during development.

Table 4. The design artefacts created during Contextual Design phase.

1. Contextual Inquiry Memos and tapes	9. Visions
2. Affinity Diagrams	10. Consolidated Vision
3. Affinity Insights	11. Concrete Vision
4. Affinity Design Ideas	12. Storyboards
5. Flow Diagram	13. User Environment
6. Flow Insights	14. Paper Prototype
7. Flow Design Ideas	15. Shipping User Environment
8. Consolidated Flow	

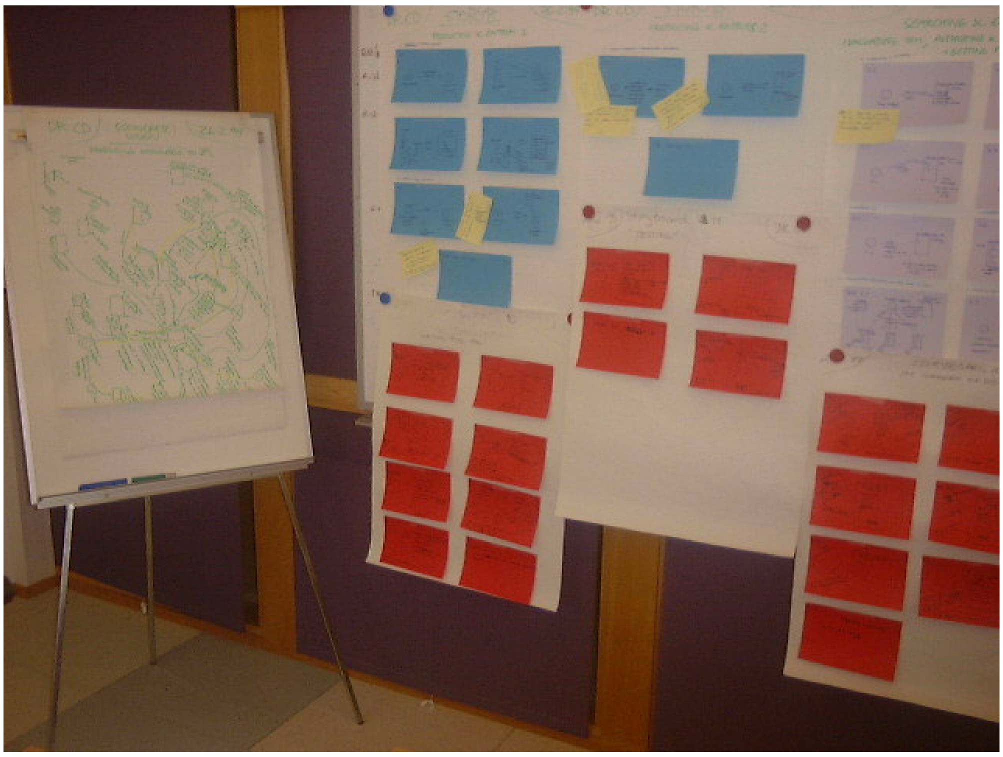


Figure 17. Intermediate design artefacts that were produced during Contextual Design. On the wall are task scenarios (storyboards; artefact number 12) and on the board is the final "concrete" vision (artefact number 9).

Artefacts 1-8 provide valuable information for the rest of the CD process and for the creation of the rest of the artefacts. They can be characterised as “artefacts for use internally in the CD process”. Artefacts 9-15 are deliverables that seemed to provide most interesting results to people that have not participated in the development work as they provided summarised and already analysed information for the users. Should someone, however, wish to acquaint himself or herself with detailed background information about the outcome of the process, artefacts 1-8 provide good possibilities for it. The Contextual Design process is constructed so that traces to originating information will remain known as subsequent artefacts are created. In order to provide easy access to this information, the Knowledge Storage must be able to provide links to all these artefacts.

As the focus of the CD development work itself was in the design of the support system for the user-centred development work, the most interesting results from the standpoint of the implementation of the Knowledge Storage appeared to be following artefacts:

- vision (explains the overall idea and ultimate goal of the system)
- user environment model (defines the structure of the system from the users’ standpoint)
- storyboards (provide early validation possibilities to evaluate the utility of the system and one important communication model for developers’ use)

The memos and tapes provided all basic data that was used to construct the artefacts mentioned above. Affinity diagrams and flow diagrams were very important from the standpoint of the internal work of the development group. They can also be used for intermediate reviewing in the middle of the CD work. After the CD work is completed, the most important “communication” artefacts appear, however, to be vision, user environment model, and storyboards (with supporting paper prototypes). These are described in more detail in the following sections.

5.3.1.3.1 Affinity Diagram

Even though affinity diagram is mainly of internal interest during the Contextual Design work, is consists of very important “raw material” that provides viewpoints to the resulting application. The construction pieces of affinity diagram are the findings (short statements from each user interview i.e. contextual inquiry) from the sharing session created by the development group.

Affinity diagram is a four-level tree-structure presentation that is used to create groups of related issues that provides a view to the activities that the solution should support. Affinity diagram is composed of findings that emerge in the sharing sessions that follow the interviews (contextual inquiry). The raw findings constitute the lowest level statements. They are grouped into the higher-level categories by the development group. These groupings are, therefore, already analysis of the raw material.

The top level issues in this work describing the activities in a distributed software organisation were as presented in the table 5.

Table 5. Top level issues in affinity diagram describing activities in a software development company operating in a distributed/networked way.

1. We develop our products (I and II) 2. We collect information 3. We need to share info among ourselves 4. Technology and product “mature” 5. We market and sell our products 6. We have marketing material	7. Case customers / customer projects 8. We want to know how customers like our products 9. We support customers 10. We still need to improve our processes

From the standpoint of “supporting the sharing of information about users” (that is the goal of the work), topics 2, 3, 7, 8, and 9 are the most important ones. However, information under other topics provides valuable constraints for the specification and implementation work.

Excerpt from the category “We need to share info among ourselves” (see table 6) clarify structure and contents of the affinity diagram. This topic also clarifies the need for information sharing in the development environment. This topic contained issues presented in the next table.

Table 6. Excerpt from the category “We need to share info among ourselves” from the affinity diagram.

3.1	We are an international company
3.1.1	We work differently in different cultures
3.1.2	We have divided the development to two countries
3.1.3	We have language related problems
3.2	We share information via electronic systems
3.2.1	We share information about bugs via bug reporting system
3.2.2	We share explicit information within the company via databases
3.2.3	We use Email a lot
3.2.4	Support calls are forwarded to development if necessary, and they send solutions back to support
3.3	We communicate in many ways
3.3.1	We share information informally
3.3.2	We give advice to each other
3.3.3	We have meetings
3.3.4	Expert seminars are an important forum to share information between partners and us
3.4	We do not always use electronic systems
3.4.1	The use of bug tracking system is inconsistent
3.4.2	Even if we have the bug tracking system, we use face-to-face communication

As an example of the raw findings (affinity notes) from the interviews and the sharing sessions, the statements under “3.3.1 We share information informally” are presented in table 7.

Table 7. Excerpts from the source data about topic “We share information informally” from the affinity notes.

518. U5.	Q: What is a ”social group of developers” like?
265 U2.	Nothing wins the hallway and coffee table discussions.
625. U6.	In the local office there is no formal meeting practice between development and sales, the discussion takes place in daily basis. (Problem)
175 U1.	In technology companies, people tend to forget open human communication. In small group of 4-5 persons, even tacit knowledge is spread out effectively to all participants. You should not over-organize everything.
425 U4.	Information for solution documents is in bits and pieces around the organization, not in any documented form. (Problem)
264 U2.	I used internal newsgroups a couple of years ago a lot but I don’t use it anymore. There are no interesting discussions and not enough active participants. It is not global anymore. Earlier it was very important tool. (Problem)
263 U2.	Forum is the only way to spread unsolicited detailed development related information, “without knowing the target person”. Electronic mail does not work in these kinds of situations. (Problem)

The structure of the *affinity notes* that is visible in the previous excerpt, consists of the identification number of the affinity note (e.g. 625) and the source of the affinity note (e.g. U2 = user 2). In the statement itself, there are three different types of statements: direct findings from the user interviews, questions (Q:) raised by the development group, and problems (P: or "Problem") in the current working practices that are either pointed out by the interviewed persons or by the development group.

In the development of Knowledge Storage, the total amount of *affinity notes* that were grouped under the ten categories was 497. Additionally, the amount of statements that were not put into any of these categories (considered as "junk") was 11. Altogether, the interviews of seven people (U1-U7; expected representative "users") produced 508 qualitative statements for the development of the knowledge storage (U1:84, U2:97, U3:102, U4:75, U5:41, U6:42, U7:67).

The affinity diagram provided a working artefact for the "affinity walk" (a session in which the whole development group and appropriate external persons also look through the affinity diagram) that was used for creating insights, questions, and design ideas (DI) about the findings. This part of CD is in a way creating a funnel that starts to restrict the possible solutions. In this case, the results that the affinity walk produced related to information sharing are presented in table 8.

Table 8. Information sharing related results from the affinity walk.

Affinity issues

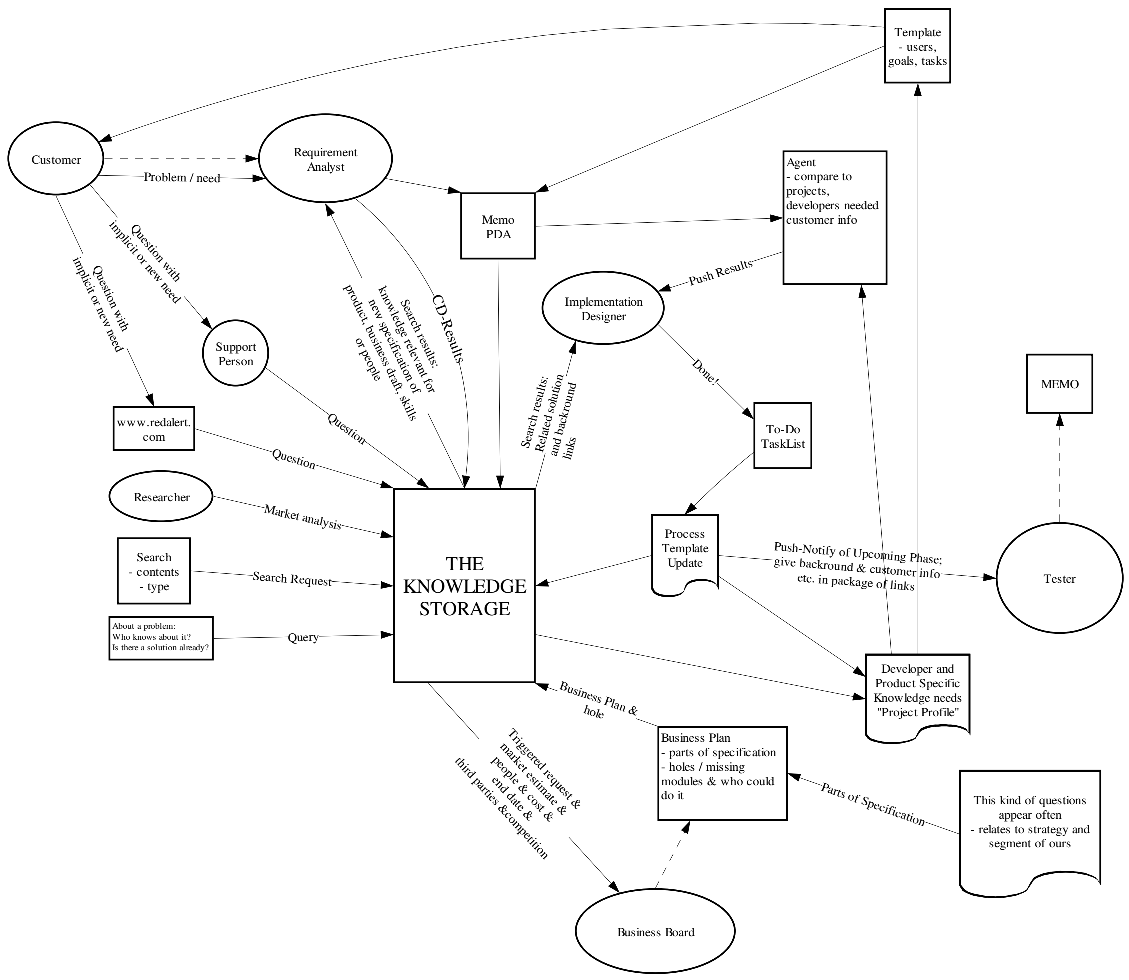
- System software development difficult, do not create 'sectors' / excess structure which makes things even more difficult
- We do not use our products enough ourselves, for example process products. How well people know those that are used?
- Even large software development organisations are organised as a bunch of garage shops
- Different projects/teams/persons collect customer info in their own ways
- People have perceptions of 'other parts' of the company doing things they do not do or not doing things they actually do
- There are databases and info but do they reach all who should know
- It would be important to know what is known and where
- Active interaction between related roles is/would be beneficial
- Fostering experience that makes for efficiency is important
- TA: Investigate why some development groups use for example electronic bulletin board systems and others do not. Use this information in visioning new process / tool.
- Information overflow acute, people do not need a further input line. People will use a new one if they truly feel that they benefit from it.

Design Ideas

- DI: develop more support for role-based communication in-house
- DI: In the earliest phases of new product development the driver if either technology or customer -> involve sales more in design
- DI: Organize / structure product related documentation to be accessible through one interface which covers the whole development process and different functions. Major docs 'topmost'.

5.3.1.3.2 Vision

During the CD process applied in this case, three different visions were created. First, a high-fly vision without technical or operational restrictions was created. It aimed at capturing all problems that has been detected in the interview sessions (see figure 18).



Template
- users,
goals, tasks
Customer
Requirement Analyst
Problem / need
Memo PDA
Agent
- compare to projects,
developers needed
customer info
Push Results
Implementation Designer
Question with implicit or new need
Question with implicit or new need
Support Person
Search results: knowledge relevant for new specification of product, business draft, skills or people
CD-Results
Search results: Related solution and backround links
Done!
MEMO
To-Do TaskList
www.redalert.com
Question
Question
Researcher
Market analysis
THE KNOWLEDGE STORAGE
Search
- contents
- type
Search Request
About a problem:
Who knows about it?
Is there a solution already?
Query
Process Template Update
Push-Notify of Upcoming Phase; give backround & customer info etc. in package of links
Tester
Developer and Product Specific Knowledge needs "Project Profile"
Business Plan & hole
Business Plan
- parts of specification
- holes / missing modules & who could do it
Parts of Specification
This kind of questions appear often
- relates to strategy and segment of ours
Triggered request & market estimate & people & cost & end date & third parties &competition
Business Board

Figure 18. The information support environment ("Knowledge storage") in the centre of development information. This figure is the result of the "hi-fly" long-term vision that was drawn first in the visioning session. This vision presents possibilities towards future as not all parts are at all possible to realise and implement with current technology.

The system that was outlined in this vision was capable of detecting almost all information needs and wishes of development stakeholders and providing all relevant information as soon as that information entered the system. The system of this vision is aware of all developers' expertise and it is capable of putting together a project group; Additionally, it was capable of digging out all required basic components and putting them together for initial specifications and implementation. This vision, without a doubt, was beyond the implementation possibilities during the study so two additional, downgraded, scenarios were required. Some elements from this scenario, however, remained in the implementation the most important being the central knowledge storage. The most important idea in this vision was the possibility to utilise the information sharing system for directing information to relevant stakeholders as soon as information is available. This requires the practitioners be presented in the system with some kinds of interest profiles according to some easily understandable characteristics derived from the context (working environment).

The concrete vision that serves as the basis for the functionality of the Knowledge Storage is, for clarification reasons, presented together with the use scenarios.

5.3.1.3.3 The User Environment Model

The user environment model (UE) describes the structure of the system under development from the users' standpoint. It provides a rather low-level description of the visible functions in the system. The expected user interface should follow the structure of the UE in order to meet the requirements detected during Contextual Design.

The user environment model of Knowledge Storage consisted of 20 parts (see figure 19 and appendix C for detailed information about the contents of the figure). The creation of the model was conducted with the help of the storyboards that described users' actions with the system but in the context of their tasks.

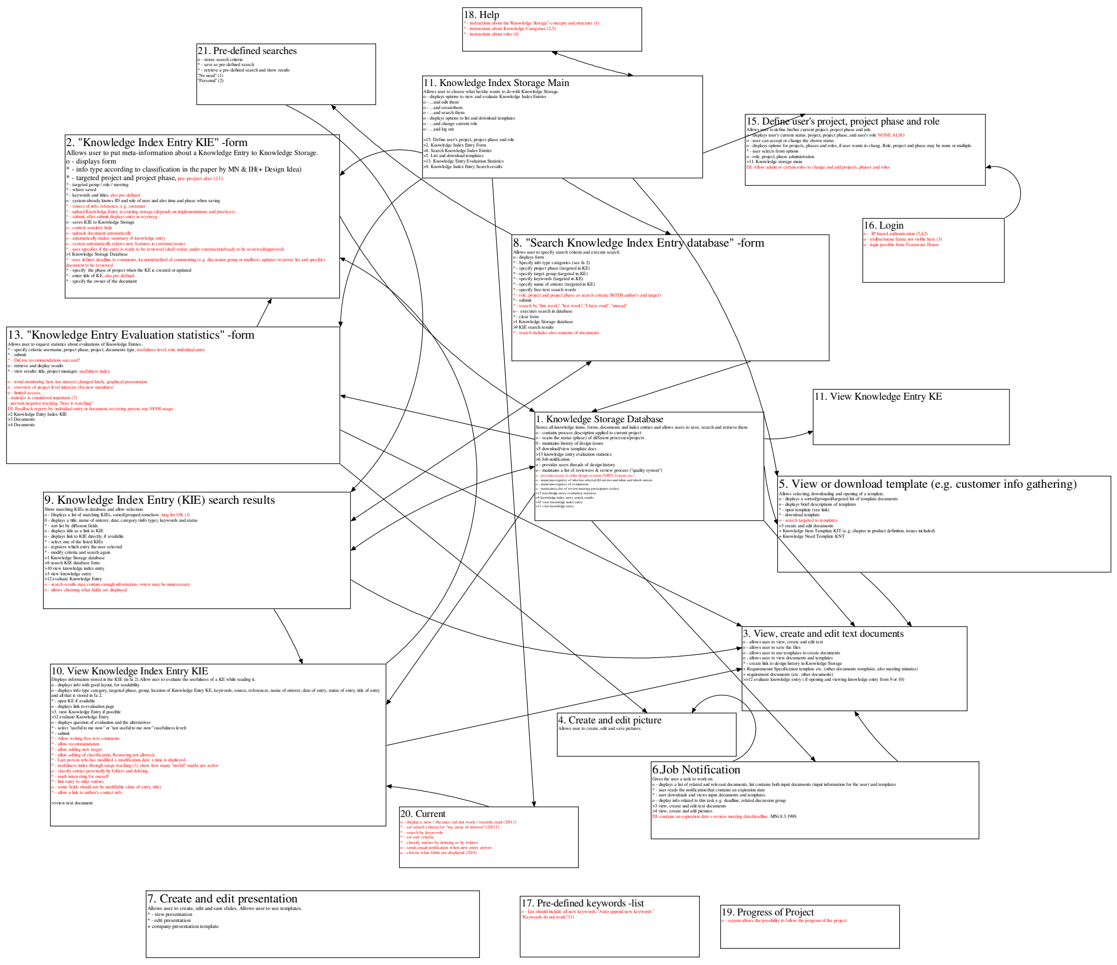

18. Help
21. Pre-defined searches
11. Knowledge Index Storage Main
15. Define user's project, project phase and role
2. "Knowledge Index Entry KIE" -form
16. Login
8. "Search Knowledge Index Entry database" -form
13. "Knowledge Entry Evaluation statistics" -form
11. View Knowledge Entry KE
1. Knowledge Storage Database
5. View or download template (e.g. customer info gathering)
9. Knowledge Index Entry (KIE) search results
3. View, create and edit text documents
10. View Knowledge Index Entry KIE
4. Create and edit picture
6.Job Notification
20. Current
7. Create and edit presentation
17. Pre-defined keywords -list
19. Progress of Project

Figure 19. The user environment (UE) model that defines the basic structure in the resulting Knowledge Storage system together with its user interface. The contents of the model are described in more readable form in Appendix C.

5.3.1.3.4 Concrete Vision and Use Scenarios: Accessing User Centred Design Artefacts in the Knowledge Storage

When Knowledge Storage was developed, main focus was in the support for the management (creation, storing, retrieval) of the artefacts created during a Contextual Design process that aims at capturing the activities of the user in an unfiltered form. With the support of Knowledge Storage, product development practitioners can follow user centred development processes and have detailed method descriptions of corresponding activities. The *templates* (sort of quality documents) in the Knowledge Storage provide support for this.

Use scenarios for gathering information with the used-centred development methodology (in this case “Contextual Design”) and for storing, communicating and using the models created with it. The scenarios were aligned with the creation of the concrete vision (see figure 20).

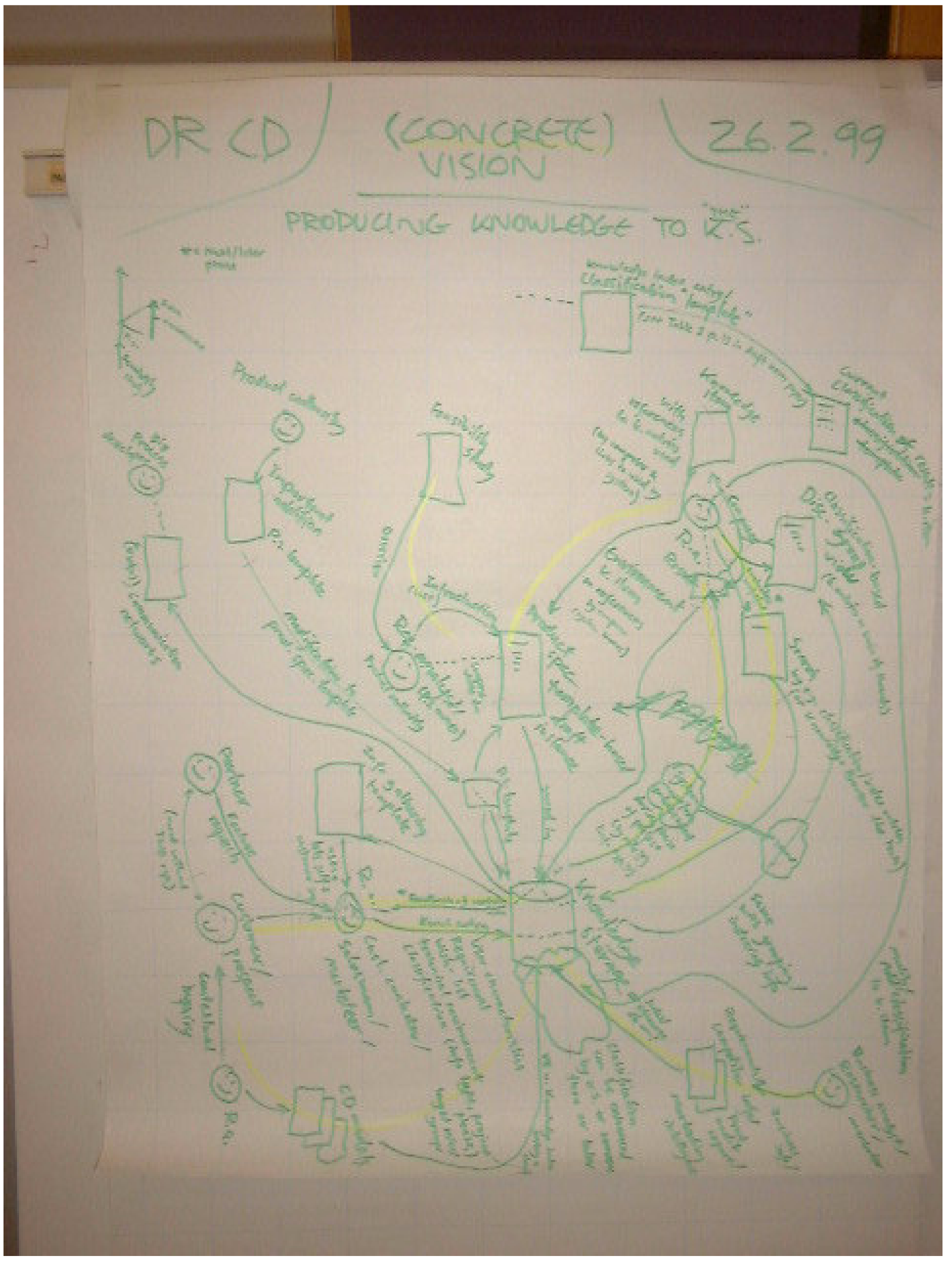
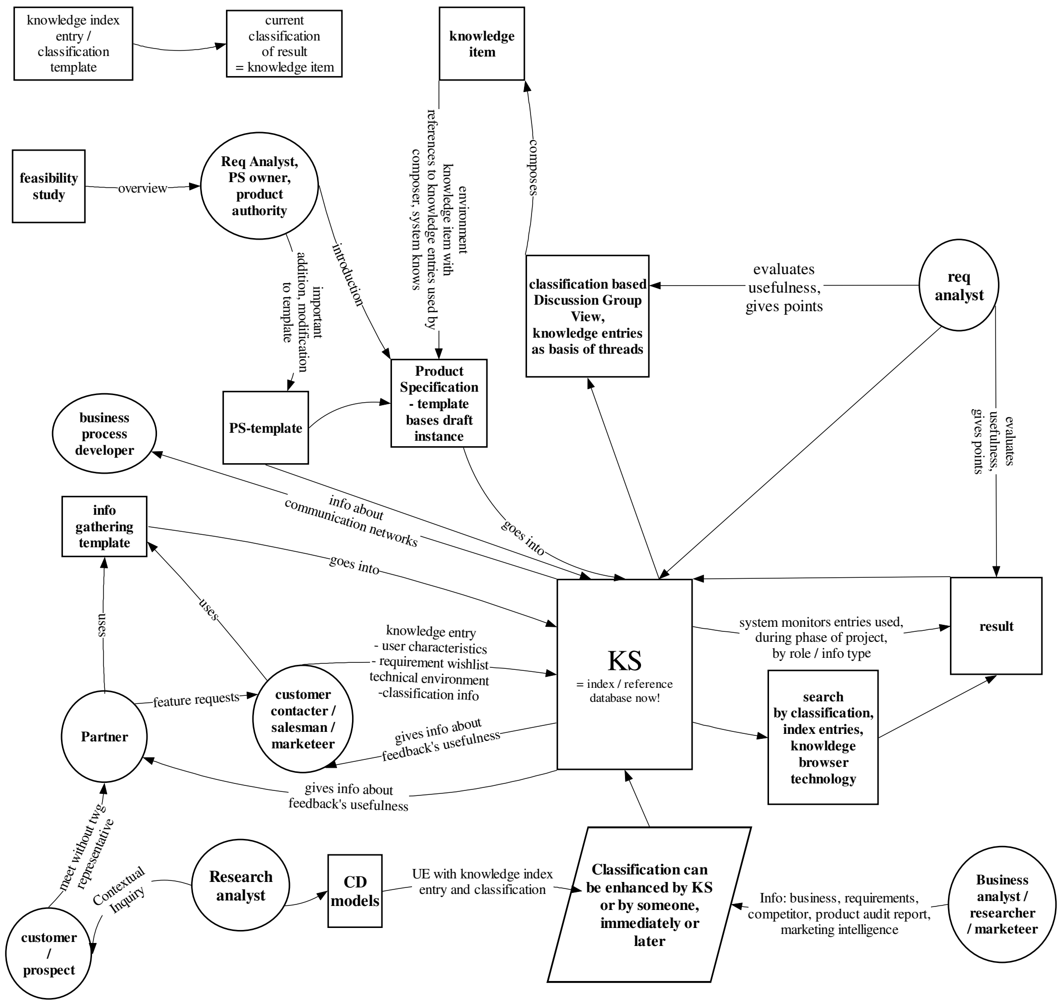


Figure 20. The concrete vision drawn on the board. The use scenarios presented in the text reflect this model.



knowledge index entry / classification template
current classification of result = knowledge item
knowledge item
feasibility study
overview
Req Analyst, PS owner, product authority
environment knowledge item with references to knowledge entries used by composer, system knows
composes
important addition, modification to template
introduction
classification based Discussion Group View, knowledge entries as basis of threads
evaluates usefulness, gives points
req analyst
Product Specification - template bases draft instance
PS-template
business process developer
evaluates usefulness, gives points
info gathering template
info about communication networks
goes into
goes into
uses
uses
KS
= index / reference database now!
result
system monitors entries used, during phase of project, by role / info type
knowledge entry - user characteristics - requirement wishlist technical environment -classification info
feature requests
Partner
customer contacter / salesman / marketeer
gives info about feedback's usefulness
search by classification, index entries, knowlidege browser technology
gives info about feedback's usefulness
meet without twg representative
Contextual Inquiry
Research analyst
CD models
UE with knowledge index entry and classification
Classification can be enhanced by KS or by someone, immediately or later
Info: business, requirements, competitor, product audit report, marketing intelligence
Business analyst / researcher / marketeer
customer / prospect

Figure 21. Part of the concrete vision drawn on the board describing the central elements that supported the creation of the use scenarios.

Storing user centred development artefacts into Knowledge Storage. A requirement analyst (e.g. a system designer) visits a customer site and meets end-users. The requirement analyst conducts a Contextual Inquiry and records all important actions and activities (according to the predefined focus of the observation) that the interviewee performs during the two-hour observation. After having conducted the interview, the requirement analyst shares his/hers knowledge with the other participants of the Contextual Design session. The models based on this information will be then created and documented on sheets on the walls.

After having completed Contextual Design, the requirement analyst saves the models on the walls into electronic form. The models will be saved using different development information systems like a word processor and a flowchart program. The resulting files are saved on a network server.

At the same time when the requirement analyst saves the files, he/she creates an index entry into the Knowledge Storage by filling out a form that is used to add a new *knowledge entry*. The location of the newly created file will be stored in the knowledge entry together with information about the requirement analyst's current role, project and project phase. The requirement analyst can also target the document to other roles that may find the information useful.

Searching and using user centred knowledge entries. When the requirement analyst needs information about users and their tasks and characteristics, he enters the Knowledge Storage and logs in with a username and password and provides the system with current role, project and project phase. Then the requirement analyst fills in the form used for initiating the search. The form contains possibility to search for any classifying information including the role, project and phase of the author and target role. Knowledge categories are used as search criteria as well. If the structured search capabilities are not sufficient, the user may turn to free text search. To search for Contextual Design artefacts, the user can apply a filter with "CD" in the document title if all Contextual Design artefacts have been stored with that mark.

Viewing knowledge entries. After having submitted the query, Knowledge Storage displays the matching knowledge entries. The requirement analyst can now view any entry by clicking its title. Knowledge Storage records this action. This way the system remembers what entries the user has used. Additionally, this information can be used for evaluation purposes and to create networks of knowledge entries.

The knowledge entry contains a browser launchable link to the document that contains the actual information. The information may also be visible already in the knowledge entry itself.

Evaluating knowledge entries. After having read the entry or the document, the user can evaluate whether the entry was useful for him/her at the moment. This statement of interest will be recorded in the Knowledge Storage and it can be used as and indicator of important information items.

This evaluation was considered to be very interesting and important during the paper prototype presentations. Four members of a development organization were interviewed in paper prototype simulations. The results from using the paper prototypes showed that the interviewees found the possibility to give and get contribution evaluations an important feature in the design. The main benefits expressed by the interviewees included knowing if one's contributions are appreciated and used by others, awareness of type and nature of contributions found useful by others, increase in motivation to make contributions and improvement of the quality of information shared in the organisation.[51]

Two types of "contribution evaluation" were discovered: quantitative and qualitative. We implemented only the "quantitative contribution evaluation". Quantitative evaluation is based on "hit count" to an individual knowledge entry (a small information item created by an individual developer). Hit count is calculated based on two statistics: the amount of times that a particular entry is viewed on screen and the amount of explicit "This information is useful for me now" evaluations. Quantitative evaluation gets partly created automatically: the system keeps a log of all entries that have been shown on individual practitioners' screen. The evaluation of "This information is useful for me now" requires that the user clicks the evaluation link that is attached to each knowledge entry.

Digital storyboards. The user scenarios were documented in our project as storyboards. Each storyboard consisted of a few simple hand-drawn pictures of all activities described in the scenario. The pictures were recorded by photographing them with a digital camera (see figure 21). The digital pictures were saved as files and stored in a form that is accessible by a web browser. Thus, it is possible to access the storyboards straight through the Knowledge Storage.

[51] *Social navigation* is a promising field to study the social effects in the emerging information spaces. Research in this field has shown how direct (real-time collaboration) and indirect interaction with other users can affect how users navigate and use information space systems (Munro & al. 1999). Indirect interaction is mediated by the "traces" of other users' actions in the information space. Annotations, recommendations, hit counts and other "footprints" left by others guide users to the sources of information they are looking for. Contribution evaluations can be seen to be an example of indirect interaction in an information space.

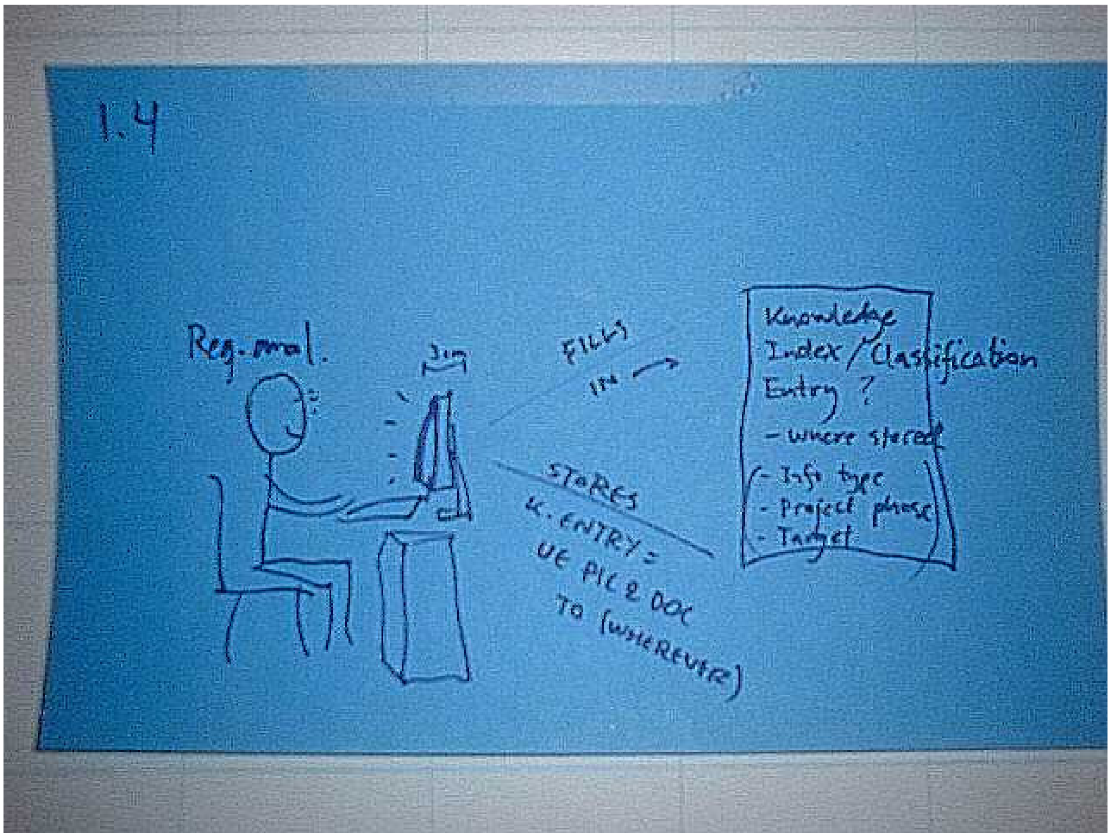

1.4
Req. anal.
Jim
FILLS IN
STORES K. ENTRY = UE PIC & DOC TO (WHEREVER)
Knowledge Index / Classification Entry ?
- where stored
- Info type
- Project phase
- Target

Figure 22. A drawing from a series of storyboards created during the development of KS. In this picture the requirement analyst stores the user centred design documents to Knowledge Storage.

5.3.2 Knowledge Storage Prototype: Model and Implementation

Even though Contextual Design was capable of providing in-depth information about the “Context of Use” for the creation of requirement specification of the Knowledge Storage, it provided little means for the actual implementation of the prototype. The creation of functioning prototypes (other than paper prototypes) falls mainly outside the Contextual Design process. In our case, we wanted to go beyond the specification of the user environment (UE) description for piloting and evaluation purposes. This led us to implement and test the Knowledge Storage model with a prototype.

Several constraining criteria applied to the implementation and piloting with the prototype:

- In order to gain real results from the pilot use, a prototype needs to be really functioning for piloting, not for demonstration purposes only (perhaps not for heavy 'production use' but for a limited amount of project staff still)
- In order to support distributed working in separate offices, the prototype needs to be available in the current networking infrastructure of the pilot organisation (preferably accessible via web browser; this indicates ip-networking, NetBIOS is another possibility, though)
- The system should run in "standard computer environment" in the pilot organisation (no extra support personnel was available; the expected support and maintenance tasks must be conducted by the research group itself)
- The creation and modification of the system should be rather fast (no extensive effort to the implementation is available)
- Information that will be managed with the system should be transferable to other systems after piloting if the use of the prototype discontinues

5.3.3 The Knowledge Storage Concept

5.3.3.1.1 Elements of the Knowledge Storage

Information and knowledge in product development deals very much about technology and product features. The creation of a commercially viable product also points out the need for market and customer as well as user-related information. These data form the core of the information space in product development.

Developers need information about the underlying assumptions behind the product features and about the systematic working practices to produce all the information that is used to define and implement the product. Much of this information is provided by the quality documentation that provides guidelines for development processes and *templates* of the artefacts that need to be created in various phases of a project. Templates are skeletons of documents. Documentation is a key element in development work even though it is often considered as something that is not as interesting as the actual innovation work.

When the creation of the structure for the Knowledge Storage began, it became soon evident that the traditional document based (document level) classification and storing mechanism[52] is not a feasible enough approach. It is not easy to put to an entire document into one *knowledge category* because a document contains sections that address different topics. For instance, a project specification may contain information about markets, development resources, available technology, intended target customers and users. All this information is important but different people (*roles*) need different information and wish primarily to contribute and get information about issues relevant to themselves. In the development of Knowledge Storage, we needed to shift towards *information item* based information management.

Information item is a small package of information (like a statement "users are willing to pay at most $100 of a product like this") that contributes to a specific knowledge category. Information item is not, however, restricted to be just a small statement, it may be a whole document describing various issues[53]. A *knowledge category* is a collection of information items within a specific topic (like "business issues" or "user information"). As information accumulates to a knowledge category, it forms related information that grouped together about that topic. When these kinds of pieces of information become related to each other in a way that is meaningful for humans, knowledge is created. In a way, *affinity notes* in Contextual Design constitute one type of information item that may be classified as a *user-centred information item*[54].

During development work, the amount and quality of information about a specific topic may act as exit criteria instead of time/deadline. It may be defined in the development quality system that "as soon as we know enough about this specific pre-defined important topic, we can proceed to the next stage of the project". This approach could be characterised as "management by information", or "management by knowledge".

[52] With "traditional" we mean that a document is handled as one unity stored to one thread in a tree hierarchy on a local or shared network disk. Access permissions are usually restricted and only one person is allowed to edit a document at a time. This approach points out that just changing the document storage platform e.g. from shared disks to web-browser accessible document repositories is not enough. Additional consideration about the activities related to the management of the information within the documents needs to be designed.

[53] We noticed, however, that managing whole documents with Knowledge Storage may not be efficient, because many of the documents in development address a broad range of important issues. A better way to handle this would be a splitdown of the content of the documents into smaller pieces that address issues falling into one *knowledge category* only.

According to the interviewed developers, information overflow is evident in almost all current development tasks. It is not too uncommon that a developer has hundreds of e-mail messages that are somehow "on hold" in their inboxes. This indicates that adding items on this information channel will not take things too much forward. Despite this, the developers need right information at the right time. In product development work this means that as the project proceeds from one phase to another, appropriate information about the process and the product under development should be available. Not everything is important throughout a project. Information should be targeted to developers according to the phase of the project. Not all information should be visible to the developers at all stages of a project. The problem in this approach, however, lays in the tagging of information to be of importance at a specific stage of the project: Each information item should have describing meta-information that contains information about appropriates uses of the information as well including the stage of the project in which the information item is needed (or at least considered useful/important).

Additional restriction relates to possibilities to shift from one information system platform to another. To create practical knowledge categorisation in real development environments, we need to retain compatibility with existing documentation and quality systems: We must include information that currently exists in product development documentation. This information forms the base for the categorisation of product information (*knowledge categories*). Initially, the knowledge categories exist in the headings and subheadings of the development documents that describe the product and the project. In our research project, we conducted a qualitative analysis about the structure and contents of these documents to support the piloting of the prototype tool.

[54] An example of an *affinity note* originated *user-centred information item* could be: "618. U6: I use Palm Pilot and during customer meetings I record ideas, sales points, sales blocks etc. as text documents, and sometimes I send the memo to technical manager, our manager and project manager, and developers."

5.3.4 Structure of the Knowledge Storage

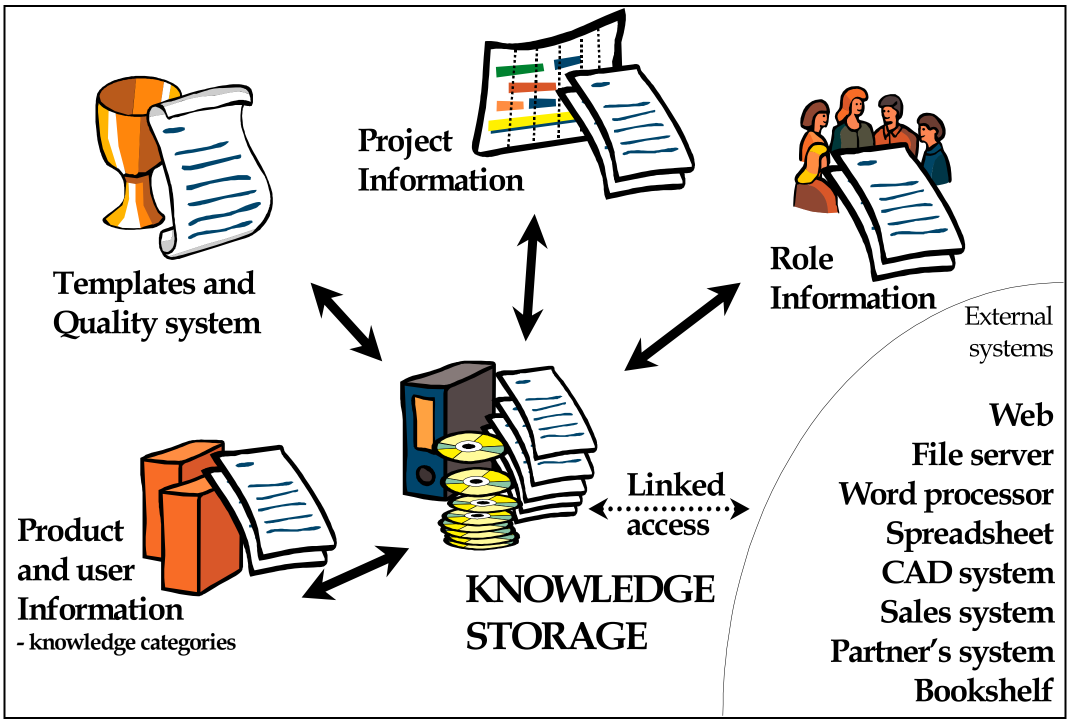


Figure 23. The Knowledge Storage contains information about roles (stakeholders of a development project), project and the product under development containing information about users and usability. In addition to these, it contains various templates (e.g. quality documents, specification templates, and memorandum skeletons) that can be used throughout the development project. The knowledge storage itself contains links with classifying information to various documents that have been created with external development information systems.

Knowledge Storage consists of four basic elements that can be characterised to be either administrative (roles, projects, templates) or substance (product) (see figure 23):

Roles. A single individual seldom conducts development work alone. There are several stakeholders and groups that contribute directly or indirectly to the product. A development group consists of a project manager, a product manager, developers (designer and implementer), human factors specialists, graphic designers, localisation experts, marketing staff, sales people, and general management. These people are in contact with customers, end-users, sales partners, government representatives and so on. Each stakeholder representative brings in important information and communicates it to others in a suitable form.

The concept '*role*' aims at capturing the different stakeholder groups. An individual participant of a project may act in different roles during a project. Therefore, there needs to be a possibility to change the role on the fly in any stage of the project. In order to improve the communication mechanisms in a development project, we need to identify all relevant stakeholders and their corresponding roles and make them visible to each other. In addition to changing the role within a project, the participants may work simultaneously in several projects. This means that the participants may bring in information from other projects as well.

Project information. Product development is conducted in projects. Projects are divided into phases and a project manager follows the progress of the phases of the project. Projects have certain administrative characteristics like a project plan, a project manager, schedule, milestones and deadline, resources (in terms of money and man-months), and project structure i.e. phases of the project. Developers and designers are assigned to a project and gather information that is useful for the project and the product from the perspective of their current role.

Depending on the stage of the project, the participants gather different types of information: In the beginning, information about the feasibility of the concept is being gathered. During detail design and implementation, important topics relates to technology and possible detailed level solutions. When the testing of the product is in progress, it is important to gather information about lapse and typical construction related problems. A great amount of information contributes mainly to a specific phase in a project, not throughout all phases. However, important information for a specific project phase might come up already in the earlier phases when it is not yet relevant.

Product information. Product information is the core of the Knowledge Storage. Almost all activities in a development project aim at gathering information that can act as constructive parts for the product. The quality of the product is heavily dependent on the coverage and completeness of the information that has been used to construct it.

Product information is divided into *knowledge categories*. This classification can be created per project or per organisation basis. Critical topics can be defined explicitly. Knowledge categories can contain information about markets, economic business issues, customers and users, available and applicable technology, application-specific constraints, product features, and product portfolio. All information that goes into the Knowledge Storage will belong to at least one knowledge category.

The principle “right information to right people at the right time” is implemented in the Knowledge Storage concept through *targeting*. All knowledge entries are targeted to specific receivers. Targeting is done by using role, project, and project phase information. Whenever a user creates a knowledge entry, he/she can target that information to a specific role in a specific project so that the information will appear at a specific phase of the project to that role. People will face the information as the project proceeds but they will not be overwhelmed with all the information coming to them otherwise in a wrong time.

As the project proceeds, a product manager or a project manager can determine when the development team has gathered enough information about a specific knowledge category. This means that a development project can even be managed by the information that it has produced. The quality and amount of information can be used as exit criteria from one project phase to another.

Templates. Most companies have a quality system that contains instructions about important activities that aim at ensuring adequate level of quality. Activities are performed in various phases of a project and they need to be supported with instructions that are detailed enough.

The *Templates* section of Knowledge Storage consists mainly of quality system instructions and document templates that guide developers to work in a predefined and systematic way. Templates can be files for various office and development systems as well as links to quality instructions in the company Intranet.

Templates can be e.g. skeletons of a project plan, templates for product specification, templates for test procedures, questionnaires for initial customer contact sessions, templates for reporting a product defect, in-house style guides for user interface design or check-lists for addressing user centred issues in different phases of a project. Even though these documents may reside in separate places in the organisation (like any other document or piece of information accessed through Knowledge Storage), users of Knowledge Storage can access them through one single interface.

Traces. All information in the Knowledge Storage will be linked to each other. As people read, browse or create information through the system, it will be monitored according to the role, project, and project phase the user currently has. This way system will construct a network of information all the time according to the decisions that the users make. The system keeps a log of threads that can be used to trace information later, perhaps for purposes like social navigation.

External links. Since we do not have the possibilities to include all functionality of the other development information systems within the Knowledge Storage, we must provide links to external systems and their files. This way we can index almost all information that is being dealt with in a development project. The Knowledge Storage acts as a front-end to these external systems.

The structure of the Knowledge Storage can be summarised with a representation presented in table 9.

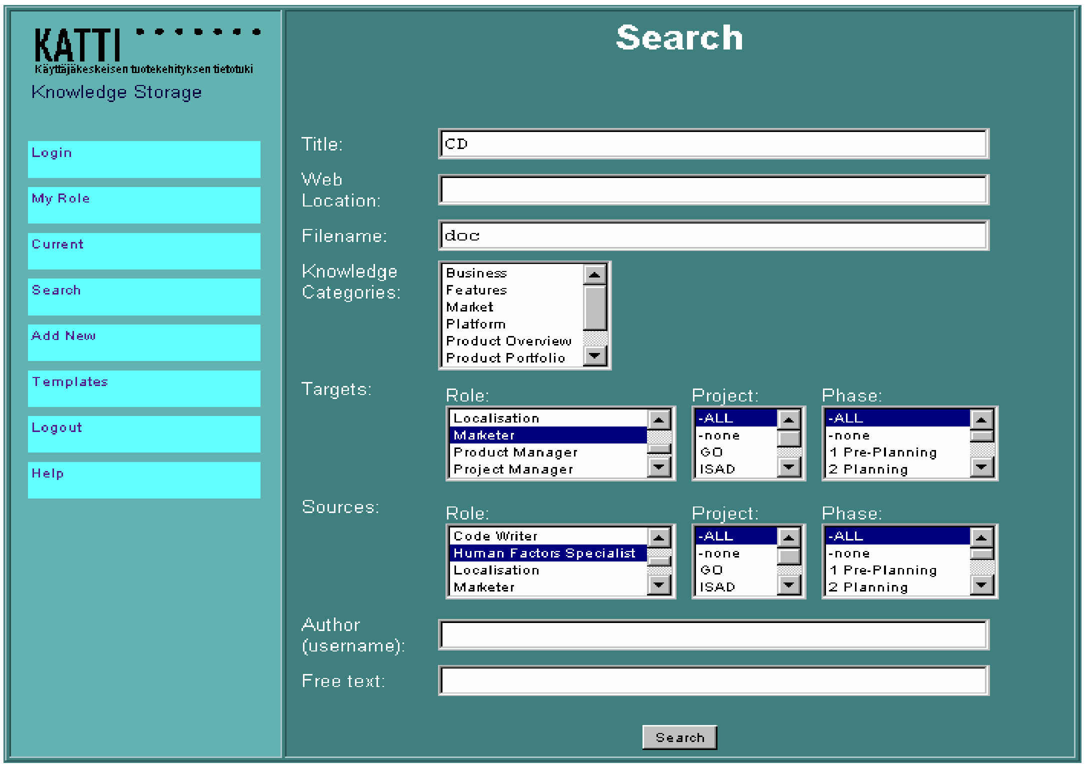
Table 9. The elements of Knowledge Storage with expected uses.

template_element	this element provides developers the important issues and topics that need to be considered during development work (related to quality systems / quality documentation)
item_element	this element defines the issue/data (knowledge item) itself; it is either text or a link to an external file or service (e.g. web link)
contextual_element (metadata to the item_element)	this element captures the user's (developers) context (providing easier access to it later) and consists of following items
temporal_element	this element keeps track of following things
new_information	information that has been added recently and is targeted to the role that the user has
reviewed_information	information that the user has already reviewed
old_information	information that is relevant (for design rationale) or obsolete ("forgotten"; no further utilisation expected; e.g. nobody has reviewed this information)
social_element	this element provides information about who has created, read, or reviewed information. It may be a *trusted person*, a *person with a similar role*, or *a colleague* working e.g. in the same project or development group
process_element	this element points out how to use the information in the knowledge item (item_element); this may be derived from the template_element

5.4 Implementation of the Prototype

5.4.1 User Interface of the Knowledge Storage Prototype

To test the concept of the Knowledge Storage in real settings, we created a prototype tool (see figure 24) that implements the concept. The system runs attached to a web server and is accessible with a web browser.



KATTI
Käyttäjäkeskeisen tuotekehityksen tietotuki
Knowledge Storage

- Login
- My Role
- Current
- Search
- Add New
- Templates
- Logout
- Help

Search

Title: CD
Web Location:
Filename: doc
Knowledge Categories: Business / Features / Market / Platform / Product Overview / Product Portfolio

Targets:
- Role: Localisation / **Marketer** / Product Manager / Project Manager
- Project: **-ALL** / -none / GO / ISAD
- Phase: **-ALL** / -none / 1 Pre-Planning / 2 Planning

Sources:
- Role: Code Writer / **Human Factors Specialist** / Localisation / Marketer
- Project: **-ALL** / -none / GO / ISAD
- Phase: **-ALL** / -none / 1 Pre-Planning / 2 Planning

Author (username):
Free text:

Search

Figure 24. Sample screen shot from "Search" screen of the Knowledge Storage prototype tool. In the presented situation the user is searching for Contextual Design (CD) documents with a “doc” in the filename that has been submitted by a human factors specialist and targeted for marketer’s use.

The user interface of the system consists of two major sections: the toolbar showing the functionality of the system is on the left “frame” (it is actually a cell of a table, not a “browser frame”; the actual HTML frames have been found to be difficult for users to manage in some cases) and the data area is on the right.

The top-level user interface contains eight functions (the screen shots about the user interface are presented together with the technical presentation later in section "End User Functions"):

- **Login** screen is used to enter username and password. Username is used to identify knowledge entries that the user has already read and entries that still remain unread.
- **My Role** screen is used to enter user's current role, project and project phase. User is allowed to change all these at any time when using the system. It is even possible to change the current role, project and phase when submitting an individual entry.
- **Current** screen shows all new entries that have been submitted and targeted to the role, project and phase that the user has currently selected. The Current screen is divided into two sections: new entries and recently used entries. During the development of Knowledge Storage it was found out that developers were constantly searching for documents and information that they had been reading recently.
- **Search** screen is used to search for knowledge entries. The user is allowed to apply criteria for any of the classification fields that are being used when submitting entries into the system. After having pushed the "submit button" in the search screen, the system displays a list of matching entries. If the user wants to make a new search, he/she can select the Search screen from the toolbar on the left again. From the list of matching entries, the user can select an interesting entry by clicking its title. Then the system shows that particular entry. We expect that the users of Knowledge Storage do not only rely on the search capabilities of the system itself. In addition to the internal search capabilities, there are tools available ("Knowledge Browser") that run attached to a browser and index all information that has been reviewed in the browser.
- **Add New** screen is used to add new entries into Knowledge Storage. The user is presented with a form that he/she needs to fill in. The user can classify the information by the knowledge category, targeted role, project and project phase. In addition to this, the document will be automatically marked with information about user's own role, project and project phase. This information can be used as search criteria.
- **Templates** are document skeletons or instructions about conducting a specific design activity. Templates may contain parts of the quality system of the company.
- **Logout**. When the user ends the session with Knowledge Storage, he can logout from the site to clear all application specific cookies etc.

- **Help**. If users experience problems in using the Knowledge Storage they can look for help from the help screen. According to Nielsen's (1993) heuristic rules, every system must have help available.

5.4.2 Implementation Platform

The heavy trend towards 'Internet technology' led us to look for Internet server/applet solutions. During the research project, we had already created some early prototypes with Windows RAD tools (Rapid Application Development) but the expected networking infrastructure and internationally distributed pilot organisation made it rather clear that already the prototype should run on 'internet/intranet technology'.

The Knowledge Storage prototype was implemented in Windows NT environment using Microsoft Internet Information Server (IIS) and ASP scripting (Active Server Pages). The web server IIS is provided as an additional software component in all Windows NT delivery packages. The ASP enhancement to MS-IIS (from version 3.0) enables the web server to create dynamic HTML pages that are created on the fly based on the server application's interaction with the user.

Alternative development environments instead of MS-IIS accompanied with ASP could have been various server platforms and enhancements that have an open API (application program interface). At the time of the realisation of this study, possibilities that were found to be applicable were:

- Netscape Server (with LiveWire enhancements at the time of creation o the prototype)
- Apache (at the time of prototype implementation mainly for Unix/Linux environment; http://www.apache.org/)
- A simple and small but modifiable web server implementation (like TinyWEB that came with Delphi; see http://www.ritlabs.com/tinyweb/)
- PHP (Personal Home Pages; an enhancement to Apache web server and currently http://www.php.net/; also initially for Unix/Linux environment)
- traditional CGI applications and
- Java applets and servlets.

Because of existing in-project experience with ASP scripting, the requirement of quick implementation, and the compatibility with existing computer environment in the pilot organization, the MS-IIS platform was selected.

5.4.3 Implementation Basics

Technically, the prototype implementation of Knowledge Storage is a collection of ASP scripts with static HTML code that run attached to Microsoft Internet Information Server (IIS version 3.0 with ASP enhancements). To explain the development platform, there is an example in the following figure of the "Hello, World" written with ASP scripting (the example is extracted from the source code that comes with the ASP enhancement to IIS v 3.0).

ASP scripts contain code written in VBScript, a subset of Visual Basic, or JScript, a Microsoft variant of Java. ASP scripts include the possibility to be connected to a database that is accessed through the standard ODBC interface in the server machine. The database engine can thus be any engine that is supported by ODBC access. We wanted to test the system with lightweight solutions and used MS Access as the database throughout prototype development and piloting.

Static HTML ->	`<HTML>`
	`<HEAD><TITLE>Creating Hello World with Incremental`
	`Text Size Increase</TITLE></HEAD>`
Dynamic part ->	`<BODY BGCOLOR=#FFFFFF>`
written in Visual Basic	*`<% for i = 3 to 7 %>`*
	`       <FONT SIZE=<% = i %>>Hello World</FONT><BR>`
Static HTML ->	*`<% next %>`*
	`<BR>`
Include external code ->	`<BR>`
from a library file	*`<!--#include virtual="/ASPSamp/Samples/srcform.inc"--`*
	`>`
	`</BODY>`
	`</HTML>`

Figure 25. The "Hello, World" application with increasing font size written with MS-IIS ASP scripting (the code is provided attaced to the ASP enhancement by Microsoft). Make a note of the combination of the static HTML code and the dynamic parts written in Visual Basic. An ASP application returns static HTML pages to the user, which are readable with any contemporary web browser. No additional plug-ins or Java enhancements are required unless the system makes use of external Java applets. All the code written in Visual Basic or Java is translated in the server machine. The `<%` tag starts a VBscript or Jscript code section and tag `%>` ends it.

The ASP environment is a translating environment. Code is not compiled into executables. The ASP enhancement to the IIS web server (version 3.0; from version 4.0

and upwards these features has been included as a standard part in IIS) translates and runs the dynamic parts of the returned HTML pages at run-time. In certain circumstances, this may make the application somewhat slow. However, many large web sites have been implemented with the same platform (obviously www.microsoft.com) so it may even be useful for heavy production use as well. Our intention, however, was not in the implementation of a heavy-duty production system – just a prototype system for pilot use, but with the possibility to future extension. Therefore, we evaluated that this platform fulfils the requirements of our prototyping needs.

5.4.4 The Technical Modules of the Prototype

The prototype consisted of 33 files (modules) that provided the basic functionality for piloting. The main modules reflect the structure of the user interface. One reason for this is the fact that the user environment (UE) description created at the end of the Contextual Design stage defines the structure of system from the standpoint of the user. The user environment model provided basic implementation structure for our prototype. The modules that implement the user interface are described in table 10.

Table 10. The user interface and corresponding code files of the Knowledge Storage prototype. The `Basic.asp` is a module that provides the fixed standard parts (function bar on the left) of the user interface. The `lib.inc` is a general function library that provides general functions that are needed in several parts of the application.

Main User Interface Functionality	Module	Supporting/Following Modules
Login	Login.asp	ChangePassword.asp, SetGroup.asp, Basic.asp
My Role	Myrole.asp	SaveRole.asp, Basic.asp
Current	Current.asp	Basic.asp, View.asp, Vote.asp
Search	Search.asp	SearchResult.asp, View.asp, Vote.asp, Evaluation.asp, Lib.inc, Basic.asp
Add New	AddNew.asp	SaveNewEntry.asp, Lib.inc, Basic.asp
Templates	Template.asp	Basic.asp
Logout	Logout.asp	Bye.asp, Basic.asp, lib.inc
Help	Help.asp	Basic.asp
Feedback	Feedback.asp	Basic.asp, faq.htm

Additional modules that have not been described in the user environment (UE) specification provide required technical functionality for the system.

5.4.4.1.1 End-User Functions

The end-user functions that are listed in the chapter "The Prototype user Interface" are implemented as separate dynamic ASP/HTML pages. These functions are explained in more detail below.

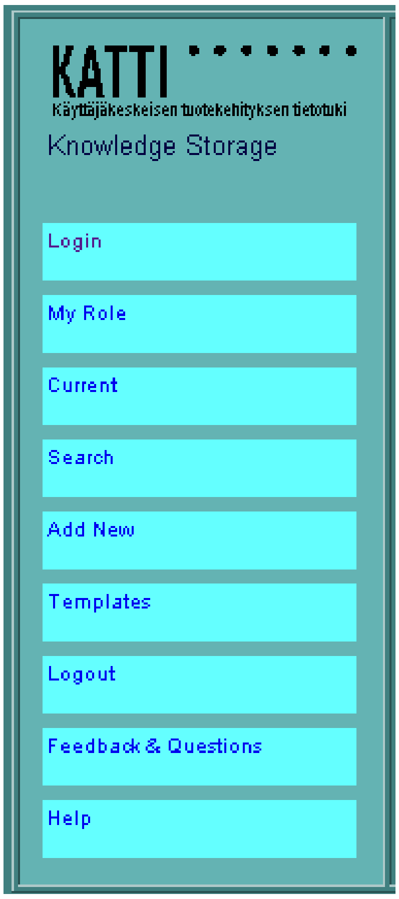
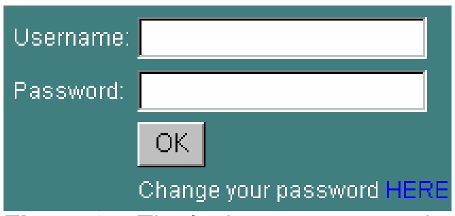


Figure 26. The basic function bar of the Knowledge Storage created by the `basic.asp` module.

The aim with the implementation of Knowledge Storage prototype has not been the creation sophisticated data and code structures. Instead, it provides a rather "quick and dirty" realisation that aims at implementing the conceptual model of the Knowledge Storage in a rather straightforward way for piloting purposes. To reach this goal, some technical implementation must have been done. This implementation is described here shortly.

The **basic.asp** module (see figure 26) contains the basic HTML code for the "function bar" that is on the left in the user interface. The HTML code in basic.asp is actually a "broken", partial, HTML page that is completed with the dynamic parts provided by each user interface module. The basic.asp HTML code is called in each user interface module through the ReadFile(FName) function that resides in the lib.inc module.

The **login.asp** module (see figure 27) presents the user with a typical authentication form consisting of a question about username and password. Users are not allowed to enter other parts of the system without authentication. Before having done the authentication, all other functions redirect the user to this module. Additionally, the user has the possibility to proceed to the "change password" screen that checks for the user's current password and asks for the new passwords.


Username:

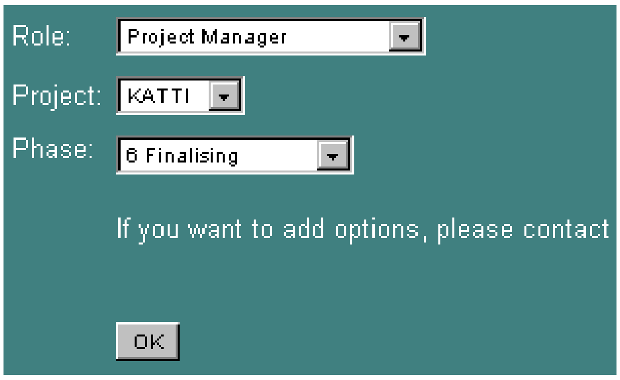
Password:

Figure 27. The login screen created by the `login.asp` module.

For security reasons, a knowledge storage system in real production use should make use of network traffic encryption like SSL. In the prototype, we did not implement this as a "standard feature". Instead, we relied on the possibility of the web server software to manage it. And, it was decided that if the pilot system runs behind a firewall in a "friendly environment", strong encryption is not required at the very beginning during piloting.

Even though the Knowledge Storage prototype was not implemented to include any in-built data traffic encryption, the passwords in the system were encrypted with a very simple and, thus weak, encryption. The goal with this encryption was to prevent "occasional users" from getting the passwords from the database of Knowledge Storage straight in readable form, should they get an unintended direct access to the database.

The password encryption uses a simple and easily reversible xor-algorithm but the decrypted password cannot be too easily derived without computational support. In the discussions with the technical representatives of the pilot organisation, this encryption was regarded as appropriate for the expected piloting circumstances. For possible changes in the algorithm (to be replaced with stronger ones), usage extensions, and unexpected multi-purpose use, the encryption algorithm is included in the library module lib.inc (documented later) just like many other generic functions and procedures of the system.


Role: Project Manager

Project: KATTI

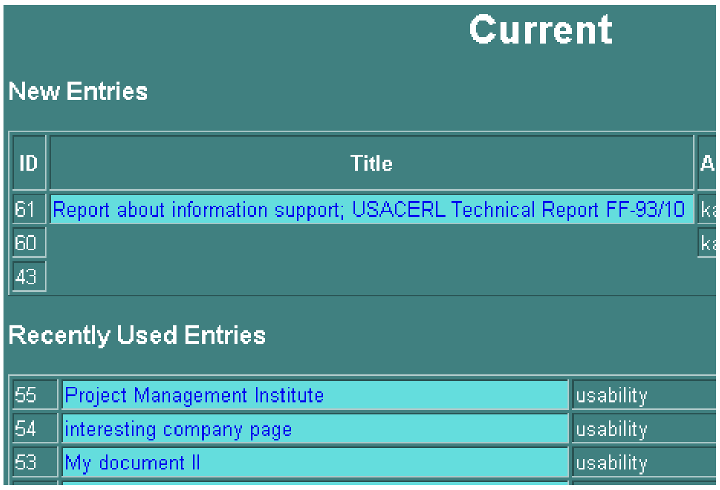
Phase: 6 Finalising

If you want to add options, please contact

Figure 28. The role selection screen created by the `MyRole.asp` module.

After login, the user is required to select a role for himself or herself for the following session with the Knowledge Storage. This is done with the **MyRole.asp** module (see figure 28). MyRole.asp presents the basic meta-tags to the user that will be connected to each knowledge entry. The meta-tags can later be used to search for knowledge entries based on the development context information. The meta-tags capture the current role of the user, the name of the development project the user is assigned to, and the phase of the project. This information characterises the development environment so that future reviews (sort of post-mortem analysis) of development information can be done from different perspectives.

MyRole.asp contains also one generic function called DetectSelected(FieldName, Value). This function is used to select specific items in the various "select boxes" in the user intreface that hold information about source and target projects, phases, and roles. The function is called by telling the name of the field (Fieldname) and the value (Value) of that field. If the value of the field in the database is (Value), that item in the list will get selected. By using this function, it is possible to provide the user with the same values that he or she had had during the previous session.



Current

New Entries

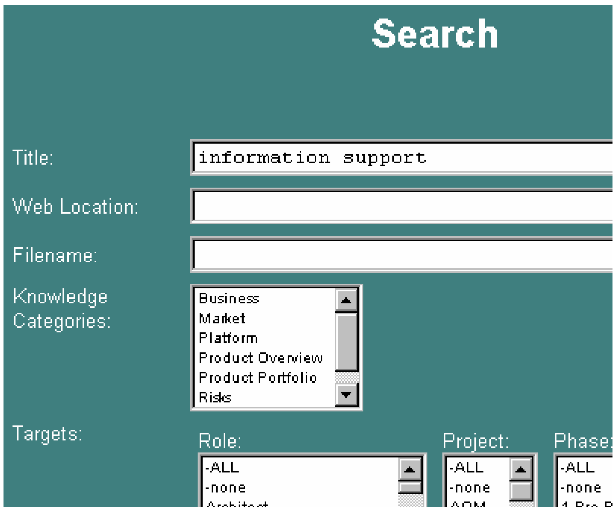
ID	Title	A
61	Report about information support; USACERL Technical Report FF-93/10	ka
60		ka
43		

Recently Used Entries

55	Project Management Institute	usability
54	interesting company page	usability
53	My document II	usability

Figure 29. The screen about most recent events/entries created by the `current.asp` module.

An important aspect in development work is keeping in touch with various development events and information that concerns the role that the user is performing. The **current.asp** module (figure 29) implements this functionality. As the system keeps a log of knowledge entries that the user (and the role) has reviewed, it also is capable of knowing what knowledge entries are new to the user. This module presents the completely new knowledge entries on top of the list in "New Entries". As it was detected in the contextual inquiry interviews during the design stage of Knowledge Storage that developers tend to deal with certain issues within a limited period of time, the system shows also a list of entries that have been used lately i.e. during the last two to three weeks.

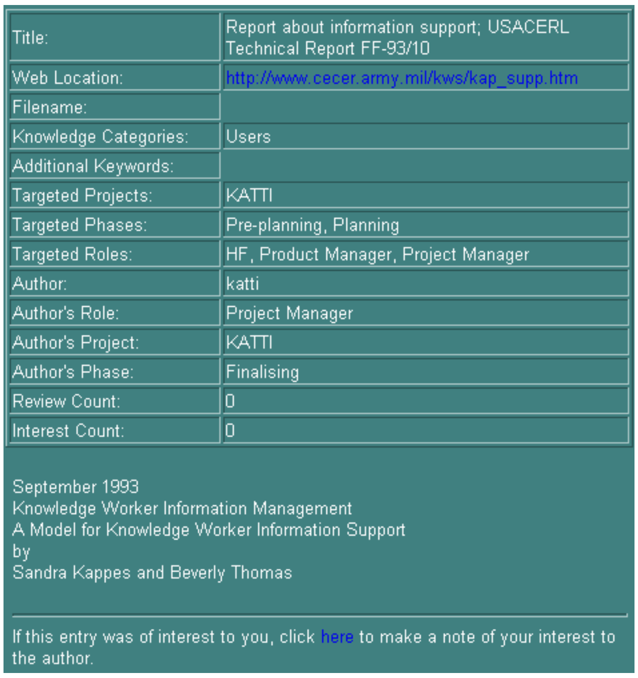


Search

Title:	information support
Web Location:	
Filename:	
Knowledge Categories:	Business Market Platform Product Overview Product Portfolio Risks
Targets:	Role: -ALL -none · Project: -ALL -none · Phase: -ALL -none

Figure 30. The search initiation screen created by the `search.asp` module.

As soon as the amount of information, Knowledge Entries, increases in the Knowledge storage database, searching becomes essential. The **search.asp** module (figure 30) provides the user with full possibilities to search of information in the database according to the available meta-information. The user can fill in and search for information from all knowledge entry fields. The search will be conducted from the database by using an appropriate SQL SELECT statement. The resulting set of database records is presented to the user as a standard list like in the "Current" screen. The list and its HTML formatting is done by module SearchResult.asp.



Title:	Report about information support; USACERL Technical Report FF-93/10
Web Location:	http://www.cecer.army.mil/kws/kap_supp.htm
Filename:	
Knowledge Categories:	Users
Additional Keywords:	
Targeted Projects:	KATTI
Targeted Phases:	Pre-planning, Planning
Targeted Roles:	HF, Product Manager, Project Manager
Author:	katti
Author's Role:	Project Manager
Author's Project:	KATTI
Author's Phase:	Finalising
Review Count:	0
Interest Count:	0

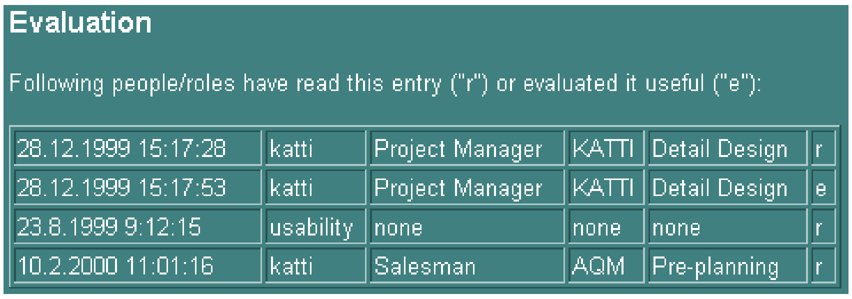
September 1993
Knowledge Worker Information Management
A Model for Knowledge Worker Information Support
by
Sandra Kappes and Beverly Thomas

If this entry was of interest to you, click here to make a note of your interest to the author.

Figure 31. The screen that shows the details of a single knowledge entry (the `view.asp` module).

The **view.asp** module (figure 31) shows the contents of an individual knowledge entry. Is shows the title of the entry, location of the actual document or a web address, creation date, project, phase, and creator's role and username. The free form explanation is also shown. View.asp also creates a record mark of the user that has seen that particular entry and provides the user with the possibility to "vote" for the knowledge entry. These statistics can then later be used to analyse the use of different information in the system.

The **vote.asp** module stores the "this is intersting for me right now" votes provided by the users to the database. This function does not provide any visual feedback to the user about the success of voting. This may be regarded as a usability defect. After having stored the result of the voting to the database, this function redirects the user to the "current" screen. The vote.asp records the vote in general level (appears in the listings) and creates a more detailed database entry of "people/roles that have considered this entry to be useful". These entries are shown by the module **evaluation.asp** when someone wants to review the voting results of a single knowledge entry.



Evaluation

Following people/roles have read this entry ("r") or evaluated it useful ("e"):

28.12.1999 15:17:28	katti	Project Manager	KATTI	Detail Design	r
28.12.1999 15:17:53	katti	Project Manager	KATTI	Detail Design	e
23.8.1999 9:12:15	usability	none	none	none	r
10.2.2000 11:01:16	katti	Salesman	AQM	Pre-planning	r

Figure 32. The evaluation screen that is created by the `evaluation.asp` module.

The **evaluation.asp** module (see figure 32) is used to provide the users with feedback about their contribution for the development community. This is done by presenting overall statistics about the usage of the knowledge entry to the user. The basic part of the user interface layout of this module is identical with module view.asp. Additionally, it shows detailed information about who have accessed and reviewed the knowledge entry in the past. The evaluation.asp shows also the number of votes "this is interesting to me right now" that the knowledge entry holds.

The adding of new entries to Knowledge Storage is done with the **AddNew.asp** module that is visually identical to the “search” screen. The user can fill in all information in the fields that he or she wants to and by pressing the “submit” button.

5.4.4.1.2 Library functions

The library module **lib.inc** contains generic functions that provide services for several modules. The lib.inc contains following functions. These functions are presented shortly below.

FUNCTION AdminRights(username) checks whether the user (username) has ADMIN RIGHTS. The return value is a logical true or false. Users with ADMIN

RIGHTS are allowed to maintain user accounts and various lists (projects, phases, roles etc.) provided by the system.

FUNCTION BinToAsc(strInput) converts a (password) string stored in numerical/binary format into readable "ASCII" characters. The function returns the binary formatted string in clear ASCII format.

FUNCTION EncryptString (strInput, strKey) does a simple XOR "encryption" to a string based on the string strInput and the encryption key strKey. Additionally, it does some additional extra proprietary string formatting. This function returns the encrypterd string. This function provides the password encryption and can be easily replaced with a stringer encryption algorithm if required. The basis for the function has been presented at http://www.vbonline.com/vb-mag/9712/item/q&a.htm[55].

The system makes use of a few generic string manipulation functions. These are explained below.

FUNCTION CheckString (s, endchar) checks whether a string (s) contains a specific character (endchar). In Knowledge Storage, (endchar) is often the quotation mark.

Function ToChar(str,char) reads characters from string (str) until a specific character (char) has been found. This function can be used to parse formatted text files that may provide some very simple "line-databases".

Function FromChar(str, char) reads characters from string (str) starting from character (char) and returns the resulting string.

Function ToCharFrom(str,char, StartCharNum) reads characters from string (str) until a specific character (char) has been found. The reading of (str) starts from character (StartCharNum). This function can be used to read a substring that appears in the middle of the original string (str) between two separating characters.

[55] The question that was presented in the web page concerning this issue was something that we wanted to have for piloting purposes: “I want to encrypt a password (string of 8 letters) in order to store it in a database, and I can't find any encryption function. I don't need a complex function. Can you help me?” The answer was “If you are not looking for a really strong encryption scheme but will settle for something that just keeps "casual" users out then there are several methods. One of the easiest is to XOR the bytes of the password with another value. The advantage is that if you XOR the resulting bytes with the key you get the original value back.”

Function ReadFile(FName) reads a text/HTML file (FName) from the disk and returns it as plain text. This function provides the possibility to use common sections, without the need to maintain redundant HTML code, in the user interface. In practice, the "function bar" that appears on the left in the prototype user interface, makes use of this function.

Database tables (in the prototype environment implemented with MS Access) that the system uses contains are presented in the table 11.

Table 11. The database tables in the prototype version of Knowledge Storage.

Table Name	Contents of the Table
Entries	The *knowledge items* with classifying meta-data
Users	User accounts
ListItems	All values that appear in the listboxes of the system (roles, project names, phases of the project etc.)
Logins	A log of logins to the system with date information
OldEntries	A table of "shown knowledge entries" for statistics and feedback about the contribution of each knowledge item to the development community.
Const	Administrative information about the system

5.5 Proof-of-concept Evaluation: Pilot Case

The main design and development of the Knowledge Storage concept was done between November 1998 and April 1999. The planning of the pilot project started in the beginning of May 1999. The very first expectations were that the prototype should be installed and that pilot use could begin by the beginning of July. However, the pilot use was delayed to begin at the beginning of August. The results that are presented in this paper cover the time between the beginning of August and the end of September.

Overview of CSCW Evaluation. The evaluation of CSCW systems has been addressed in the academic field for instance in the CSCW Evaluation Methodologies Workshop (1998) that aimed at the definition of "taxonomy of evaluation methodologies for CSCW systems, identifying the type of systems for which a technique is most useful, the stage of development in which a methodology is appropriate, the resources needed to conduct an evaluation, and the appropriate measures for the various techniques."[56]

[56] http://www.acm.org/sigchi/cscw98/program/workshops.html#w1 (checked 8-Jul-2004).

In their workshop paper, Steves & Scholtz (1998) present their evaluation about a Teamwave Workplace based CSCW application that is "used in the context of automatic gas-metal robotic welding". In their study, the CSCW application is used in trouble-shooting problem welds that require the analysis by geographically dispersed teams of people with a variety of expertise.

Steves & Scholtz make use of following evaluation data collection methods and tools: 1) user interviews, 2) direct observations, 3) an email list for the welding researchers, and 4) augmented log data from Teamwave Workplace. They present following categories of metrics that can be used in the evaluation of the CSCW application: 1) General (How is the system being used), 2) Communications (When are synchronous vs. asynchronous communications used), 3) Navigation (How well are users getting around in the system), 4) Room and tool use (How are various rooms being used and what sort of data is located in each? Is there an efficient organization scheme for data in and between related rooms?), 5) Collaborations relative to workflow (Can we characterize the collaborations relative to welding roles and/or to workflow), and 6) Critical measures of success, as defined by the user population.

In the case of the Knowledge Storage (KS), an evaluation scheme of the same kind was utilised. First, the Knowledge Storage concept was evaluated with the expected test users in a workshop-like interview situation. This evaluation was used in order to find out how well the terminology and concepts in KS were understood by the participants. Other evaluations with the running prototype were used to find out 1) General: How the system is being used ("Knowledge Storage Statistics"), 2) Communications: (statistics based on the utilisation / viewing of documents in KS) , 3) Navigation/usability: How well are users getting around in the system (usability testing), 4) Collaborations relative to workflow ("Questionnaire about Working Conditions"), and 5) Other measures defined by the user population ("Formal Feedback" and "Additional results"). These metrics and respective results are presented in more detail in the following chapters.

5.5.1 Initial Lightweight Concept Evaluation

In addition to the developers that were interviewed in the Contextual Design interviews, we presented the concept with supporting use scenarios and screen templates of the prototype to the very first pilot case participants. Most of the case participants were from another country (where the meeting was also held) and they had not participated in the creation of the requirements of the concept. During the initial presentation in the beginning of the pilot case, following issues emerged:

- The concept of a "role" appeared novel and interesting to be implemented as an information support system feature (or a data structure). People do not want to see all the information all the time, information overflow is a real issue. From this standpoint a system should not show all information to all roles all the time. Instead, it should filter information according to the role of the person and current phase of the project. "Role" was considered not to have been implemented in other applications in the way presented in the concept of Knowledge Storage.
- Participants of a development project are assigned in multiple projects. They have “different hats in their head” in different times. It is a good thing that Knowledge Storage allows flexible change of roles.
- Web as a technical platform was considered to be a feasible one and it was evaluated to offer almost unlimited linking between applications in the future as more and more of them will be available through the Web.
- In the beginning, when Knowledge Storage is being used as a “table of contents” to all development time information, all users must have suitable client software (e.g. a word processor or a spreadsheet) for reviewing the information in the linked files. As soon as it is possible to display the information and documents from the external sources right in the user interface of the Knowledge Storage, users have the possibility to review all information without additional client software other than the browser. These will truly enable global, mobile, even wireless, users to access the information. As more and more of the files are saved in browser compatible format, the users of Knowledge Storage become less dependent of different client applications.
- The requirement of storing a document separate from its index entry may be a critical disadvantage for the system. After having completed the storing of a document to a network file server, people may not remember the creation of an index entry. It should be possible to combine these two activities.
- The concept of documents that appear in multiple threads instead of a traditional hierarchical filing system may be too advanced. Computer users are so familiar with the hierarchical trees in their hard disks that it may be difficult to learn a new approach to manage the files through a searchable table of contents.
- The statistics about entries being “read” and evaluated “useful” provide feedback about the utility of information in different knowledge categories. During the paper

prototype evaluation, this feedback about other people's interest in "my entries" appeared to provide motivation for the users to use the system.

- There are several separate internal development information systems that should be integrated with the Knowledge Storage. If that could be done, it would be an advantage. The reduction of separate information systems would increase the systematic management of information.

5.5.2 Conformance of KS Concept Requirements to Pilot Users' Working Conditions

In order to evaluate the conformance of pilot users' (N=13) working environment (and information environment) to the topics gathered from the developers that participated in the initial Contextual Inquiry session (U1-U7; N=7). To do this, a questionnaire addressing the topics was created. The questionnaire is presented in detail in appendix A. The questionnaire had four sections:

1. Basic information was used to collect basic information about the roles of the participants: titles and basic work tasks.
2. Section "Information flows" was used to collect information about different information flows related to the development process: the sources, targets, frequencies and contents of the flows.
3. "Using documents" was used to collect information about how the participants use both the documents of their own project and the documents of other projects.
4. "Work process related problems" was used to collect information about how well the problems in the pilot project group match with the problems encountered in the project that was analysed in the interviews contributing the creation of the KS concept.

The questionnaire was answered by 13 persons: 6 developers, 3 consultants, 2 project managers, 1 team leader and 1 marketing manager.

As an overall result to the questions about direct information flow[57] between developers, answers to questions 2.4[58] and 2.5[59] revealed following issues:

- The 13 people dealt with a total of 79 different information elements.
- Only managers had been asked specific information by people in other projects and also stakeholders outside the company.
- Only 1 developer out of 6 had been asked specific information by people outside the project.
- Managers were the only ones that had asked specific information from people outside their own project

The work process / task related issues (problems) were addressed with the questions presented in the table 12.

Table 12. The question "Considering your current work and tasks, do you agree with the following statements?" revealed the most important problems in the pilot project participants' work. (for the full questionnaire, see appendix A)

1 = I fully disagree	AV = average
2 = I disagree	SD = standard deviation
3 = I do not know	SP = serious problem (if 9 or more persons of 13 answered 4 or 5)
4 = I agree	N = 13
5 = I fully agree	

n1	*n2*	*n3*	*n4*	*n5*	*AV*	*SD*	*SP*	
0	1	2	9	1	3,8	0,7	x	1. Information does not flow well enough between development project groups.
0	0	1	10	2	4,1	0,5	x	2. It is difficult to get a clear picture about what other projects and organisational units do.
0	3	2	7	1	3,5	1		3. There is no common language between organisational units sales, marketing, development etc.
0	4	2	7	0	3,2	0,9		4. There are not enough meetings between project groups and between organisational groups.
0	0	0	4	9	4,7	0,5	x	5. There is not enough information about customers and end users.
0	2	4	6	1	3,5	0,9		6. Customer visits are not documented well enough.

[57] Davis & al. (1999) have utilised information flow analysis in the research and development of the LIRÉ (Living REpository) prototype application that is a Web-based information and document repository to support collaborative work.

[58] Question 2.4: Mark with "X" the stakeholders that have explicitly and clearly expressed that they need certain information from you. For each line you can specify the contents of the information. (This question was to reveal the targets of the formal information flows beginning from the answerer and the information content in these flows.)

[59] Question 2.5: Mark with "X" the stakeholders to whom you have explicitly and clearly expressed what information you need from them. For each line you can specify the contents of the information. (This question was to reveal the sources of the formal information flows ending to the answerer and the information content in these flows.)

0	0	3	8	2	3,9	0,6	x	7. Handling of the feedback from customers is difficult because we have no tools or process for it.
0	2	7	3	1	3,2	0,8		8. It takes too much time for new employees to get familiar with their work.
0	0	2	10	1	3,9	0,5	x	9. When a person leaves, too little of his/her knowledge remains in our organisation.
0	3	7	1	1	2,8	1,2		10. Development work guidelines are not followed well enough.
0	2	1	9	1	3,7	0,9	x	11. There are too many internal information systems.
0	1	1	10	1	3,8	0,7	x	12. Internal information systems are not integrated well enough.
0	0	4	6	3	3,9	0,8	x	13. It is often hard to find answers to specific questions from the development information systems.
0	2	3	7	1	3,5	0,9		14. Changes to plans, requirements or specifications are not documented well enough.
0	5	2	6	0	3,1	1		15. Keeping important documents up-to-date requires too much effort.
0	6	2	5	0	2,9	1		16. I get too much information that is not relevant to me right now but would be at some other time.
0	1	7	5	0	3,3	0,6		17. I have to poll too many information systems to get important information.
1	6	0	6	0	2,8	1,1		18. I have to interrupt my work too often to answer detailed support-like questions.
0	3	1	9	0	3,5	0,9	x	19. I do not know whether the documents I create are read.
0	1	4	7	1	3,6	0,8		20. I do not know well enough whether the documents I create are useful for others.

To analyse and prioritise the most important work related problems, it was defined, that a problem is considered serious if at least 9 out of 13 people (69%) answered that the problem exists in their work (answer was 4 or 5). Based on this, following problems were considered serious:

- Information does not flow well enough between development project groups.
- It is difficult to get a clear picture about what other projects and organisational units do.
- There is not enough information about customers and end users.
- Handling of the feedback from customers is difficult because we have no tools or process for it.
- When a person leaves, too little of his/her knowledge remains in our organisation.
- There are too many internal information systems.
- Internal information systems are not integrated well enough.
- It is often hard to find answers to specific questions from the development information systems.
- I do not know whether the documents I create are read.

In general, these issues were well aligned with the results from the initial contextual inquiry interviews. These comments provided some additional points for the creation and testing of the functionality for the prototype.

5.5.3 Case Organisation and Pilot Project

The pilot project in which the prototype of the Knowledge Storage was applied took place in a development organisation of about 25 persons. The product development work in the organisation is conducted in separate projects that contribute to existing or new products.

There were 11 participants that tested the Knowledge Storage during the evaluation period. The pilot project was conducted in another country and organisational unity than where the prototype was developed. Initially, the development environment was evaluated to be rather similar in these two locations. There are some common development systems that people from both countries share with each other but because of technical restrictions not all systems have been available for all participants.

One "composite development project" was assigned to do the initial pilot testing for the system. "Composite project" means a project that aims at creating a new product from parts of other existing or emerging products. Not all participants of the project were directly assigned to the selected development project; there were a few persons who were working full-time for the particular project. An additional half a dozen persons participated in the project on a part-time basis. They worked primarily to other projects and were sort of "associate developers" from the standpoint of the pilot project. This created demands for virtual collaboration throughout the project. Knowledge Storage was expected to support this.

The participants of the pilot project were selected at a time when the prototype was being installed to the case organisation. Originally, it was expected that the amount of pilot users would be somewhere between five or ten persons. Instead, a total of 24 people were initially announced to be registered to the user database of the system at the beginning. A total of 11 persons actually tested the system during the testing period.

5.5.4 Background Information

Information that supported the creation and piloting of the Knowledge Storage was gathered during the "current state analysis" stage of the research project mainly during year 1998. Background information was gathered by interviewing the practitioners in the case organisation about their current working practices and working conditions, and by analysing the quality system documentation. In addition to these, a study of current development systems was conducted. The results of these activities were reported in internal reports that were delivered to the case organisation.

5.5.5 Evaluation Metrics

To evaluate the results of the pilot project in a systematic way, metrics were introduced that reflect the usage pilot project from different standpoints. The measurement was to be done by the value added of the system for the pilot project participants, the match of the implemented system compared to the vision and the usage of the system's different elements. Evaluation was realised by using an evaluation set that consists of

- a questionnaire (at the beginning and at end of the pilot project)
- discussions with the pilot case participants (in intermediate and closing meetings)
- usability test of the pilot prototype
- analysis of informal feedback that we received during piloting.

Additional metrics was provided by the prototype itself: the Knowledge Storage Prototype gathered usage statistics and since the system actually runs attached to a web server, we had the possibility to use the log files that the server creates.

5.5.5.1 Questionnaire about Working Conditions

As the participants of the pilot project were located in two different countries, it was not possible to do a comprehensive interview and observation study of the working conditions in both countries. This led to the creation of a questionnaire that was used for rough screening of the working conditions mainly in the other country. The data gathered with the questionnaire was supported with a short discussion in a meeting with the people in the other country before starting piloting.

The questionnaire consisted of following sections:

- Background: job title, work responsibilities, projects
- Use of current development information systems
- Communication with other stakeholders of the development project
- Use of development time documentation
- Knowledge management related problems in current work environment

The full questionnaire is presented in appendix A.

5.5.5.2 Usability Test

According to the principles of user-centred development, the testing of a system with end users is compulsory. User observation is one way to fulfil this requirement. User observations can be conducted for example by applying usability tests or Contextual Inquiry as the method. A formal usability test of the prototype was carried out.

The test users were asked to conduct specific tasks that conform to their typical working tasks. These tasks were detected, described and documented during the Contextual Design of the Knowledge Storage. The usability test sessions were observed and the events were recorded by making detailed written notes with a PDA and using a VCR to record the voice of the test user and the events on the computer screen and keyboard. The notes and videotapes were time-synchronised already during the test to enable easy retrieval of interesting events later.

The tasks and environment was outlined during the Contextual Design of the Knowledge Storage. Every test user was given a story to simulate a situation explaining the test tasks. The story was in line with the role of each test user. Because of the story, some tasks were slightly different for each test user in their context, although they were expected to do the same functions.

The story specified following issues: the role of test user, the current project, the phase of that project, and the context of tasks (this provided a meaningful purpose for the used document and web page). There was a following briefing for all participants of the test: "You are (project manager, marketer, technical support, developer) in a project developing a tool for knowledge management, which is now in implementing phase. You have the Knowledge Storage system that you can use."

The usability test consisted of seven tasks that were provided to the test users both spoken and written:

1. Find a document that you need in order to write another document.
2. Evaluate whether the document that you just read was useful for you.
3. Check out what kind of information in other projects has been targeted to the same role that you have.
4. You found an interesting website that you evaluate to be useful for other people in your project. Make this information available for all participants.

5. You have created a document that is useful for specific roles in all projects. Save the file to a shared document library and reachable from the Knowledge Storage.
6. You are about to visit a customer site. Find a form that can be used to gather information about the customer visit.
7. Send feedback to the system administrators and logout from the Knowledge Storage.

5.5.5.3 Knowledge Storage Statistics

The Knowledge Storage records statistics about its usage. The use of application logs has been applied earlier[60] in e.g. Nieminen & al (1995). As the Knowledge Storage runs attached to a Web server, it gathers also standard log files about usage by default that can be used. Additional in-system statistics were recorded about logins to the system to evaluate the frequency of use.

5.5.5.4 Formal feedback

We included a specific "feedback" function in the Knowledge Storage prototype for gathering formal feedback. This was implemented as an email link that the users were able to view at any time while using the Knowledge Storage.

5.5.6 Results from Pilot Evaluation

5.5.6.1 Pilot Case Events

Despite the fact that Knowledge Storage was a prototype, the introduction of the system was done according to the principles of a good introduction process.

Principles of a good change management project have been presented e.g. in Kotter (1995). From a closer perspective, Kosonen & al. (1998) have also addressed this topic. These models include issues like recognition of important problems, management support, visioning and goal setting together with all participants, achievement of fast and visible results, and evaluation of the results.

The events that preceded the pilot project ("pilot planning") consisted of following stages:

[60] The utilisation of usage log-files has also been addressed by e.g. Hilbert & Redmiles (1999, 2001).

1. Planning of the pilot case (detection of major problems; getting management support)
2. Introduction of the ideas and the prototype to the pilot participants
3. Gathering of information about the state-of-the-practice in the pilot organisation in the beginning
4. Installation of the prototype system
5. Pilot use with feedback gathering.

The piloting itself consisted of following activities:
1. Questionnaire to participants about current work practices for verification of similarity to the original interviews that provided design data for Knowledge Storage.
2. Initial introduction to Knowledge Storage for the expected participants (alongside the training of Contextual Design as the results of Contextual design were expected to be managed with Knowledge Storage)
3. Pilot use began at 1st August and it ended at 30th September 1999. The usage was monitored and evaluated afterwards. Support was provided throughout the pilot use.
4. Intermediate evaluation session took place at the beginning of September; The vision and goal of the project was expected to change as a result of this evaluation session
5. Reporting the results; conducted by the participating researchers from the Helsinki University of Technology
6. Presentation of the results to the pilot users at the end of September, 1999
7. Closing the pilot case and decisions about next steps (October, 1999)

5.5.6.2 Questionnaire about Working Conditions

A questionnaire about working conditions was sent to the expected pilot case participants. A total of 13 answers were received. It appeared that many of the important issues that were detected to be important in Contextual Design were considered in a rather similar way in the pilot organisation. The participants of the pilot organisation were using almost same systems and procedures as people in the organisation whose representatives were interviewed during Contextual Design.

The most important problems mentioned in section “Knowledge management related problems in current work environment” are listed in table 13.

The problems that were regarded as most important were similar to the important problems in the group that had participated in the contextual interview. This list of important problems provides data about the conceptual level match of the Knowledge Storage to the problems that practitioners encounter.

Table 13. The most important problems detected by the questionnaire (in appendix A) (N=13; scale 1 = I fully disagree and 5 = I fully agree). The total amount of problem statements was 20.

Statement	Average	Standard deviation
Statement 5. There is not enough information about customers and end users	4.7	0.5
Statement 2. It is difficult to get a clear picture about what other projects and organisational units do.	4.1	0.5
Statement 7. Handling of the feedback from customers is difficult because we have no tools or process for it.	3.9	0.6
Statement 9. When a person leaves, too little of his/her knowledge remains in our organisation.	3.9	0.5
Statement 13. It is often hard to find answers to specific questions from the development information systems.	3.9	0.8

5.5.6.3 Usability Testing

Usability testing took place after the piloting had begun and the results of the usability test were not implemented in the pilot system during the evaluation period.

There were four participants in the usability tests. Participants were not from the pilot case organisation unit but from the same site in which the development of Knowledge Storage had been conducted. The length of each usability test was between 40 and 75 minutes and all test users conducted all tasks.

The main results from the usability test were following:

- The need for integration of Knowledge Storage and internal development systems is evident, especially with company standard document storage. It is too difficult and time consuming to make an entry manually. People need to do additional tasks in order to get the benefits of Knowledge Storage. To Knowledge Storage it is crucial to have a “critical mass” of information available in order to be beneficial for the users, so it is most important that the creation of entries is easy.

- In the system, there are many concepts that are not clear for user. Either the meaning of concepts should be self-evident, or there should be a brief explanation about them. For instance the concept "Filename" in page "Add new" is easy to misunderstand being a field for uploading a document to the system when it acts just as getting the "path" to the file.
- The possibility to upload files to the system is a thing worth considering. It has certain benefits, like the possibility to add documents to entries outside the Intranet systems. This, however, brings among problems about version management.
- Role thinking behind the system should be made easy to understand for new users. This could be done by some sort of brief introduction in Help or "Current" view.
- Search function should be made simpler; the current "Search" view contains too many items. It was mentioned to be "frightening". There should be an advanced help for "Search", though. It should be possible to use an asterisk (*) as a wildcard in Search.
- Currently authors cannot edit or delete their entries. This can be a problem, if they for example want to correct some mistakes in them.

5.5.6.4 Knowledge Storage Statistics

The statistics of the usage[61] of the prototype showed that 11 persons from the pilot organisation had used the system during the initial piloting. In addition to that, the researchers and developers of the Knowledge Storage had used the system.

According to the statistics, the pilot project participants had logged into the system or changed their role 48 times during the time between 2nd August and 7th September (see figure 33). The use of the system was most intensive in the beginning.

During the pilot period, the pilot group submitted 14 documents that were indexed with the Knowledge Storage. The documents were from three different development projects and covered various development issues like minutes of a meeting, product descriptions and test results. In addition to the 14 documents, there were three additional documents that had been submitted to the Knowledge Storage before the actual piloting had started.

[61] This analysis methodology has been applied earlier by the author e.g. in Nieminen & al (1995) and presented also in Kasvi & al. (2000).

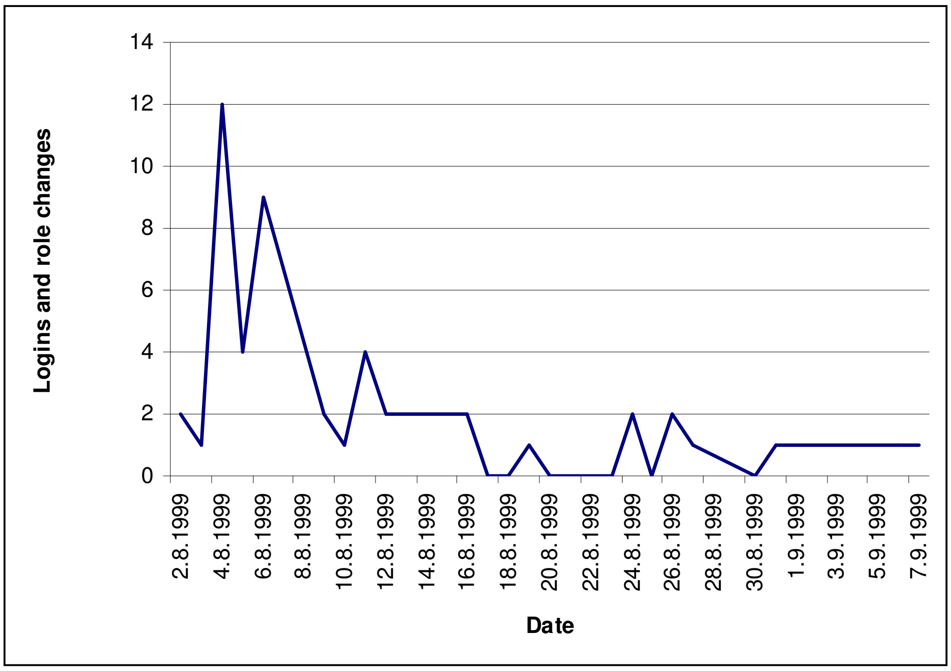


Figure 33. The amount of logins and role changes in Knowledge Storage during the initial pilot period.

5.5.6.5 Formal Feedback

Only a few feedback messages were delivered during the beginning period of the pilot case. These messages covered topics like the registration and amount of participants, the gathering of proper documents that can be put into the Knowledge Storage and remarks about the functionality of the prototype.

One of the feedback messages after testing of about three weeks revealed that the system was not fulfilling the expectations that the researchers and developers had thought about. The pilot case participants had their doubts about being able to "help in the evaluation" to get experiences from using the system. The users felt that they did not have so many documents that they would have needed such indexing and search capabilities within the pilot project. This was evaluated to be one of the most important messages during the beginning of the pilot period.

5.5.6.6 Additional Results

The latest results were gathered in a teleconference with two representatives of the pilot participants. In this meeting it was discussed about the results of the questionnaire that had just been conducted. The results included following issues.

In a multi-project environment, information about neighbourhood projects is important. It was evaluated to be very important especially in the pilot case because it was actually a composition of different development projects and some existing products.

The predefined knowledge categories were not suitable despite the fact that they were gathered from the organisation's quality system and quality documentation. It was suggested that all stakeholders would have the possibility to add new knowledge categories into the system.

The prototype was evaluated as "easy to use" but there were some important features missing. One of the most annoying things was that editing and deleting knowledge entries was not possible. This was detected to be a very important feature. In the current form, the system is far too strict. It is, for example, not possible to make a correction to typos.

The "Search" functionality had required additional guidance. The results of search were not something that the users would have expected to come out from the system. This defect was detected, however, to be a technical bug in the prototype.

The integration of Knowledge Storage between other development systems appeared to be crucial as well. As one of the underlying ideas during the development of knowledge storage had been the integration of different development systems, it should get realised already in the prototype. Before the system can really help development projects, the Knowledge Storage should act as a real front end to the rest of the systems. These kinds of systems are, for instance, document libraries and company internal messaging systems.

In the version of Knowledge Storage that was applied in the pilot case, the system showed all new entries to all persons who logged into the system (these are seen in the "Current" page). However, the participants mentioned that they would like to see only those entries that have been targeted to them. This way they would see only those entries that seem to be of relevance to them at the time. This would decrease the information overflow and ease the focusing to essential issues.

The Knowledge Storage requires that the users of the system create all indexing information by themselves[62]. The pilot users would have liked the support of an automated indexing system instead of just relying on the “human created indexes”. There are tools available for this in the commercial market and the testing of these tools was encouraged.

Since the pilot project participants were software developers creating software for the web environment as well, they were interested in the technical realisation of the system. This fact needs to be taken into account when evaluating the results of the pilot project. The developers had interest towards the “search” and the “Web based database record editing and updating” functionality of the system.

5.6 Conclusions about Knowledge Storage

Technically and on the conceptual level, Knowledge Storage provides answers to many of the problems that have been considered difficult in product development work according to the empirical data presented in this work. Knowledge Storage is a concept of active issue based documentation and information sharing system and it allows knowledge sharing between different product development groups. It is a kind of a combination of document storage, email, and workflow system and allows targeting of information. This feature provides possibilities for decreasing the “information overflow” problem.

A key concept in the system is "role". This concept enables the system takes into account the organisational structure of development work. It is able to provide different “role based” views to the information that it holds. To support feedback about each user’s contribution for the development organisation overall, the system can provide feedback to all persons via ”Contribution Evaluation” feature. According to the interviews, this feature was evaluated to motivate people in using the system.

Basically, the Knowledge Storage concept gives answers to many of the problems that were detected during the development and considered important.

[62] This result is similar to the result presented by Davis & al. (1999). In chapter *Caveats and Concerns* they mention that “The creation and maintenance of the keyword network turned out to be the single most critical problem with the ongoing use of LIRÉ”.

5.6.1 Knowledge Storage in the management of User-centred Information

An important question in the whole study is how can Knowledge Storage be used to manage the user centred issues presented by those topics? The ultimate goal in applying Knowledge Storage in user-centred design activities is the “on-line knowledge” about things that the organisation knows about usability and user. The aim is to enable the organisation to share this information with all required stakeholders that may be organisationally separated from each other.

From the standpoint of usability and end-user centred design following aspects are interesting:

- Knowledge categories direct developers to take into account users and usability. With the KS concept it is possible to follow, how that information is being constructed.
- User-centred issues can act as exit criteria in a project phase; the criteria can be set to “sufficient amount of information about users is known by the project group”. And the information can be gathered by anyone in the group as they have the possibility to review through knowledge storage, what are the still open questions.
- With the KS concept one can point out and manage roles that have a mission to gather, distribute and follow information that is related to users.
- It is possible to include explicit management of user related information into specific phases of a project. It is possible to direct pieces of information to be used in a specific phase of a project so the information does not burden the developers in a wrong phase in advance.
- It is possible to restrict the visibility of user-centred issues if there are representatives and stakeholders in the project who do not need it.
- The KS concept offers ready-made and well argumented templates and work instructions for developers’ use that address user-centred topics
- The management of gathering and using of development information can happen via targeting (in advance with templates) and tracing

“User-centredness” has been defined in this study mainly from the standpoint presented by

- ISO 9241
- ISO 13407
- Usability Maturity Model UMM / HCD
- Contextual Design development method

First, the Knowledge Storage must support the design and realisation of the user-centred development practices in a managed way. Second, the Knowledge Storage must support the creation and further editing of the user-centred documents that are created during that development process.

Knowledge Storage must be capable of storing various human-centred design artefacts that are created, for instance, during Contextual Design. The “storable CD artefacts” that need to be considered are the Contextual Inquiry memos and tapes, affinity diagrams, affinity insights, affinity design ideas, flow diagram, flow insights, flow design ideas, consolidated flow visions, consolidated vision, concrete vision, storyboards, user environment, user interface, paper prototype, and shipping UI.

The origins of these user-centred artefacts are in various formats: the memos appear in hand written sheets of paper, the interview tapes are in dictation or c-cassettes. A knowledge storage system must either be able to store these artefacts internally, or it may contain information about where the artefacts physically or logically reside. Based on the meta-information that the system contains these artefacts and their contents become available for all development practitioners.

5.6.2 Knowledge Storage in the Management of General Product Development Information

Even though the development of Knowledge Storage started from the standpoint of the management of user-centred information, the resulting structure is capable of holding other types of information as well. The *knowledge categories* in Knowledge Storage direct the participants to take into account those issues that have been considered to be of great importance for the project in general.

The follow-up of information accumulation is possible to implement with Knowledge Storage. It is also possible to follow the gathering of this information in a systematic way. By applying this, it is even possible to set up exit criteria based on “sufficient amount of information in a specific knowledge category” and start managing development projects by the amount and quality of information that has been produced (“management by information”).

The general process information for product development may be in the form or quality instructions. Quality instructions can be stored and accessed through the system. They can be stored as document templates or Knowledge Storage can contain links the company internal Web sites that contain quality instructions. These instructions or links can be marked to appear in a specific stage of the project for specific roles that are responsible in keeping that information up-to-date. It is also possible to attach phase-specific quality templates in the system so that these documents appear automatically as soon as the project proceeds to that phase.

Knowledge Storage can also be used to manage expertise and responsibilities. The system can be used to point out roles, whose mission is to collect distribute and browse specific important information categories. It is also possible to instruct the development practitioners to target information to specific roles. An important feature is the possibility to mark a specific knowledge item to be used in a specific stage of a project. This way, things do not get forgotten but the information does not burden its users prematurely.

Several development activities require some kind of readymade checklists, forms, and so on. Knowledge storage can be used to deliver these kinds of documents throughout the organisation without knowing who are the actual persons that are responsible in managing this issue. The concept "role" takes care of this.

An important feature of the prototype implementation of Knowledge Storage is that it is available for developers' in a geographically and timely divided development organisation with standard wide area networking techniques.

5.6.3 Knowledge Storage Mappings to Design Rationale and Knowledge Management

The Knowledge Storage aims at implementing several elements of design rationale and knowledge management concepts.

Design support. *Dependency management* is implemented through the linking of knowledge entries. As each knowledge entry will be tagged with categorising information, one can search for entries that have similar characteristics. If users have manually linked entries to each other, it is possible to follow these threads throughout the storage. As Knowledge Storage keeps track of entries that each role has read, it may even be possible to create automatic or semiautomatic analyses from this network of information.

Collaboration happens through the Knowledge Storage user interface: all participants of the project can collaborate through a unite interface and access documents and other artefacts in a shared information library. *Reuse support* is provided by providing the project participants with access to past information and by allowing people to search for information that has been created in other project groups. **Maintenance support** is as well provided through the access to past information.

Learning support is also initially based on past knowledge that exists in the system. Learning happens e.g. through the exploration of knowledge entries that have been created in other development projects. Additionally, learning support is realised in the familiarisation of new developers or other employees. As experienced designers browse through the knowledge entries, they may get new ideas by combining information that exists in the storage. This is in concordance with the theory of Nonaka & Takeuchi (1995): this activity can be internalisation or combination.

Documentation support happens through indexing. As all documents get indexed, the contribution to documentation and search of documentation will be improved. Documentation becomes more visible to the participants of a development project. It is even possible to start thinking of documents as collections of knowledge entries. In this concept, a document is a snapshot of entries at a specific moment of time. Documents become dynamic elements that improve all the time.

Support for traceability gets enabled through the tracking of use of documents and browsing threads. With this support, it is possible to get answers to questions like “how did we end up here” and “why did we select this approach/solution” when we can see the activity of different roles in the development information space.

Knowledge Management. When examined against the knowledge creation theory (Nonaka & Takeuchi 1995), the Knowledge Storage concept can be seen to support the creation of organisational knowledge: the participants can share information through the system and they can create concepts with its support and justify them. The information management improves as qualified and verified arguments will become explicit to support important decisions.

With the aid of Knowledge Storage, information exchange between projects can happen transparently. As the Knowledge Storage concept defines “project” as one context attribute, every knowledge item gets tagged to specific project. If different projects are alone to look for information from the other project, the meta-information tags can be used to search for this data.

Knowledge Storage can also be used to support the detection of best practices. The templates in the system can be designed to be dynamic. Should someone in the organisation require information that does not exist in the current templates, he or she may add this question in it. The next person who is gathering information with the template can get the answer to the newly added question as well. With this structure, information accumulates rapidly.

The interviewed developers noted that they rarely get feedback about the work that they have done. Knowledge storage provides both automatic feedback mechanisms and an easy-to-use “this is good information” voting mechanism. All users of Knowledge Storage can review knowledge items that they have created any time they want and they can see the amount of activity that relates to the knowledge item that they have produced.

5.6.4 Feasibility: Evaluation of the Initial Experiences from Pilot Usage

Based on the experiences gained in the pilot case study, is Knowledge Storage a feasible solution for managing development time information? Can it be used to share user related information?

The answers in the formal feedback messages together with issues presented in the evaluation teleconference and assess the activity in the Knowledge Storage information space, it can be concluded that that the Knowledge Storage implementation in its current form does not fulfil all the expectations and requirements that were set at the beginning of the development project. Despite this fact, the pilot case participants did not feel that the approach has come to a dead end: there are still possibilities that Knowledge Storage could support development work in slightly different settings. Even the people who took part in the evaluation did not consider the evaluation environment optimal. It was confirmed that the initial problems to which Knowledge Storage should provide answers do still exist in the development work. Despite the problems encountered during piloting, the pilot users were still interested in further testing.

Why did the implementation not provide expected utility for the pilot participants? Possible causes for the rejection in the pilot case are:

1. **Small amount of knowledge items in the project**. The basis for the prototype implementation was on the analysis of the usage of current development time information systems. Knowledge Storage was visioned to be a system for managing a huge mass of rather small pieces of information. However, the group chosen for piloting used only a few documents in their work and they did not see any benefit in using the system for managing these documents. The information item / note sheet approach did not get realized during piloting. One reason for the small amount of knowledge items may be the short pilot period of three months.
2. **Emphasis on document-centric information management approach**. The prototype implementation was presented to the pilot group as an index to different documents that could be valuable in project development. The possibility of using of the system as a discussion platform or a knowledge-gathering tool was not presented to the group well enough and this kind of usage did not emerge during the beginning of the piloting.
3. **Differences in user groups**. The user environment of the pilot group was not similar enough to the environment of the users who participated in the Contextual Inquiry. The concept and the prototype were developed using Contextual Design method. The needs of the users who participated in the Contextual Inquiry did not perhaps still match well enough to the needs of the users in the pilot group.
4. **Lack of participation in the design of the prototype**. As the implementation of the Knowledge Storage concept calls for changes in work practices, the people whose work was about to change should have been enabled to take part in the concept development. The lack of participation can be a sufficient reason for the rejection.
5. **Small group**. The pilot group was too small. They did not see any benefit for sharing their knowledge using the new system compared to their current practices.
6. **Single project**. The piloting took place in a “single-project environment” without the need for support for multiple projects. Since we actually detected during the Contextual Design that information flow within a single project group happened rather smoothly, Knowledge Storage could not provide any value added to this

communication. We may claim that the selection of pilot project was not proper enough.

7. **No additional documentation / information was available through Knowledge Storage**. The prototype system did not include the artifacts created in the Contextual Design process even though it was visioned to do so.
8. **Finding information from current information storage was not a problem in the beginning, after all**. The pilot users were very able to find the single-project information they need from the current systems. It was very difficult for the prototype system to be good enough for them to change their current information management practices.
9. **Starting the use of Knowledge Storage in the middle of an existing project may not be applicable**. As the use of Knowledge Storage started in the middle of an existing project, lots of documentation already existed. Similarly, the project had already constructed its own information management mechanisms. The participants of the project were aware of the documentation that existed in the system already. The transfer of that information to another document storage is not feasible. It may be better to start the application of this kind of a system in the beginning of a new project. That way the practitioners may also easier adopt new working styles that are required in the application of such a system.

As a conclusion of the rejection we may say that Knowledge Storage, in its currently implemented form, does not provide clear enough utility for a development group that is located geographically in a single and same location and that is timely uniform. A main hindrance for widespread use includes the need for additional overhead work that is required by the “double saving” of the documents. First, the user is required to save the document in the normal manner and then she is required to create an index entry of the document to the Knowledge Storage “table of contents”.

To some extent, a result like this is not a surprising one. One of the first reports about the “failure” of CSCW applications has been presented by Grudin already in 1988. Grudin presents three main problems that make it difficult for CSCW applications to succeed: 1) The disparity between who does the work and who gets the benefit, 2) The breakdown of intuitive decision-making, and 3) The underestimated difficulty of evaluating CSCW applications. Even though these issues were considered in the development of Knowledge Storage, it seems to be so that the topics should have received even more consideration, even direct goal-setting and clear objectives that would have pointed out these problems. The research presented in this work provides partial contribution to problem number 3, but the other two problems remain unanswered and require additional studies.

Knowledge Storage may be used in a variety of ways. It can be applied as a concept or as an instrument. The concept might be possible to embed the concept to company specific platforms. The implemented system can be used as an index to cross-project and cross-unit documents and issues. The index can cover topics that are managed anywhere in the organisation. The Knowledge Storage provides an idea of gathering small pieces of information instead of huge documents. The small knowledge items can then be combined together to provide dynamic documents about a specific issue. The knowledge items may provide ideas for thoughts and discussions and should the discussions be conducted within Knowledge Storage itself, it provides detailed history of reasoning.

6 Conclusions and Discussion

The work presented in this thesis has had its focus in the conceptual creation and practical piloting of the supportive information management system for the user-centred development organisation. As user-centred development started to gain popularity outside the academia in real product development organisations in the mid- and late 1990's, the need for structured means for managing the information gathered with the user-centred development methods seemed to start requiring practical support as well.

However, practical findings from the different case companies within the study pointed out that mere development and introduction of new tools for development does not provide nor predict success in promoting a new important design aspect. Many companies did already have even overlapping information systems for the management of important development and product support related information – and this was certainly not always considered as good support for the development work. In fact, an often mentioned complaint was the diversity of the various information systems. Evidently, there seemed to be a need for emphasis and support from different sources within the organisation should a new important topic require special development focus. The surroundings of such an information management system need to be considered alongside the development of the practical tooling.

All these findings together led to somewhat broader – and at the same time not so deep analysis in every addressed area – approach in solving the problem and research question at hand. In many ways it is evident that the (original) main emphasis in this thesis has been in the modelling of the "systematic user-centred information management system". However, intermediate findings during the implementation of the different parts of the study, additional significant factors pointed out the need to include aligned issues into the thesis.

As a summary of this, the final results of this thesis consist of

1. The literature-based definition of the underlying concepts about user-centredness and usability
2. The illustration of the user-centred development methods that constitute practical means for conducting user-centred development activities
3. A description of the development environment that supposedly requires support to enable the “design for usability” approach within the organisation
4. The conceptual construction of the framework that may provide “the embedding environment” in the organisation in which the user-centred attitude, approach, methodology and tooling may reside
5. The conceptual model of the Knowledge Storage that may be applied as an information (or knowledge) carrier platform for the various “design for x” issues including “design for usability”
6. The prototype implementation together with initial pilot testing of the Knowledge Storage that acts on the other hand as 1) a proof-of-concept for the conceptual model about the information carrier medium and on the other hand as 2) a research instrument that can be used to reveal the possible shortcomings of such a concept and implementation (this provides a basis for deeper investigation of the concept and possible improvements in the future)

Not all of the abovementioned topics, however, provide new information for the academia. Topics 1 and 2 form the basis onto which the latter topics are constructed. Topic 3 is a state-of-the-practice survey into the current situation in the selected case development organisations and is, thus, from the practical standpoint known by the practitioners. However, building on the previous issues, topics 4-6 address issues that may be considered as new information for the academia. These topics mainly provide the answers to the research questions presented in the beginning of this thesis.

6.1 Answer to the Research Question

The overall research question of this work has been: “What kind of information management system can provide support for user-oriented product development?” In addition to the general question, following specifying questions that focus on each part of the main question were presented:

1. What is user-oriented product development? What is position and relation of user-oriented, or user-centred, activities to other product development activities and tasks? How is user-oriented product development defined and arranged in practice?
2. What is information management that supports the activities of a development organisation? How can information support be arranged in a contemporary product development environment?
3. What kind of services should an information support environment for user-oriented development provide? What is an appropriate structure and technical realisation of an information support system that is feasible to implement and test in contemporary product development environments?

The definitions to the supporting key concepts are presented in the respective parts earlier in this thesis. A summary of the answers to the restricting boundary questions is summarised here shortly:

User-oriented product development (question 1) is

- Multi-disciplinary and functionally crossing development activity
- conducted by several stakeholders (sometimes at different locations)
- that utilise or have explicit or/and implicit information about users
- that are gathered around a shared development goal and problem (product functionality that is relevant to specific use situation, or “context-of-use”, and understandable by the user) in order to solve it
- by taking into account different aspects of the product under development (technological, commercial, social; in the case of user-centred especially issues like usability) and
- the design work itself (processes and procedures for user-centredness).

Development work is conducted in projects that are often guided by quality systems that define required development processes and provide instructions about appropriate working styles (e.g. methods, document templates).

User-oriented development work does not, however, get enabled without appropriate supporting environment. Support for the user-centred development work comes from two directions:

- *organisational environment* provides the “overall attitude” towards different “design for x” approaches and

- the *supportive tools* provide practical means for acting within the desired framework.

Supportive tools consist of process guidance, methodological instructions as well as of the practical tooling for the daily development work. From the standpoint of user-centred development work, examples of such objects are the SPICE-compatible category of human-centred development processes (HCD), guidelines for conducting and reporting a usability study, and the design communication environment for structured management of important issues (like the Knowledge Storage).

The Knowledge Storage answers to the main research question about information support by providing a *conceptual model* that defines user-centred product development as activity that

- consists of many development projects and
- is performed by different roles (stakeholders with appropriate skills) that
- are assigned to be responsible of the production and maintenance of certain information areas like user-centred issues.
- Development work is conducted according to a specific process model that takes into account user-centred issues.

The technical implementation (the web-based prototype) provides an answer to the questions about contemporary ways to provide support and structure of an information support system and mechanism. In the conceptual level (like presented in the description of the model of the Knowledge Storage, chapter 5.3), the research question has also been answered. We also need to evaluate the answer from the practical standpoint, feasibility. The results from the pilot evaluation of Knowledge Storage provide means for this.

6.2 Recent and Future Work

Despite the problems in the very first pilot study, the Knowledge Storage concept was not considered to be useless. The participants of the pilot study made a remark that in slightly different (organisational) conditions the concept and even the prototype might have been useful. In the project that it was applied, string demands for improved information management mechanisms did not exist after all.

An important possibility for future comes from the concept *knowledge item.* Current practices in product development are focused in documenting things in large continuous written documents. Current trends are taking practices towards structured documentation in which documents are constructed on the fly based on meta-information that is available in the document. One of the promising techniques in this field seems to be XML. The use of XML enables documents to be constructed from small parts that can be independent from each other in the writing stage. The construction of knowledge storage presented in this work does not make use of XML but it provides similar possibilities that could be transformed toward this technology. User-centred issues can be tagged in XML representation so that whenever someone requests for user-centred information, the document storage can look for all pieces of information that have been marked to contain information about users. This way the organisation can keep up-to-date with user-centred issues.

One important possibility for future improvements of Knowledge Storage includes the automatic saving of developers' *context information* alongside the saved documents and information items. In product development, context information may consist of the detection of role, project and phase just like in the Knowledge Storage concept. The system may just work in a way that it does not require the selection and especial specification of these parameters every time a knowledge item is saved into the system. If the main hindrance concerning to additional overhead in the storing of knowledge items can be overcome, the situation can be seen to change significantly and Knowledge Storage may replace some of the existing information storages in development organisations. This provides possibilities for future development of the concept and especially the current implementation.

Part of the work that has been presented in this thesis has already gotten continuation in a couple of research projects. One topic that did not receive enough attention in this work was the more detailed modelling of user interface related objects, artefacts and documentation structures within product development. In order to make user interface design more systematic, there is still room for defining these user-centred design artefacts and their management. In a research project that had its focus in the improvement of user interface design for small handheld devices, an integrated user interface specification, design, documentation, and communication environment was created.

Papers about this continued research and the resulting *User Interface Specification Browser (UISB)* have been published at the international academic forum in 2003 (Nieminen & al. 2003a and 2003b). As this research direction goes further, we will be able to manage and design user-centred product features in a globally operating virtual development organisation that can better take into account local user characteristics.

References

Abecker, A., Bernardi, A., Hinkelman, K., Kühn, O., Sintek, M. (1998): Toward a Technology for Organisational Memories. In: IEEE Intelligent Systems. Volume: 13 , Issue: 3 , May-June 1998. Pp:40 – 48.

Applegate, L.M. (1991): Technology Support for Cooperative Work: A Framework for Studying Introduction and Assimilation in Organizations, J Organizational Computing, Vol 1, 11-39, 1991.

Bannon, L. (1993): CSCW: An Initial Exploration. Scandinavian Journal of Information Systems, vol 5 (1993). http://iris.informatik.gu.se/sjis/vol5/bannon.shtml

Bannon, L. & Bødker, S. (1997): Constructing Common Information Spaces. In: Proceedings of the ECSCW'97, the 5th European Conference on Computer Supported Cooperative Work. 7-11 September 1997, Lancaster, UK. Paper available at http://www.ul.ie/~idc/library/papersreports/LiamBannon/ECSCW.htm (checked 9-Jul-2004)

Bennett (1978): Incorporating Usability into System Design. In: Proceedings of the 1978 IEEE International Conference on Cybernetics and Society. Cited from Smith & al. (1980).

Bennett (1979): The commercial impact of usability in interactive systems. In: Shackel, B. (Ed.), Man/computer communication: Infotech state of the art report, Volume 2, 1-17. Maidenhead, UK: Infotech International (source: http://www.hcirn.com/ref/refb/benn79.php, checked 8-Jul-2004).

Bennett, J. (1984): Managing to meet usability requirements: Establishing and meeting software development goals. In Bennett, J. L., Case, D., Sandelin, J., & Smith, M.

(Eds.), Visual display terminals: Usability issues and health concerns, 161-184. Englewood Cliffs, NJ: Prentice-Hall. Cited from Good & al. (1986).

Bentley, R., Horstmann, T. & Trevor, J. (1997a): The World Wide Web as enabling technology for CSCW: The case of BSCW. CSCW Group, Institute for Applied Information Technology (GMD FIT). German National Research Centre for Computer Science.

Bentley R., Appelt W., Busbach U., Hinrichs E., Kerr D., Sikkel K., Trevor J. and Woetzel G. (1997b): Basic support for cooperative work on the World Wide Web. In International Journal of Human–Computer Studies (1997) 46, 827–846

Bevan, N. & Azuma, M. (1997): Quality in use: incorporating human factors into the software engineering lifecycle. Software Engineering Standards Symposium and Forum, 1997. Emerging International Standards. ISESS 97.

Beyer, H. & Holtzblatt, K. (1998): Contextual Design: Defining Customer-Centered Systems. Morgan Kaufmann Publishers ISBN 1-55860-411-1. 472 p. San Francisco. 1998.

Boehm, Barry W. (1988) A Spiral Model of Software Development and Enhancement. IEEE Computer, May 1988.

Booth, P.A. (1989): An Introduction to Human-Computer Interaction. Lawrence Erlbaum Associates. ISBN 0-86377-122-8. 268 p. London 1989.

Borman, L. (1996): SIGCHI: The Early Years. SIGCHI Bulletin, Vol.28 No.1, January 1996.

Boy, G. (1997): Active Design Documents. Proceedings of the conference on Designing interactive systems: processes, practices, methods, and techniques. Amsterdam, The Netherlands. Pages: 31 – 36. ISBN:0-89791-863-0. 1997

Boy, G. (1998): Cognitive Function Analysis. Volume 2 in the series Contemporary Studies in Cognitive Science and Technology. ISBN 1-56750-377-2. Ablex Publishing Corporation. London. 1998.

Boy, G: **(1998)**: Cognitive Function Analysis for Human-Centered Automation of Safety-Critical Systems. Proceedings of the CHI'98 conference, Los Angeles.

Card, S. K., Moran, T. P., & Newell, A. (1983): The psychology of human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Associates.

Chin, J.P., Diehl, V.A. & Norman, K.L. (1988): Development of an Instrument Measuring User Satisfaction of the Human-Computer Interface. In: CHI'88 Human Factors in Computing Systems, Conference Proceedings. (May 15-19-, 1988, Washington, D.C.). Special issue of the SIGCHI Bulletin. Association for Computing Machinery 1989.

Curtis, B., Krasner, H., Iscoe, N. (1988): A Field Study of the Software Design Process for Large Systems. Communications of the ACM 31(11): 1268-1287, 1988.

Curtis, B., Kellner, M. & Over, J. (1992): Process modeling. In *Communications of the ACM*, September 1992. Vol. 35, No. 9. Pp. 75-90.

Davis, J., Subrahmanian, E., Konda, S., Granger, H., Collins, M., & Westerberg A. (1999): Creating Shared Information Spaces to Support Collaborative Design Work. Technical Report 2/99. Decision Systems Laboratory, Institute of Mathematical Modeling and Computational Systems. University of Wollongong.

Dieng, R., Corby, O., Giboin, A., Ribiére, M. (1999): Methods and tools for corporate knowledge management. In: International journal of Human-Computer Studies 51:567-598. Academic Press. 1999.

Dix, A., Finlay, J., Abowd, G. & Beale, R. (1993): Human Computer Interaction. Prentice Hall. 570 p.

Duncan, A.S. & Beabes, M.A. (1995): Contextual Inquiry: Grounding Your Design in Users' Work. Tutorial Notes. Conference on Human Factors in Computing Systems. May 7-11 1995. Denver, Colorado, USA. ACM/SIGCHI.

Durstewitz, M. (1999): Computer Assistance in Aircraft Design. Position paper for the CHI-Workshop: HCI in domains. University of Kassel, Human-Machine Systems Laboratory.

Durstewitz, M. (2001): Experience-Based Decision Support in Aircraft Design and Engineering. PhD thesis. Fachbereich 15 – Maschinenbau. Universität Gesamthochschule Kassel

Earthy, J. **(1998)**: Usability Maturity Model: Human Centredness Scale (INUSE Deliverable D5.1.4s), by J. Earthy (1998). Downloadable from http://www.lboro.ac.uk/eusc/guides/d514S_1c.doc. (checked 8-Jul-2004)

Earthy, J. **(1999)**: Usability Maturity Model: Processes. TRUMP version. Version 2.2. Date 1999/08/19. Downloadable from http://www.lboro.ac.uk/eusc/guides/tr_ump_c.doc. (checked 8-Jul-2004)

Eason, K.D. (1984): Towards the Experimental Study of Usability. Behaviour & Information Technology, 3(2). Pp. 133-143.

Encyclopaedia Britannica. "philosophy" Encyclopædia Britannica Online. <http://search.eb.com/bol/topic? eu=61236&sctn=1> [Accessed 15 July 1999]. Encyclopædia Britannica, Inc. 1994-1999.

Finnish Federation of Electrical and Electronics Industry (1994): Käytettävyys (Usability). Sähkö- ja elektroniikkateollisuusliitto. Helsinki 1994.

Gomoll, K. & Nicol, A. (1990): User Observation: Guidelines for Apple Developers. Apple Human Interface Notes #1, January 1990.

Good, M., Spine, T. M., Whiteside, J., & George, P. (1986): User-derived impact analysis as a tool for usability engineering. Proceedings of CHI 86, 241-246. New York, NY: ACM.

Gould, J. D. & Lewis, C. (1985): Designing for usability: key principles and what designers think, Communications of the ACM 28(3), 300–311.

Gould, J. D. (1988): How to design usable systems. In Helander, M. (Ed.), Handbook of human-computer interaction, 757-789. Amsterdam, The Netherlands: North-Holland. Excerpt reprinted in Baecker, R. M., Grudin, J., Buxton, W. A. S., & Greenberg, S. (1995), Readings in human-computer interaction: Towards the year 2000, 93-121. 2d ed. San Francisco, CA: Morgan Kaufmann Publishers.

Gould, J. G., Boies, S. J., & Lewis, C. (1991). Making usable, useful, productivity-enhancing computer applications. Communications of the ACM, 34, 1, 74-85.

Grudin, J. (1988): Why CSCW Applications Fail: Problems in the Design and Evaluation of Organisational Interfaces. Proceedings of the 1988 ACM conference on Computer-supported cooperative work. Portland, Oregon, United States. Pp: 85-93. ISBN:0-89791-282-9

Gulliksen, J. (1996): Designing for Usability - Domain Specific Human-Computer Interfaces in Working Life. Doctor's thesis. University of Uppsala.

Hacker, W. (1973) Allgemeine Arbeits- und Ingenieurpsychologie : Psychisce Struktur und Regulation von Arbeitstätigkeiten. VEB Deutscher Verlag der Wissenschaften. 472 p. Berlin 1973.

Herbertsson J. (1999): Enterprise Oriented Design for Manufacture - On the adaptation and application of DFM in an enterprise. Linköping Studies in Science and Technology – Dissertations No. 584. Abstract available at http://www.bibl.liu.se/liupubl/disp/disp99/tek584s.htm (checked 8-Jul-2004)

Holzblatt K. & Beyer, H. (1995): Contextual Design: Using Customer Work Models to Drive System Design. Tutorial Notes. Conference on Human Factors in Computing Systems. May 7-11 1995. Denver, Colorado, USA. ACM/SIGCHI.

Howard, R., (1988). Panel Remarks : CSCW: What does it mean?' (Moderatror: L. Bannon). In: Proceedings of CSCW '88. Portland, September, cf. Bannon 1993.

Hix, D. & Hartson, H.R. (1993): Developing User Interfaces. Ensuring Usability Through Product & Process. John Wiley & Sons, Inc.

Humburg, J., Rosenbaum, S. & Ramey, J. (1996): Corporate Strategy and Usability Research: A New Partnership. In: CHI'96 Conference companion on Human factors in computing systems: common ground. Vancouver, British Columbia, Canada. P. 428. ISBN:0-89791-832-0. ACM Press New York, NY, USA 1996.

Humphrey, W.S. (1996): Using a defined and measured Personal Software Process In: IEEE Software, Volume: 13 , Issue: 3 , May 1996. Pp:77 – 88.

ISO (1996): Software Process Assessment. Part 2: A Reference Model for Processes and Process Capability. 26-July-1996. ISO/IEC/JTC/SC7/WG10/N102. Working draft for information purposes only.

ISO 9241-11 (1996): Ergonomic requirements for office work with visual diaplay terminals (VDTs) - Part 11 Guidance on usability. Draft International Standard.

ISO/FDIS 13407 (1999), Human-centred design processes for interactive systems. ISO/TC159/SC4. Final Draft. International Standard. 1999.

Jaakkola, J. & Tunkelo, E. (1987): Tuotekehitys - ideoista markkinoille. Weilin+Göös, Espoo 1987. ISBN 951-35-4166-8. 250 p.

Johnson, P. (1992): Human-Computer Interaction. Psychology, Task Analysis and Software Engineering.McGraw-Hill International UK. London 1992. ISBN 0-07-707235-9. 217p.

Järvinen, J. & Järvinen, A. (1996): Tutkimustyön metodeista. ISBN 951-97113-1-7. Opinpaja Oy, Tampere 1996.150 p.

Kangasluoma, M. (1979): Tutkimus- ja tuotekehityskäsikirja. Kustannus Oy Infopress. Gummerus, Jyväskylä 1979. ISBN 951-737-073-3. 300 p.

Kappes, S. & Thomas, B. (1993): A Model for Knowledge Worker Information Support. Knowledge Worker Information Management. USACERL Technical Report FF-93/10. September 1993. (http://www.cecer.army.mil/kws/kap_supp.htm)

Kaptelinin, V., Kuutti, K., Bannon, L. (1995): Activity Theory: Basic Concepts and Applications. In Blumenthal et al. (Eds.) Human-Computer Interaction. Lecture Notes in Computer Science. Springer, 1995. Cited from Kaptelinin & Nardi 1997.

Kaptelinin, V. & Nardi B. A. (1997): Activity Theory: Basic Concepts and Applications. CHI 97 Electronic Publications: Tutorials. http://www.acm.org/sigchi/chi97/proceedings/tutorial/bn.htm (Cheked 7-Jul-2004)

Kasanen, E., K. Lukka and A. Siitonen. (1993): The constructive approach in management accounting. In: Journal of Management Accounting Research (5): 243-264.

Keinonen, T., Nieminen, M., Riihiaho, S. & Säde, S. (1996), Designing Usable Smart Products. Helsinki University of Technology, Department of Computer Science. Report TKO-C81. Otaniemi, 1996.

Klein, M.R. & Methlie, L.B. (1995): Knowledge-based Decision Support Systems with Applications in Business. John Wiley & Sons. ISBN 0-471-95295-8. New York 1995.

Koivunen, M., Nieminen, M. & Riihiaho, S. (1996): Usability defects in Smart Products. Helsinki University of Technology, Technical Report TKO-B131, 1996. 13 p.

Kosonen, K., Buhanist, P., Kesäjärvi, S., Kymäläinen, P., Lehtonen, T., Salonen, J. & Tanskanen, T. (1998): Muutoksen etulinjassa. ISBN 952-91-0240-2. Hämeenlinna, 1998.

Kotter J. P. (1996): Leading Change. Harvard Business School Press, Boston, MA 1996. 0-87584-747-1. 187 s. 1996.

Kulvik, H. (1977): Uusien tuotteiden onnistumiseen tai epäonnistumiseen vaikuttavat tekijät. Helsingin teknillinen korkeakoulu, tieteellisiä julkaisuja 60, Espoo 1977.

Kuutti, K. (1996): Activity theory as potential framework for human-computer interaction research. In Nardi, B., (ed): Activity theory and human-computer interaction. Cambridge. Mass.: The MIT Press, pp. 17-44.

Kuutti, K., Jokela, T., Nieminen, M. & Jokela, P. (1998): Assessing Human-centred Design Processes in Product Development by using the Inuse Maturity Model. In: 7th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems, Kyoto, Japan September 16-18, 1998.

Lee, J. (1997): Design Rationale Systems: Understanding the Issues. IEEE Expert Vol. 12 May/June 1997. pp.78-85, 0885-9000/97. 1997

Lewis, C. (1982): Using the "thinking-aloud" method in cognitive interface design. IBM Research Report RC 9265. Yorktown Heights, NY: IBM T. J. Watson Research Center. Cited from Lewis & Rieman (1993/1994).

Lewis, C. & Rieman, J. (1993/1994): Task-Centered User Interface Design. A Practical Introduction. A shareware book available at http://hcibib.org/tcuid/. (Checked 7-Jul-2004).

Lubars, M., Potts, C. & Richter, C. (1992): A Review of the State of the Practice in Requirements Modeling. IEEE 0-8186-3120-1/92.

Luhtala, M., Kilpinen, E. & Anttila, P. (1994): LOGI - Managing Make-to-Order Supply Chains. Industrial Economics and Industrial Psychology. Helsinki University of Technology. Report no 153. Otaniemi 1994.

Maguire, M. (2000): A 15 year path of usability development in Europe. In: CHI '00 extended abstracts on Human factors in computer systems (Interactive Posters), The Hague, The Netherlands. ISBN:1-58113-248-4. ACM Press, New York, USA, 2000. Pp. 195-196.

Masters, S. & Bothwell, C. (1995): *CMM Appraisal Framework* , version 1.0. Technical Report CMU/SEI-95-TR-001. ESC-TR-95-001. February 1995. Software Engineering Institute, Carnegie Mellon University. Pittsburg, Pennsylvania, USA.

Moran, T.P. & Carroll, J.M. (1996): Overview of Design Rationale. In: Moran, T.P. & Carroll, J.M. (eds.). Design Rationale. Concepts, Techniques, and Use. Lawrence Erlbaum Associates, Publishers. Mahwah, New Jersey. 1996. Pp.1-20.

Muller, M.J., Halswanter, J.H. & Dayton, T. (1997): Participatory Practices in the Software Lifecycle. In: Helander, M., Landauer, T.K. & Prabhu, P. (ed.): Handbook of Human-Computer Interaction. Second, completely revised edition. Elsevier science, B.V. ISBN 0-444-81862-6. Pp. 255-298. Amsterdam, 1997.

Munro, A. J., Höök, K. & Benyon D. (Eds.) (1999): Social navigation of information space. London. Springer Verlag. ISBN 1-85233-090-2. P. 277. 1999.

Nielsen, J. (1993): Usability Engineering. Academic Press. London 1993. ISBN 0-12-518405-0. 358 s.

Nieminen, M. & Koivunen, M. (1995): Visual Walkthrough. In: Allen, G., Wilkinson, J. & Wright, P. (ed): HCI'95, People and Computers. Adjunct Proceedings. Conference at the University of Huddersfield 29 August - 1 September 1995. The School of Computing and Mathematics. The University of Huddersfield.

Nieminen, M. Keinonen, T., Riihiaho, S. & Säde, S. (1996): Usability of Work Tools in Information Society. In: Abstracts of the Work in the Information Society Conference. 20.-22. May 1996. Helsinki: Finnish Institute of Occupational Health. Pp 188-191.

Nieminen, M. (1996): Usability in Product Development Process. In: Keinonen, T., Nieminen, M., Riihiaho, S. & Säde S. (toim.), Designing Usable Smart Products. Espoo 1996, HUT, Faculty of Information Technology, Department of Computer Science, Pp. 29-47.

Nieminen, M. & Parkkinen J. (1998): Usability activities in product development. In: Vink, P., Koningsveld A.P.E., Dhondt, S. (ed.). Human Factors in Organizational Design and Management - VI. Proceedings of the sixth international symposium on human factors in organizational design and management. The Hague, The Netherlands, August 19-22,1998. ISBN: 0-08-04349-8. Pp.433-438. Elsevier / North-Holland 1998.

Nieminen, M. (1999): Elements of "Design for Usability" in Product Development. Annual Research Conference "SAICSIT 1999: Prepare for the New Millenium, Is the life after y2k? Mount Amazi Lodge, Hartebeespoort South Africa 17.-19.11.1999. Johannesburg 1999, SAICSIT: South African Institute of Computer Scientists and Information Technologists. Electronic proceedings.

Nieminen, M. (2001): Managing Human-Centered Design Artifacts in Distributed Development Environment With Knowledge Storage. In: Smith, M.J, Salvendy, G., Harris D., Koubek, R.J. (ed.): Usability Evaluation and Interface Design. Cognitive Engineering, Intelligent Agents and Virtual Reality. Volume 1 of the proceedings of HCI International 2001. ISBN 0-8058-3607-1. Pp. 988-992. Lawrence Erlbaum Associates, Publishers. 2001.

Nieminen, M., Koskinen, T., & Johnson, M. (2003a): UISB – The User Interface Specification Browser.10th International Conference on Human - Computer Interaction. Crete, Greece, June 22-27, 2003.

Nieminen, M., Johnson, M., Parkkinen, J., & Koskinen, T. (2003b): Evaluation of UISB - The User Interface Specification Browser. IEA 2003. The International Ergonomics Association XVth Triennial Congress.

Nonaka, I. & Takeuchi, H. (1995): The Knowledge-creating Company: How Japanese Companies Create the Dynamics of Innovation. ISBN 0-19-509269-4. Oxford University Press. New York 1995. p. 284.

Norman, D.A. & Draper S.W. (ed.) (1986): User-Centered System Design. New Perspectives on Human-Computer Interaction. Lawrence Erlbaum Associates, Publishers. London 1986. ISBN 0-89859-781-1. 526 p.

Norman, D. A. (1988): The Psychology of Everyday Things. Basic Books. New York. (Subsequently published under the title: The design of everyday things)

Nyström, T. (1997): Comparison of Software Reference Processes Definitions. Master's thesis, Helsinki University of Technology, Department of Information Processing Science.

Ngwenyama, O. & Nielsen, P.A. (2003): Competing values in software process improvement: an assumption analysis of CMM from an organizational culture perspective. In: IEEE Transactions on Engineering Management, Volume: 50 , Issue: 1 , Feb. 2003. Pp:100 – 112

O'Hara, F. (2000): European experiences with software process improvement. In: International Conference on Software Engineering archive Proceedings of the 22nd international conference on Software engineering, Limerick, Ireland. Pp: 635 - 640. ISBN:1-58113-206-9

Polson P.G., Lewis C., Rieman J. & Wharton C. (1992): Cognitive Walkthroughs: A Method for Theory-Based Evaluation of User Interfaces, International Journal of Man-Machine Studies, vol. 36, no. 5, pp. 741-773, 1992.

Pressman, R.S. (1987): Software engineering: a practitioner's approach. Second edition. ISBN 0-07-100232-4. 567 p. McGraw-Hill International Editions, New York 1987.

Ravden, S.J. & Johnson, G.I. (1989): Evaluating Usability of Human Computer Interfaces: A Practical Method. Ellis Horwood Limited, Halsted Press.

Redmiles, D. (2002): Introduction to the Special Issue of CSCW on Activity Theory and the Practice of Design, Computer-supported Cooperative Work, Special Issue on Activity Theory and the Practice of Design, Vol. 11, No. 1-2, 2002, pp. 1-11. See (http://www1.ics.uci.edu/%7Eredmiles/activity/final-issue.html, checked 8-Jul-2004).

Roe, R.A. & Meijer, T. (1990): Action facilitation and mental information work. In Richter, P & Hacker, W. (Eds.) Mental work and automation, Proceedings of the 6th International Symposium on Work Psychology, March 27.-29. 1990. Stoba-Druck, Dresden.

Rosenbaum, S., Rohn, J., Humburg, J., Bloomer, S., Dye, K., Nielsen, J., Rineheart, D. & Wixon, D. (1999): What Makes Strategic Usability Fail? Lessons Learned from the Field. In: CHI '99 extended abstracts on Human factors in computing systems. Pp. 93-94. ISBN: 1-58113-158-5. ACM Press, New York, NY, USA 1999.

Rosenbaum, S., Rohn, J.A., Humburg, J. (2000): A toolkit for strategic usability: Results from workshops, panels, and surveys, in the proceedings of CHI'2000, Amsterdam, (2000), 337-344.

Rosenbaum, S., Wilson, C.E., Jokela, T., Rohn, J.A., Smith, T.B., Vredenburg, K. (2002): Usability in Practice Session: Usability in Practice: user experience lifecycle - evolution and revolution. In: CHI '02 extended abstracts on Human factors in computing systems table of contents. Minneapolis, Minnesota, USA SESSION: Usability in Practice. Pp. 898 – 903. ISBN: 1-58113-454-1. ACM Press New York, NY, USA 2002.

Rosson, M.B., Maass, S. & Kellogg, W.A. (1987): Designing for designers: An analysis of Design Practice in the Real World. In: Proceedings of the CHI'87 conference. ACM 1987.

Ryder, M. (2004): A collection of web links and resources about activity theory. Last update 1-Jul-2004. http://carbon.cudenver.edu/~mryder/itc_data/activity.html (Checked 8-Jul-2004).

Schmidt, A., Beigl, M., Gellersen, H-W. (1999): There is More to Context than Location. In: Computers and Graphics 23:893–901.

Schmidt, K., (1991): Riding a Tiger, or Computer Supported Cooperative Work. In: L. Bannon, M. Robinson & K. Schmidt, editors. Proceedings of the Second European Conference on CSCW. Dordrecht, Kluwer. Pages 1-16.

Schmidt, K. (1997): Of Maps and Scripts : The Status of Formal Constructs in Cooperative Work, Proceedings of GROUP97, Phoenix, Arizona, 1997, pp.138-147.

Shackel, B. (1986). Ergonomics in design for usability. In Harrison, M. D., & Monk, A. F. (1986), People and computers: Designing for usability, 44-64. Proceedings of HCI 86. Cambridge, UK: Cambridge University Press.

Sharp, H., Finkelstein, A. & Galal, G. (1999): Stakeholder Identification in the Requirements Engineering Process. 10th International Workshop on Database & Expert Systems Applications, Florence, Italy, 1-3 September, 1999, Proceedings. IEEE Computer Society, 1999, ISBN 0-7695-0281-4

Shneiderman, B. (1988): Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley.

Smith, S.L. & Mosier, J.N. (1986): Design Guidelines for Designing User Interface Software. The MITRE Corporation. Technical Report MTR-10090. These guidelines are available at http://hcibib.org/sam/ (checked 20-May-2003).

Smith, G.L., Stephens, S.A., Tripp, L.L. & Warren W.L. (1980): Incorporating usability into requirements engineering tools. In: Proceedings of the ACM 1980 annual conference, January 1980. ISBN:0-89791-028-1.

Sommerville (1985): Software engineering. Second edition. Addison-Wesley. ISBN 0-201-14229-5. Wokingham 1985. 334 p.

Steves, M. P. & Scholtz, J. (1998): Modified Field Studies for CSCW Systems. Available at www.nist.gov/itl/iad/IADpapers/modified_cscw.pdf (checked 3-June-2003).

Takalo, J., Taramaa, J., Savolainen, P., Partanen, J. (2000), "Experiences of Distributed Product Data Management of Electro-Mechanical Products in Multisite Organisation", Proceedings of the 3rd International Symposium on Tools and Methods of Competitive Engineering, April 2000, Delft, Netherlands, 2000, pp 217-224. (Abstract available at http://www.io.tudelft.nl/tmce2000/proceedings/abstractsDMC.htm; checked 8-Jul-2004).

Tamminen, S., Oulasvirta, A., Toiskallio, K. & Kankainen, A. (2004): Understanding Mobile Contexts. In: Personal and Ubiquitous Computing 8: 135-143. 2004.

Teikari, V., Pulkkis, A. & Vartiainen, M. (1985) Taitavan työsuorituksen oppimistutkimusta yhteistyössä. Strömbergin konepajakoulu. Helsinki (in Finnish).

Tiako, P.F., Lindquist, T. & Gruhn, V. (2001): Process support for distributed team-based software development workshop. ACM SIGSOFT Software Engineering

Notes archive Volume 26, Issue 6 (November 2001). Pp. 31 – 33. ISSN: 0163-5948. 2002.

Toivonen, S., Vuori, M. & Pekkarinen, M. (1996). Yhteenveto kesällä ja syksyllä 1995 tehdyistä "käytettävyyskyselyistä" pk-yrityksille. VTT Valmistustekniikka. http://www.vtt.fi/aut/rm/val45/usabil2/kysely.tr2/tr2w.htm

Toye, G., Cutkosky, M.R., Leifer, L.J., Tenenbaum, M.J., Glicksman, J. (1994): SHARE: A Methodology and Environment for Collaborative Product Development. In: International Journal of Intelligent and Cooperative Information Systems, 1994. See http://www-cdr.stanford.edu/ONR/EndofYearSummary96.html

Ulrich, K.T. & Eppinger, S.D. (1995): Product Design and Development. McGraw-Hill, New York 1995. ISBN 0-07-065811-0. 289 p.

Valjus, P. (1994): Limittäinen tuotekehitys valtaa alaa. In: Tekniikan näköalat 1/1994 (18.1.1994), TEKES, Helsinki. Pp. 26-27. ISSN 0782-7350.

Vartiainen, M., Pulkkis, A., Kasvi, J.J.J. & Nieminen, M. (1996): The critical job and organizational demands for the design of Information Support Systems. In: Hacker, W., Richter, P. & Wagner, T. (eds.): Analysis and Design of Mental Work. IV. Workshop TU Helsinki - TU Dresden. Dresden, December 1995. Dresden University of Technology. Institute of Cognitive & Motivational Psychology amd Methods of Psychology; Institute of Work, Organizational and Social Psychology. Pp. 19-36.

Vasconcelos, J., Kimble, C., Gouveia, F., Kudenko, D. (2000): Group Memory System for Corporate Knowledge Management: An Ontological Approach. In: Proceedings of the 1st European Conference on Knowledge Management (ECKM'2000). Bled School of Management, Slovenia, October 2000, ISBN: 0 9510066 4 9, pp. 91-99.

Vredenburg, K., Mao, J-Y., Smith, P.W. & Carey, T. (2002): A Survey of User-Centered Design Practice. Proceedings of the SIGCHI 2002 conference on Human factors in computing systems: Changing our world, changing ourselves. Session: Design Methods table of contents. Pp. 471 - 478. ISBN:1-58113-453-3. Minneapolis, Minnesota, USA. 2002.

Ward, R.P., Fayad, M.E. & Laitinen, M. (2001): Software Process in the Small – A small software development company's most difficult challenge: changing processes to match changing circumstances. In: Communications of the ACM. April 2001, Vol. 44, No. 4.

Weiser, M. (1993): Some computer science issues in ubiquitous computing. In: Communications of the ACM. Volume 36 , Issue 7 (July 1993). Special issue on computer augmented environments: back to the real world. Pp: 75 – 84. ISSN:0001-0782. 1993.

Whiteside J., Bennett J. & Holzblatt K. (1988): Usability Engineering: Our Experience and Evolution. In Helander, M. (ed.): Handbook of Human-Computer Interaction, 1988.

Whitten, N. (1990): Managing Software Development Projects. Formula for Success. John Wiley & Sons, New York 1990. ISBN 0-471-51255-9. 276 p.

Yli-Olli, P. & Hokkanen, T. (1991): Softa 9000. Ohjelmistotuotannon laadun kehittäminen. Mecrastor Oy. ISBN 951-96282-0-7. 154 p.

Unpublished References

Rauterberg M. (1991): Participatory Concepts, Methods and Techniques for Optimising Software Development. Lecture material for "The Analysis and Development of Human-Information Systems" 18.-19.11.1991, Helsinki University of Technology. In: Brödner, P. & Simonis, G. (1992): Work Design and Participatory System Development. 28 p.

Appendices

Appendix A. Background information questionnaire to the participants of the piloting of the Knowledge Storage Prototype.

Appendix B. Pilot case introduction paper for participants.

Appendix C. Contents of the User Environment model of Knowledge Storage.

Appendix D. The list of questions about product development that were presented to product developers.

Appendix A. Background information questionnaire to the participants of the piloting of the Knowledge Storage Prototype.

KATTI • • • • • • •

Käyttäjäkeskeisen tuotekehityksen tietotuki

Information Support for User Centred Development

KATTI Knowledge Storage Pilot Project

Marko Nieminen , Ismo Hautala
Helsinki University of Technology
Usability Research Group

Questionnaire for
KS 2000 Pilot Project
participants

This questionnaire is being used to gather background information about the tasks and activities of the participants of current project. The information will be used to support the current state analysis and analysis at the end of the project. Using the results of this questionnaire we verify that the work conditions and problems experienced in the work environment are similar to those that were detected in the interviews that have been conducted in Helsinki.

Basic information

Position in organisation (e.g. developer, project manager, product manager, marketer)

____

Current projects and responsibilities in them:

Project	**Responsibilities and tasks in the project**

Information flow

1. How often do you use following systems? (Write your answer between the parentheses)

1 = never
2 = seldom
3 = monthly
4 = weekly
5 = daily

( ) Web browser
( ) In-house problem database
( ) Support database
( ) Right Now
( ) Intranet
( ) Web Community
( ) Web Documents Folders
( ) Web Discussion Groups
( ) Web Interactive Business Cards
( ) Office Mail
( ) Office Calendar
( ) Office Forum
( ) Office Library
( ) Web Service Mail
( ) Web Service Calendar
( ) Web Service Forum
( ) Web Service Library

( ) Word
( ) Excel
( ) PowerPoint
( ) FrontPage / other HTML editor, what?
____
( ) Programming tools (C++, Java, VB…) , what?
____
( ) Graphics tools, what?
____
( ) Other:
____

2. How easy is it to store, retrieve and find information and documents? (Write your answer between the parentheses)

1 = very difficult
2 = difficult
3 = moderate
4 = easy
5 = very easy
- = I do not use

( ) Storing documents to document library
( ) Storing documents to Web Document Folders

( ) Finding documents from document library
( ) Finding documents from Web Document Folders

( ) Retrieving documents from document library
( ) Retrieving documents from Web Document Folders

( ) Finding information with Knowledge Browser
( ) Finding information from intranet
( ) Finding information with Web Service Library
( ) Finding information with Web Forum
( ) Finding information with document library
( ) Finding information with Forum
( ) Finding information with Web Document Folders
( ) Finding information with Web Discussion Groups

( ) Finding information with Internet newsgroups
( ) Finding information with Internet search engines
( ) Finding information from mail system
( ) Finding information with Search Pro from document library

( ) Discussing professional issues with Forum
( ) Discussing professional issues with Web Discussion Groups
( ) Discussing professional issues with Internet newsgroups
( ) Discussing professional issues with ______________________
( ) Discussing professional issues with ______________________

4. How often do you communicate with different stakeholders? (Write your answer between the parentheses)

1 = never
2 = seldom
3 = monthly
4 = weekly
5 = daily

My current project

( ) Project manager
( ) Product manager
( ) Developers
( ) Validation personnel
( ) Other personnel in my project

Other projects

( ) Project managers
( ) Product managers
( ) Developers
( ) Validation personnel
( ) Other personnel

Other organisational groups

( ) Marketing personnel
( ) Sales personnel
( ) Support personnel
( ) Localisation personnel
( ) Human factors personnel
( ) Business Board personnel
( ) Technical writers

External groups

( ) Customers
( ) End-users
( ) Partners
( ) Sub-contractors

( ) Other: ______________________________
( ) Other: ______________________________
( ) Other: ______________________________
( ) Other: ______________________________

5. Mark with "X" the stakeholders that have explicitly and clearly expressed that they need certain information from you. For each line you can specify the contents of the information.

My current project	**Content**
( ) Project manager	________________
( ) Product manager	________________
( ) Developers	________________
( ) Validation personnel	________________
( ) Other personnel in my project	________________
Other projects	
( ) Project managers	________________
( ) Product managers	________________
( ) Developers	________________
( ) Validation personnel	________________
( ) Other personnel	________________
Other organisational groups	
( ) Marketing personnel	________________
( ) Sales personnel	________________
( ) Support personnel	________________
( ) Localisation personnel	________________
( ) Human factors personnel	________________
( ) Business Board personnel	________________
( ) Technical writers	________________
External groups	
( ) Customers	________________
( ) End-users	________________
( ) Partners	________________
( ) Sub-contractors	________________
( ) Other:	________________
( ) Other:	________________
( ) Other:	________________
( ) Other:	________________
( ) Other	________________

6. Mark with "X" the stakeholders to whom you have have explicitly and clearly expressed what information you need from them. For each line you can specify the contents of the information.

My current project		**Content**
( )	Project manager	________________
( )	Product manager	________________
( )	Developers	________________
( )	Validation personnel	________________
( )	Other personnel in my project	________________
Other projects		
( )	Project managers	________________
( )	Product managers	________________
( )	Developers	________________
( )	Validation personnel	________________
( )	Other personnel	________________
Other organisational groups		
( )	Marketing personnel	________________
( )	Sales personnel	________________
( )	Support personnel	________________
( )	Localisation personnel	________________
( )	Human factors personnel	________________
( )	Business Board personnel	________________
( )	Technical writers	________________
External groups		
( )	Customers	________________
( )	End-users	________________
( )	Partners	________________
( )	Sub-contractors	________________
( )	Other:	________________
( )	Other:	________________
( )	Other:	________________
( )	Other:	________________
( )	Other:	________________

Creation of documents

7. How do you use the documents of your current project? (Write your answer between the parentheses)

R = Read
W = Write
- = I do not use

( ) Business Plan
( ) Strategy Documents
( ) Opportunity Description
( ) Project plan
( ) Quality system documents
( ) Requirement specification
( ) Functional specification
( ) Architectural specification
( ) Product specification
( ) Project status reports
( ) Review status reports
( ) Marketing material (product or general)
( ) Release Notes
( ) Support Bulletin
( ) Project Newsletter (free form status report)
( ) Project meeting memos

( ) White Papers
( ) Validation test reports
( ) Usability test reports
( ) Contextual Design Models

( ) Other: ______________________________
( ) Other: ______________________________

8. How do you use the documents of other projects in your current project? (Write your answer between the parentheses)

R = Read
W = Write
- = I do not use

( ) Business Plan
( ) Strategy Documents
( ) Opportunity Description
( ) Project plan
( ) Quality system documents
( ) Requirement specification
( ) Functional specification
( ) Architectural specification
( ) Product specification
( ) Project status reports
( ) Review status reports
( ) Marketing material (product or general)
( ) Release Notes
( ) Support Bulletin
( ) Project Newsletter (free form status report)
( ) Project meeting memos

( ) White Papers
( ) Validation test reports
( ) Usability test reports
( ) Contextual Design Models

( ) Other: ______________________________
( ) Other: ______________________________

Problems related to work

9. Considering your current work and tasks, do you agree with the following statements? (Write your answer between the parentheses)

1 = I fully disagree
2 = I disagree
3 = I do not know
4 = I agree
5 = I fully agree

( ) 1. Information does not flow well enough between development project groups.
( ) 2. It is difficult to get a clear picture about what other projects and organisational units do.
( ) 3. There is no common language between organisational units (sales, marketing, development etc.)
( ) 4. There are not enough meetings between project groups and between organisational groups.
( ) 5. There is not enough information about customers and end users.
( ) 6. Customer visits are not documented well enough.
( ) 7. Handling of the feedback from customers is difficult
( ) 8. It takes too much time for new employees to get familiar with their work.
( ) 9. When a person leaves, too little of his/her knowledge remains in our organisation.
( ) 10. Development work guidelines are not followed well enough.
( ) 11. There are too many internal information systems.
( ) 12. Internal information systems are not integrated well enough.
( ) 13. It is often hard to find answers to specific questions from the development information systems.
( ) 14. Changes to plans, requirements or specifications are not documented well enough.
( ) 15. Keeping important documents up-to-date requires too much effort.
( ) 16. I get too much information that is not relevant to me right now but would be at some other time.
( ) 17. I have to poll too many information systems to get important information.
( ) 18. I have to interrupt my work too often to answer detailed support-like questions.
( ) 19. I do not know whether the documents I create are read.
( ) 20. I do know well enough whether the documents I create are useful for others.
( ) Other:___________________________________

Appendix B. Pilot case introduction paper for participants.

KATTI • • • • • • •
Käyttäjäkeskeisen tuotekehityksen tietotuki
Information Support for User Centred Development

KATTI Knowledge Storage Pilot Project

Knowledge Storage 2000

- Information Management for the New Era -

Summary. The developers and stakeholders of the development project are offered the possibility to utilise the prototype of the Knowledge Storage. Knowledge Storage is a knowledge management tool that can be used to index all documents and information that has been created with different information systems in a development project. The use of Knowledge Storage is not, however, compulsory. It acts as a front-end to the existing data storages. Thus, developers can continue using existing practices but Knowledge Storage offers new possibilities for information storing and retrieval.

In addition to providing support for development work only, Knowledge Storage appears to offer structures and solutions that can be used in the forthcoming development of the product under development as well.

The development of the Knowledge Storage has been conducted in close co-operation of participants from our company and researchers from Helsinki University of Technology (Usability Group).

Knowledge Storage and the KATTI Research Project

Knowledge Storage 2000 is an improvement project within our company that has its focus on managing development time information from the standpoint of user centred development issues. *KS 2000* is a case project inside a larger research project called "Information Support for User Centred Development - KATTI". KATTI is conducted in cooperation with Helsinki University of Technology and two other research organisations. KATTI belongs to a Finnish national technology programme that is funded by the Finnish Technology Development Centre TEKES and the participating companies.

The goal of the KATTI project is "to develop in a cooperative way processes and methods for collecting, managing and utilising user related information in product development". The project has started in June 1997 and it lasts up to the end of year 1999. Further information about KATTI is available at http://www.interactive.hut.fi/katti.

KATTI Case: Knowledge Storage

The improvement of user centred development practices has resulted in a concept of a *Knowledge Storage*. The Knowledge Storage project *KS 2000* has, characteristics with the current development project that is developing an intranet quality management system. The common elements include quality documents and templates, actors (roles), and process stages. The focus of *KS 2000* is the management of information related to user centred product requirements. The goal of the project is to:

- Create more systematic procedures to support user centred design
- Raise the maturity level of usability planning (framework: Usability Maturity Model UMM)
- Develop and introduce a design rationale process and a generic prototype tool "KS 2000" to support it. Some elements of KS 2000 may be used in the authoring of the system under development.
- To apply Contextual Design methodology to achieve these goals

The KATTI research project is about to begin its third phase, piloting and testing the developed user centred development tools and practices. Current state analysis and tool and work practice development has been conducted during the previous two years. The piloting work will be conducted during the next half a year with the participants of the current project.

Practical Problems: Managing and Sharing Information

The issues that were to be researched were discovered by interviewing 11 persons from different parts of the development organisation in Helsinki. The project aims at answering to four information management related problems that emerged in the interviews:

1. The management of project and product information is difficult
2. Sharing of information is not efficient enough between projects and between organisation units
3. Separate information management systems are hindering free information flow
4. User related information does not reach developers because of non-systematic ways of collecting information and different ways of presenting it

Solution: A Web-accessible Index to Development Time Information

As a solution to the discovered problems the participants of the KATTI project have developed a prototype knowledge management system "Knowledge Storage 2000" and procedures for its effective use.

The solution is a web-based index to development time information (initial step towards design rationale) and a front-end to existing systems. KS 2000 can be used to add describing information about people (roles), products and projects to existing documents. Additionally, KS 2000 contains templates that can be used in various stages of a development project. The prototype provides a method for the user to get feedback about his/her contributions to the rest of the project group and even to the rest of the organisation. A more detailed description of KS 2000 will be delivered soon.

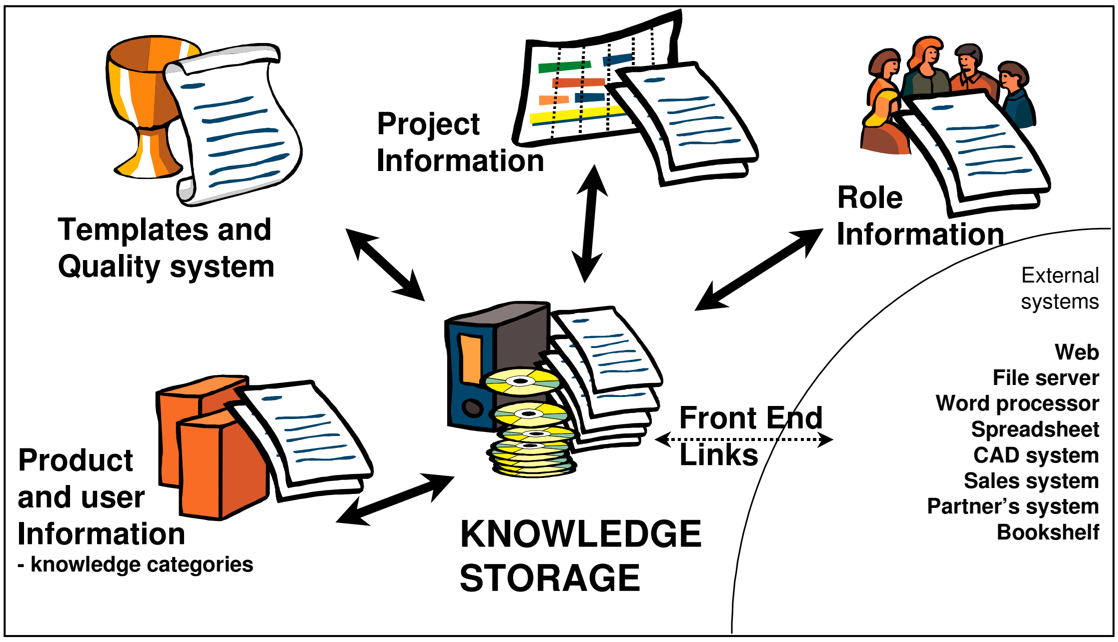
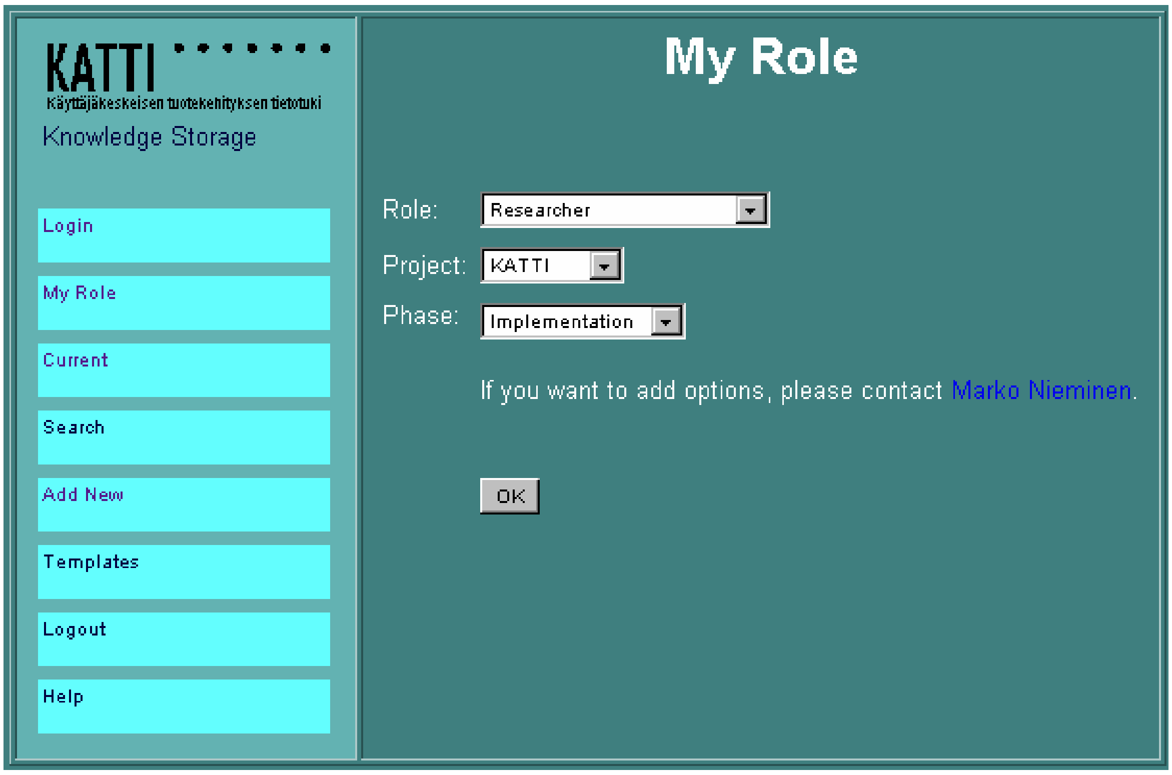


Figure 1. The Knowledge Storage contains information about roles (stakeholders of a development project), project and the product under development. In addition to these, it contains various templates (e.g. quality documents, specification templates, and memorandum skeletons) that can be used throughout the development project. The knowledge storage itself contains links with classifying information to various documents that have been created with external development information systems.

Testing the prototype

The prototype is available for testing at http://www.interactive.hut.fi/ks. You can test it with the user name "katti" and the password "demo". The pilot version of the prototype that will be used by the participants during piloting will be installed in the Intranet for secure access.



Role: Researcher

Project: KATTI

Phase: Implementation

If you want to add options, please contact Marko Nieminen.

Figure 2. Sample screen shot from Knowledge Storage prototype tool. This screen is used to tell the system the current role of the user within the development project.

Forthcoming Events with the Pilot Project

The participants of the pilot project are offered the possibility to use the *KS 2000* prototype. The aim is that the participants have the possibility to apply the prototype easily without too much overhead work to their daily tasks. We want to keep all disturbances, additional work and additional assignments to the minimum. The existing data from the contextual design sessions will support the current state analysis at the beginning of the pilot project.

When the piloting starts, we suggest following activities during the KS 2000 piloting:
1. Questionnaire about current work practices (by mid-June)
2. Initial introduction during Contextual Design training
3. Pilot use can begin at 1st July and it will end at 30th September, 1999. The usage will be monitored and evaluated afterwards. Support will be provided throughout the pilot use.
4. Intermediate evaluation session at the end of August; The vision and goal of the project may change as a result of this evaluation session
5. Reporting the results; This will be conducted by the participating researchers from the Helsinki University of Technology
6. Presenting the results to the pilot users at the end of September, 1999
7. Closing the pilot case and decisions about next steps at the beginning of October, 1999.

Appendix C. Contents of the User Environment model of Knowledge Storage.

Legend:

o This function is responsible for providing this function/service

> This function has one-directional link to another function

>> This function has bi-directional link to another function

*

+

1. Knowledge Storage Database

Stores all knowledge items, forms, documents and index entries and allows users to save, search and retrieve them
o - contains process description applied to current project
o - scans the status (phase) of different processes/projects
0 - maintains history of design issues
>5 download/view template docs
>13 knowledge entry evaluation statistics
>6 Job notification
o - provides users threads of design history
o - maintains a list of reviewers & review process ("quality system")
o - provides access to other design systems (NIPO, Genesis etc.)
o - maintains registry of who has selected KI entries and when and which entries
o - maintains registry of evaluations
o - maintains a list of review meeting participants (roles)
>13 knowledge entry evaluation statistics
>9 knowledge index entry search results
>10 view knowledge index entry
>11 view knowledge entry

2. "Knowledge Index Entry KIE" -form

Allows user to put meta-information about a Knowledge Entry to Knowledge Storage.
o - displays form
* - info type according to classification in the paper by MN & IH(+ Design Idea)
* - targeted project and project phase, pre-project also (11)
* - targeted group / role / meeting
* - where saved
* - keywords and titles, also pre-defined
o - system already knows ID and role of user and also time and phase when saving
* - source of info, reference, e.g. customer
* - upload Knowledge Entry to existing storage (depends on implementations and practices)
* - submit, after submit displays entry in wysiwyg
o - saves KIE to Knowledge Storage
o - context sensitive help
o - uploads document automatically
o - automatically makes summary of knowledge entry
o - system automatically relates new features to customer issues
* - user specifies if the entry is ready to be reviewed (draft status: under construction/ready to be reviewed/approved)
>1 Knowledge Storage Database

* - user defines deadline to comments, location/method of commenting (e.g. discussion group or mailbox), updates reviewer list and specifies document to be reviewed
* - specify the phase of project when the KE ic created or updated
* - enter title of KE, also pre-defined
* - specify the owner of the document

3. View, create and edit text documents
o - allows user to view, create and edit text
o - allows user to save the files
o - allows user to use templates to create documents
o - allows user to view documents and templates
* - create link to design history in Knowledge Storage
+ Requiremennt Specification template etc. (other documents templates, also meeting minutes)
+ requirement documents (etc. other documents)
>>12 evaluate knowledge entry (if opening and viewing knowledge entry from 9 or 10)

4. Create and edit picture
Allows user to create, edit and save pictures.

5. View or download template (e.g. customer info gathering)
Allows selecting, downloading and opening of a template.
o - displays a sorted/grouped/targeted list of template documents
o - displays brief descriptions of templates
* - open template (see link)
* - download template
o - search targeted to templates
>3 create and edit documents
+ Knowledge Item Template KIT (e.g. chapter in product definition, issues included)
+ Knowledge Need Template KNT

6.Job Notification
Gives the user a task to work on.
o - displays a list of related and relevant documents, list contains both input documents (input information for the user) and templates
* - user reads the notification that contains an expiration date
* - user downloads and views input documents and templates
o - display info related to this task e.g. deadline, related discussion group
>3 view, create and edit text documents
>4 view, create and edit pictures
DI: contains an expiration date = review meeting date/deadline. MNi 8.3.1999.

7. Create and edit presentation
Allows user to create, edit and save slides. Allows user to use templates.
* - view presentation
* - edit presentation
+ company presentation template

8. "Search Knowledge Index Entry database" -form
Allows user to specify search criteria and execute search
o - displays form
* - Specify info type categories (see fa 2)
* - specify project phase (targeted in KE)
* - specify target group (targeted in KE)
* - specify keywords (targeted in KE)
* - specify name of enterer (targeted in KE)
* - specify free-text search words
* - role, project and project phase as search criteria: BOTH author's and target's
* - submit
* - search by "this week", "last week", "I have read", "unread"
o - executes search in database
* - clear form
>1 Knowledge Storage database

>9 KIE search results
* - search includes also contents of documents

9. Knowledge Index Entry (KIE) search results
Show matching KIEs in database and allow selection.
o - Displays a list of matching KIEs, sorted/grouped somehow, long list OK (1)
0 - displays a title, name of enterer, date, category (info type), keywords and status
* - sort list by different fields
o - displays title as a link to KIE
o - displays link to KIE directly, if available
* - select one of the listed KIEs
o - registers which entry the user selected
* - modify criteria and search again
>1 Knowledge Storage database
>8 search KIE database form
>10 view knowledge index entry
>3 view knowledge entry
>12 evaluate Knowledge Entry
o - search results may contain enough information ->view may be unnecessary
o - allows choosing what fields are displayed

10. View Knowledge Index Entry KIE
Displays information stored in the KIE (in fa 2).Allow user to evaluate the usefulness of a KE while reading it.
o - displays info with good layout, for readability
o - displays info type category, targeted phase, group, location of Knowledge Entry KE, keywords, source, references, name of enterer, date of entry, status of entry, title of entry and all that is stored in fa 2.
* - open KE if available
o - displays link to evaluation page
>3. view Knowledge Entry if possible
>12 evaluate Knowledge Entry
o - displays question of evaluation and the alternatives
* - select "useful to me now" or "not useful to me now" (usefulness level)
* - submit
* - Allow writing free text comments
* - allow recommendation
* - allow adding new target
* - allow adding of classification, Removing not allowed.
* - Last person who has modified + modification date + time is displayed.
* - usefulness index through usage tracking (1), show how many "useful"-marks are active
o - classify entries personally by folders and deleting
* - mark interesting for oneself
* - link entry to other entries
o - some fields should not be modifiable (date of entry, title)
* - allow a link to author's contact info

>>view text document

11. Knowledge Index Storage Main
Allows user to choose what he/she wants to do with Knowledge Storage.
o - displays options to view and evaluate Knowledge Index Entries
o - ...and edit them
o - ...and createthem
o - ...and search them
o - displays options to list and download templates
o - ...and change current role
o - ...and log out

>15. Define user's project, project phase and role
>2. Knowledge Index Entry Form
>8. Search Knowledge Index Entries
>5. List and download templates
>13. Knowledge Entry Evaluation Statistics

>9. Knowledge Index Entry Search results

12. View Knowledge Entry KE

13. "Knowledge Entry Evaluation statistics" -form
Allows user to request statistics about evaluations of Knowledge Entries.
* - specify criteria: username, project phase, project, documents type, usefulness level, role, individual entry
* - submit
* - Did my recommendations succeed?
o - retrieve and display results
* - view results: title, project manager, usefulness index

o - trend monitoring: how has interest changed lately, graphical presentation
o - overview of project level interests (for new members)
o - limited access,
- statistics is considered important (7)
- prevent negative tracking, "boss is watching"
DI: Feedback reports by: individual entry or document, receiving person, top 10/100 usage
>2 Knowledge Entry Index KIE
>3 Documents
>4 Documents

15. Define user's project, project phase and role
Allows user to define his/her current project, project phase and role.
o - displays user's current status: project, project phase, and user's role NONE ALSO
o - user can accept or change the shown status
o - displays options for projects, phases and roles, if user wants to chang. Role, project and phase may be none or multiple.
* - user selects from options
o - role, project, phase administration
>11. Knowledge storage main.
DI: Allow admin or certain roles to change and add projects, phases and roles

16. Login
o - IP-based authentication (5,4,2)
o - toolbar/menu frame not visible here (3)
o - login possible from intranet

17. Pre-defined keywords -list
o - List should include all new keywords, "Auto-append new keywords"
"Keywords do not work!"(1)

18. Help
* - instructions about the"Knowledge Storage"-concepts and structure (1)
* - instructions about Knowledge Categories (2,3)
* - instructions about roles (4)

19. Progress of Project
o - system allows the possibility to follow the program of the project

20. Current
o - display:s: new / the ones rad last week / recently read (20/11)
* - set serach criteria for "my areas of interest" (20/11)
* - search by keywords
* - set sort criteria

* - classify entries by deleting or by folders
o - sends email notification when new entry arrives
o - choose what fields are displayed (20/4)

21. Pre-defined searches
o - stores search criteria
* - save as pre-defined search
* - retrieve a pre-defined search and show results
"No need" (1)
"Personal" (2)

Appendix D. The list of questions about product development that were presented to product developers.

Questions for designers

Project

1. What is the name of the project that you are currently working with?
2. What is the product that is being developed in the project? What is (will be) the result of the project?
3. I the result of the project an intermediate result to some other project or is it an independent result on its own?
4. Is the product is being developed to be used by a specific customer that is currently known or will the product being sold to as is to an undefined customer base?
5. What kinds of documents belong to the final product?
6. What documents are used during the development?
7. How would you describe the project from the following statements: exploration of new ideas, application of new technology, new version that is based mainly on an existing product, product improvement, error fixing, maintenance, or something else.
8. What is the current phase of the project?
9. When will the project end?
10. When will the result of the project be published?
11. How long will you work yourself in this project?
12. How well have you been able to proceed according to the planned schedule?
13. Has approach proceeded according to the plan?
14. How does it seem to you, are you going to stay in the schedule in the future?
15. How was the schedule and contents of the project planned? Who participated in planning?
16. How has the schedule been followed?
17. What is the procedure for accepting changes in the schedule?
18. What kind of milestones (intermediate goals) do you have in the schedule?
19. What kind of problems have you encountered during the project?

Subcontracting / Subprojecting

20. What kinds of subcontracting or subprojecting do you have in this project?
21. In what kind of issues have you been using subcontracting?
22. In what kind of issues you do not use subcontracting? (Core expertise?)

Personnel and communication

23. Are there other participants in product development in addition to the core development group? => Do people have special expertise areas?
24. How has the participation off external people been organised? (Subprojecting etc.?)

With whom you communicate…

25. daily
26. weekly
27. in review meetings
28. seldom
29. otherwise?
30. How do you keep in touch with the other members of the project group?
31. Do you communicate with people outside your own organizational unit? Who are they?
32. How often and to whom you report about the progress of the project?
33. How do report of the progress?

Working conditions and tools, education and training

34. What other projects are you involved in currently?

35. How do the projects support each other that you are involved in? Or, is it a question of using your special skills in several projects?
36. Are you normally in a hurry when you work? How does hurry show in your work (documentation, reading documents, learning new skills)?
37. What tools (hardware, software) do you use to support your work?
38. Who made you familiar with this task? How was it done?
39. How do you get new information that concerns your expertise area?

Process documentation and quality system
40. What kind of written descriptions exist about the progress of the development process?
41. What kind of process documents you use and maintain?
42. How is process documentation being archived?
43. How do you use the documents from previous projects or other processes?
44. To what phases has the process been divided in the process description?
45. Do you have a quality manual?
46. What sections does it have?
47. How have you used the quality manual in this project?

Review practices
48. You have a review practice in development work?
49. What kinds of issues are being dealt with in reviews?
50. Are reviews marked in the schedule? Do the reviews have a scehdule?
51. What reviews do you participate in?

Measurement of product development
52. How do you know that you have succeeded in product development?
53. How do you measure the result of the product development? What issues are being measured?
54. How do you get information if the product has been considered as successful from the standpoint of the end-user? Miten saat tiedon kehitystyösi onnistumisesta tuotteen käyttäjien mielestä?
55. How do you measure customer satisfaction?
56. Are there awards from successful projects?

Customer and end-user participation
57. How are you in contact to the customers and end-users?
58. How you use the feedback from end-users?
59. How do customers end-users participate in product development?
60. Have you been in contact with the users of earlier products?
61. How have you received information about the use of the products that you have developed? What kind of information have you received?

Specifications
62. How were the features of the new product defined? Who participated in the work?
63. How did you participate in the creation of the specifications for the new product?
64. In what form were the specifications delivered to you as a developer?
65. What is the name of such a specification?
66. Who delivered them to you? How?
67. What kinds of sections do specifications contain?

Prototyping
68. What kind of models or prototypes do you use in product development?
69. In what stages of development do you use them?
70. How do you use them?

Usability
71. Is usability mentioned in work instructions or in quality system?
72. Is usablity mentioned in the previous documents with other concepts?