NTNU
Norwegian University of
Science and Technology

A Study of Development and Maintenance in Norway

Magne Kristoffer Davidsen

Master of Science in Computer Science
Submission date: June 2009
Supervisor: John Krogstie, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Problem Description

In 1993, 1998 and 2003 surveys were performed to investigate development and maintenance of IT-systems in Norwegian organizations. Data collection for a similar survey was carried out in the fall of 2008. The assignment will be to analyze the quantitative data from this survey, and also perform a set of case studies. The case studies will further investigate the same problems, in ways not possible in a traditional survey investigation. Together with a literature review, the survey investigation is expected to give us new knowledge about mechanisms affecting resource utilization related to information systems support in organizations.

Assignment given: 15. January 2009
Supervisor: John Krogstie, IDI

ABSTRACT

System maintenance has for a long time been reckoned as the IT departments' largest expense. This investigation is an extension to similar investigations performed earlier, and our motivation is to document how different organizations overcome challenges related to development and maintenance of IT-systems. The background for this work is the idea about success in the IT department is defined by the work it does. The purpose of the IT systems is to support business processes. A measure for the efficiency of the IT department can therefore be its ability to develop new functionality, needed by the core business of the organization.

This report is presenting the results of a survey investigation performed in 2008. Respondents from 65 Norwegian companies and organizations participated. This investigation is the fourth in a series of such investigations, with similar investigations performed in 1993, 1998 and 2003.

We hope the results can increase the knowledge within this area, and contribute to a better awareness of dependencies and different factors that affects time spent on development and maintenance. In addition to results regarding development and maintenance, we have also focused on outsourcing of different IT activities. Another area of interest is deployment of a service-oriented architecture, and how this may affect maintenance and development practices within the organizations.

The share of maintenance we have found in this investigation is 62%, when we look at maintenance and development alone. The share of application portfolio upkeep is found to be 63%. These numbers seem to stay very stable over the years. In 1993 the share of maintenance was 59%, in 1998 it was 73% and in 2003 it was 61%.

PREFACE

This report is the documentation of my Master Thesis, written in the 10th semester of the Master Programme in Computer Science at NTNU, Norwegian University of Science and Technology. The work with this report started January 2009, and continued the research done for my Depth Study in the fall of 2008.

First and foremost I would like to thank John Krogstie for being a patient and understanding supervisor, and for dedicating a lot of his time to this project. His advice and input has been crucial to the quality of this study.

I would also like to thank The Norwegian Computer Society for the co-operation, and all the respondents who took the time to complete our survey.

Finally I would like to thank the IT leaders dedicating their time to our case studies. This contribution was of the outmost importance for the study.

Thank you!

Trondheim,
June 5th 2009

_____

Magne K. Davidsen

CONTENT

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

The introduction chapter presents the research objective and motivation for this investigation. Further it describes the project context, our problem definition and finally a description of the report outline.

1.1 RESEARCH OBJECTIVE

System development is one of the major fields of Computer Science, and the use of software systems in organizations is still increasing. Development of new systems also increases, but there are many old software systems in use. Earlier investigations show that systems 20 years old are not uncommon, and they are still relied on to take care of core businesses. Many of these systems were probably not expected to last this long, and therefore maintenance was not necessarily a main concern of the developers.

This leads us to the motivation for doing investigations like this one, as it is interesting to look at time and money spent on maintenance and development. How the resources are spent, and what kind of strategies different companies choose for their software portfolio management may tell us something about state-of-practice in Norwegian IT-departments. Earlier investigations show that maintenance takes up more time and resources, than any other IT-activity. Not only so, but the amount of maintenance may also still be increasing [2].

In addition to the questions asked earlier, regarding maintenance and development seen in context of different attributes, we have added two fields of interest; Service-oriented architecture and outsourcing. Numbers on how widely outsourcing and SOA is in use is interesting alone, but especially in the light of development and maintenance. As mentioned this has not been a part of the earlier surveys, but as these subjects are becoming more and more relevant in context of development and maintenance, it felt natural to investigate this.

In earlier years software maintenance costs were underestimated, and seldom taken into account when development projects were established. Projects were unmanageable and code difficult to maintain [3]. This seems to have improved over the years, but a depth study of the area is still of value, to record “the state-of-practice”.

We also hope the case studies can give more insight on how IT leaders think about development and maintenance. We might not get any certain results from these case interviews, but this input might discover interesting information about the problem area that can be further investigated.

1.2 CONTEXT

This study is a follow up study to survey’s done by Krogstie et al. in 1993, 1998 and 2003. Given this, the study will not only be an investigation in how the situation is today, but also a study of trends over time.

There are also done many similar investigations abroad, especially in the USA, which we can compare our results to. The investigations we have found most relevant, will be presented in Chapter 2.

Even though many similar investigations have been performed, there has not been a tradition to follow up the surveys with case studies. We hope that our decision to extend the descriptive results with qualitative data from interviews will give valuable insight to the field of study.

1.3 Problem Definition

The problem definition was formulated by the supervisor, and has not been changed during the period of work. The problem definition can be found in the very beginning of this report.

1.4 Report outline

The report is organized as follows:

In Chapter 2 we will do a background study of the field. Some terms will be defined, and we will look into the "state-of-the-art", as we review some previous investigations.

In Chapter 3 we will describe the research methods we have used. This includes giving a summary of the questionnaire, go through the statistical analysis and present the interview guide used during case studies.

In Chapter 4 all the hypotheses are presented, grouped in different categories.

In Chapter 5 we will look at how this investigation can contribute to the field.

In Chapter 6 we present all the descriptive results from our investigation, and compare them in detail to the other investigations relevant to our study.

In Chapter 7 we will test our hypothesis with statistical analysis.

In Chapter 8 we will summarize the most important aspects of the interviews carried out in our case studies.

In Chapter 9 we will discuss our results from the survey, the hypothesis-testing and the case study on a higher level than before.

In Chapter 10 we will evaluate our own investigation, and point out its limitations.

In Chapter 11 we will conclude our investigation, and propose further work.

2 BACKGROUND

This chapter describes basic concepts and terminology used in this report, and presents generally accepted truths about software evolution. Furthermore it will introduce previous investigations performed on the relevant areas, and summarize the "state-of-practice" that gives grounds for our investigation.

2.1 MAINTENANCE CATEGORIES

Before we investigate maintenance and development, we need to have a clear understanding of these concepts. There are many different definitions of these terms, and different researchers have various views on what type of activity shall be included in the concepts. It is therefore important that we provide a non-ambiguous definition of what we regard as maintenance and development in this investigation.

Maintenance is the modification of a software system, after it has been deployed and delivered. Over the years maintenance has typically been divided into three categories; corrective maintenance, adaptive maintenance and perfective maintenance. These categories were first suggested by E.B. Swanson [4], and have later been updated and further specified. Also, a fourth category has been added; preventive maintenance. While some researchers also include user support tasks in their definition of maintenance [2], we do not.

The latest definitions, as provided by IEEE [5]:

1. Corrective maintenance: Reactive modification of a software product performed after delivery to correct discovered problems.
2. Adaptive maintenance: Modification of a software product performed after delivery to keep a software product usable in a changed or changing environment.
3. Perfective maintenance: Modification of a software product after delivery to improve performance or maintainability.
4. Preventive maintenance: Modification of a software product after delivery to detect and correct latent faults in the software product before they become effective faults.

Further, perfective maintenance can be divided into enhancive maintenance, and non-functional perfective maintenance [6]. In enhancive maintenance, features and functionality are added. In non functional perfective maintenance, the quality aspects of the system are improved (e.g. security, responsiveness) [1].

In addition to the traditional distinction between different kinds of maintenance, [1] introduces two new concepts:

1. *Application portfolio upkeep.* Work made to keep up the functional coverage of the information system portfolio of the organization. These categories are found to the left in Figure 2-1 and includes:

- Corrective maintenance.
- Adaptive maintenance
- Non-functional perfective maintenance.
- Development of replacement systems.

2. *Application portfolio evolution.* Development and maintenance where changes in the application increase the functional coverage of the total application systems portfolio of the organization. These categories are found to the right in Figure 2-1 and includes:

    - Development of new systems that cover areas, which are not covered earlier by other systems in the organizations.
    - Enhancive maintenance.

Application portfolio upkeep can be argued to be a better measure than traditional maintenance, when it comes to the IT department's efficiency. This because the focus is on whether or not the work done adds functionality to the systems, while maintenance and development only focuses on whether the work is done on an existing or a new system.

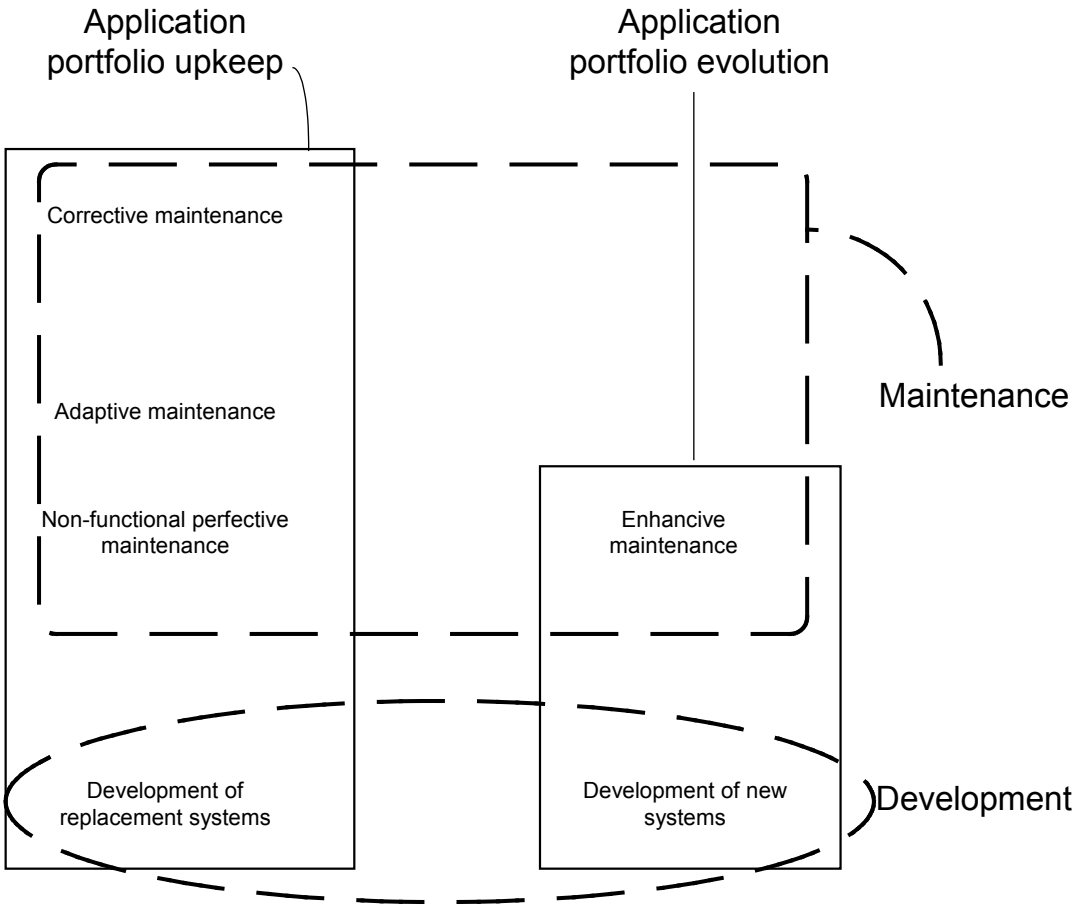

Figure 2-1: Definitions of maintenance and development [1]

2.2 SOFTWARE EVOLUTION

Software evolution is a term used to describe the continuous process of maintaining, updating and upgrading the software after it has been developed. The reasons for change may vary, and how the evolution manifests may differ. We will now look into different aspects of software evolution.

There are many different approaches to how one should develop software, and also how the software shall be monitored and maintained after the software is deployed. We will now have a look at what different life cycle models have been popular in the history of software development, as well as alternative methodologies to let information systems evolve.

2.2.1 TRADITIONAL PROCESS MODELS

This will be a short introduction to the most popular traditional process models that have been in widespread use throughout the history of software. This is only meant as background information, and we will not go into detail of pros and cons of the different models.

Code-and-Fix model
This is an ad-hoc, and not specifically well defined, two-phased method. The model simply states that you code some, and the fix it. The fixing can either be related to error correction, or further functionality implementation. There is no room for analysis, design or testing of the application as a whole. Every stage of software development has to be fit into either of the two stages; code or fix [7].

Waterfall model
This sequential model was first introduced in the 1970s, and was for a long time regarded the basis for most programming projects. The idea is to split the software lifecycle into distinctive phases, and complete each phase before you enter the next one. The seven phases of the waterfall model are [8]:

1. Requirement specification
2. Design
3. Implementation
4. Integration
5. Testing
6. Installation
7. Maintenance

Incremental model
The incremental model is a response to the linear waterfall model. Instead of finishing all the different phases for the entire system, you start with a subset of the requirements. After the whole process is completed for these requirements, you have a running system which you can put into production. Then you go back to requirements, and extend the functionality of the system. This is repeated until you have a system fulfilling all your requirements [9].

Spiral model
The idea of the spiral model is much like the incremental model. However, the phases are named a little different, emphasizing risk analysis and customer evaluation. The four phases are [10]:

1. Planning – requirements gathering
2. Risk analysis – analysis and prototyping
3. Engineering – coding and testing
4. Evaluation – customer evaluation

Also these phases are incrementally executed, until the system passes the customer evaluation without any new requests.

Agile software model
"Agile" is a concept gathering many different software development methodologies, such as Extreme Programming, Test Driven Development and Scrum. Common features of these methodologies are short iterations, and contribution from customers in the development process. The Manifesto for Agile software development states four values that summarize agile as a concept [11]:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

In the early days of software development, the customers and users typically approved the requirement specification up front, and received a system some time later, hence the waterfall model. Over the years we have moved towards a trend where users of the systems are more included in the development and maintenance of the systems. Change requests and bug reports are collected from the users, and used in further development of the software.

2.2.2 MAINTENANCE PROCESS MODELS

The need for maintenance process models in addition to the traditional process models has been acknowledged for some time [7]. It can be argued that the traditional process models do not deal with the evolutionary nature of software systems. They only deal with maintenance as the phase after deployment, without any specified approach to problems that may occur [12]. Still, maintenance process models are not as well developed, nor as well understood, as the traditional process models. Although the use of these models may not be widespread, they are relevant to our investigation as background information. We will now review some of these models briefly.

Quick fix model
This is the equivalent to the Code-and-fix model. The model has two phases; identify the problem, and fix it immediately. This method deals with the problems efficiently, but without any regards of long term consequences of the fixes [12]

Osborne's model
Osborne regards difficulties in communication and poor management the main reasons for technical problems that arise during maintenance. He therefore suggested the following strategies to address these issues [12]:

1. Maintenance requirements need to be included in the change specification.
2. A quality assurance program is required to establish quality assurance requirements.

3. A metrics needs to be developed in order to verify that the maintenance goals have been met.
4. Managers need to be provided with feedback through performance reviews

The Staged Model
This model was first introduced in 1999, and consists of five stages. The main goal of this model was to split the "maintenance stage" into three stages; evolution, servicing and phase-out. These are thought of as sequential stages in the lifecycle of an information system. The original stages are [13]:

1. Initial development
2. Evolution
3. Servicing
4. Phase-out
5. Closing down

In an enhanced version of this model, called "The Versioned Staged Model", a change is proposed. These stages are not longer sequential, but different "Evolution versions" are established along the lifetime of the system. These different version main in turn be serviced, phased out and closed down [13].

The iterative enhancement model
This model is an adaptation of the traditional incremental model. This model regard the changes made to a software system while it is in production as a iterative process. The iterations consists of three phases [12]:

1. Analyze the system
2. Classify proposed modifications
3. Implement changes

The system must be thoroughly documented for this model to be applicable, as the analysis involves reviewing of the documentation.

2.2.3 LEHMAN'S LAWS OF SOFTWARE EVOLUTION

To get at understanding of how software might evolve over time, we will now present Lehman's Laws of Software Evolution as a reference.

The first three of these laws of software evolution were first formulated in 1974 by Lehman. Over the following twenty years the laws has been revisited, and today they count eight laws, describing behavior of software evolution over time [14]. Lehman claims that all the laws relate to "E-type systems", systems that operate in or solve a problem related to the "real world". It is worth mentioning that these are not "laws" in the original sense of the word, but more like patterns that the software evolution was observed to follow.

Lehman's Laws of Software Evolution:

1. (1974) Continuing Change — E-type systems must be continually adapted else they become progressively less satisfactory
2. (1974) Increasing Complexity — As an E-type system evolves its complexity increases unless work is done to maintain or reduce it
3. (1974) Self Regulation — E-type system evolution process is self regulating with distribution of product and process measures close to normal
4. (1978) Conservation of Organizational Stability (invariant work rate) - The average effective global activity rate in an evolving E-type system is invariant over product lifetime
5. (1978) Conservation of Familiarity — As an E-type system evolves all associated with it, developers, sales personnel, users, for example, must maintain mastery of its content and behavior to achieve satisfactory evolution. Excessive growth diminishes that mastery. Hence the average incremental growth remains invariant as the system evolve
6. (1991) Continuing Growth — The functional content of E-type systems must be continually increased to maintain user satisfaction over their lifetime.
7. (1996) Declining Quality — The quality of E-type systems will appear to be declining unless they are rigorously maintained and adapted to operational environment changes.
8. (1996) Feedback System (first stated 1974, formalized as law 1996) — E-type evolution processes constitute multi-level, multi-loop, multi-agent feedback systems and must be treated as such to achieve significant improvement over any reasonable base.

These patterns can be summarized to two acknowledged truths about software evolution; software in use will undergo changes as time passes, and as the software change the complexity and entropy of the program increases.

However, not everyone agrees to the Lehman Laws. Especially open-source software has been believed not to follow all these patterns. Some of Lehman's assumptions regarding software do not automatically hold for open-source software, and this has been further investigated through case research. Both Linux, the open-source operating system, and Firefox, the open-source web browser, have been subject to such investigations [15] [16]. Both studies confirm that also open-source software endorse Lehman Laws to some extent, but agrees that the laws should be more generalized if they were to include all software. Especially law #3 and #5 does not relate to an open-source approach to evolution and further development of software.

2.3 SERVICE-ORIENTED ARCHITECTURE

Service-oriented-architecture provides a designing framework for system development and software integration, based on an organization's business processes. The motivation for such architecture is mainly reuse of legacy components and data, through standardized interfaces within the organization [17]. If this is achieved it is believed to ease maintenance and development and make the organization more agile in terms of chaning its business processes, given that the software and the business processes are well aligned. Since SOA is believed to ease and increase efficiency of

development and maintenance, this is of interest to our study. We want to investigate if these assertions hold.

The two aspects of SOA that will directly affect software evolution in the organization is maintenance of the service-oriented system itself, and the fact that legacy systems will increasingly migrate to SOA environments to make legacy functionality available [18]. SOA is therefore interesting in terms of software evolution, and we will now have a look at investigations performed to map out SOA's influence in this area. We will not go into detail on describing the SOA design patterns, or discuss whether such architecture is appropriate. The main focus will be on SOA's influence on software maintenance and evolution, but also to what extent SOA is implemented in the IT systems as of now.

A study from 2008 mentions these implications as the main challenges when a SOA approach is taken, in contrast to traditional application portfolio evolution [18]:

- The diversity of services consumers and service providers
- Shorter release cycles because of the capability of rapidly adapting to changing business needs
- The potential to leverage legacy investments with potentially minimal change to existing systems

Even though part of the motivation for deploying a service-oriented architecture is to simplify maintenance and development of components, the system itself might actually become more complex after SOA is introduced. With every component loosely coupled, new aspects of maintenance and evolution come into question. An investigation of software maintenance and complexity performed in 1998 states that one of the biggest factors effecting evolution efficiency, is the belief that new tools and techniques will "solve all our problems" [19]. This also applies to SOA. If the strategy is not implemented correctly, the organization might end up with a random selection of services that are never used [18]. This will obviously increase the application portfolio upkeep, without any advantages to the organization.

More specific challenges related to the maintenance of a service-oriented architecture might be [18]:

- The services must be designed to serve a potentially unknown set of users
- Release cycles of services, and consumer applications must be synchronized
- Need to determine who is responsible for shared services, and their maintenance
- Some services might be external, and not fully controllable

With a Gartner report done in 2008 showing that the share of organizations hoping to adapt a service-oriented architecture is falling from 53% in 2007, to 25% in 2008 [20], it is interesting to see whether or not Norwegian organizations are planning to implement SOA.

In a 2007 study only 37% of the enterprises in the investigation achieved a positive return on investment for deploying SOA [21]. Of course, this could be because of various reasons, such as short time passed since deployment or too high expectations,

but it is still interesting to look at the trends of organizations having deployed a service-oriented architecture.

Judging from the investigations we have looked at, SOA may have a major impact on software maintenance if implemented correctly. However, there is still no agreement that these theories actually apply. It is therefore natural to include questions regarding implementation of SOA in our survey, to contribute to the ongoing global investigation of SOA techniques and their effect on IT-activity.

2.4 OUTSOURCING

In general, outsourcing is the action of moving some routines, processes or functions out the organization, and let a third-party execute them instead. We are only interested in IT activity, so in our definition of outsourcing, this includes all IT-activity not done by the organizations own staff or consultants working in-house. This IT-activity could be development or maintenance of IT-systems, IT-operations or IT Management.

Outsourcing of IT-services has been increasing over the last years, and is included in what have been called "The New Wave" of outsourcing [22]. This is a term for outsourcing of "white collar" jobs, such as tasks related to legal services, economic services and technology services. The goals are the same as for any other type of outsourcing. The organizations want either to reduce cost, risk or both of activities not related to the core business. Low availability of, or a wish not to employ, staff with the required skills can be another motivation to outsource activities not related to core business. Outsourcing is of interest to our study, since we want to investigate whether or not outsourcing affects the efficiency of development and maintenance.

A study of outsourcing performed by Gartner Group in 2004 shows that 70% of the Fortune 500 companies do outsource some of their IT-activities [23]. Another investigation done by Eurostat in late 2007, reveals that as much as 72% of Norwegian companies outsource some of their IT services [24]. Maintenance is usually not regarded as core business for an organization, and is therefore a natural candidate for outsourcing. However, there are several issues that come into question when outsourcing this part of the software lifecycle [23]:

- If development of the software was outsourced, shall the same contractor perform maintenance? Advantages may be knowledge about the system, while disadvantages may be price and maintenance skills of the contractor, as there may be better offers.
- Shall one contractor be given responsibility for all the maintenance, or shall the tasks be divided? If more contractors are used, the organization can choose specialized contractors to special tasks. However, in these situations the contractors are only partially responsible for the maintenance, leaving the parent organization with the risk.
- Independent of what strategy is chosen for the outsourcing of maintenance activity, it is important to the parent organization to be well aligned with its subcontractor, or subcontractors. The outsourcing organization should be included in document reviews, and validation of changes to the software. It is

also crucial that tools for change requests and bug reporting are shared between the two parts.

When IT-services are outsourced, it be maintenance, development, operations or support, obviously the in-house IT-activity decreases. What is more interesting is whether or not the organizations are able to profit from this outsourcing in general, for example by spending less resources on maintenance and more on development. This will be further researched in this investigation.

2.5 PREVIOUS INVESTIGATIONS

In this chapter we will give a short overview of similar investigations performed earlier. Since this investigation is a follow up to surveys done in 1993, 1998 and 2003, it is natural that we will look to these investigations especially for comparison. We will also look at other similar investigations performed from 1977 and up until now. Unfortunately, we have not been able to find similar investigations performed in the latest years. Jahr's investigation from 2003 is therefore the most recent we look at.

2.5.3 LIENTZ AND SWANSON (1977)

The investigation performed by Lientz and Swanson in 1977 has become a reference point for investigations of its kind [25]. It was a very extensive investigation, with 487 participating organizations chosen randomly from the Data Processing Management Association. Also our investigation is following the Lientz and Swanson approach.

In this investigation 51% of the system-developers time was spent on traditional maintenance. Corrective maintenance summed to 21,7% of the total time used on maintenance, adaptive maintenance 23,6%, perfective maintenance 51,3% and functional perfective maintenance summed to 41,8% .

The five problem areas of most importance found in the investigation are listed below:

1. User demands for enhancements
2. Quality of system documentation
3. Competing demands for maintenance personnel
4. Quality of original system
5. Meeting scheduled commitments

2.5.4 NOSEK AND PALVIA (1990)

An investigation carried out by Nosek and Palvia in 1990 [26]. The survey contained many of the similar questions as the Lientz and Swanson investigation, and can be regarded as a follow up. 52 American organizations participated in the survey, and traditional maintenance was reported to take up 62% of the resources, when looking at development and maintenance only.

The five problem areas of most importance found in the investigation are listed below:

1. Availability of maintenance personnel
2. The programmer's efficiency
3. The technical environment/platform
4. The users knowledge
5. The quality of the product

2.5.5 *DEKLEVA (1990)*

In 1990 Dekleva performed an investigation, similar to the Lientz and Swanson investigation from 1977 [27]. 67 software maintainers that had earlier shown interest for this kind of investigations participated in the survey. The share of traditional maintenance reported in this investigation was 66%.

The five problem areas of most importance found in the investigation are listed below:

1. Changing priorities
2. Inadequate testing
3. Performance measurement difficulties
4. System documentation incomplete or nonexistent
5. Adopting to the rapidly changing business environment

2.5.6 *ARFA (1990)*

This was an investigation carried out in Tunisia in 1990 [28]. The authors claim to follow Lientz and Swanson approach carefully, and the study has been conducted as live interviews with analysts and software managers. A total of 150 completed survey forms properly distributed among various sectors of the Tunisian data processing industry were used in the analysis.

The authors conclude that even though this was an investigation conducted among Tunisian organizations, the information discovered is related to 1990 and software in general, more than to Tunisia.

The report from this investigation shows that 48,9% of the effort goes to maintenance, while 51,1% goes to development, when we exclude other activities.

The top five problem areas related to software maintenance were listed as:

1. Quality of application system documentation

2. Lack of user understanding of application system
3. Data integrity in application system
4. Number of maintenance programming personnel available
5. Quality of original programming of application system

2.5.7 KROGSTIE (1993)

In 1993 Krogstie performed a survey investigation among Norwegian organizations [29]. There were 52 participating organizations, and invitations were sent to organizations randomly chosen among The Norwegian Computer Society's members.

The investigation focused on maintenance and development, and reported that maintenance work amounted to 59% of the work done, when looking at only maintenance and development.

The top five problem areas related to software maintenance were listed as:

1. Quality of original programming of application system
2. Quality of application system documentation
3. Turnover of maintenance personnel
4. Competing demands for maintenance personnel
5. Inadequate training of users

2.5.8 HOLGEID (1998)

In 1998 Holgeid performed a follow up study to Krogstie's investigation in 1993 [30]. The results from this study are based on answers from 53 Norwegian companies, and invitations were sent randomly to organizations on The Norwegian Computer Society's member list.

The investigation reported that maintenance work amounted to 73% of the work done, when looking at maintenance and development alone. Among other relevant results, Holgeid reported that size of the organization correlated with the amount of maintenance work being done. Organizations with many employees had had less maintenance work than organizations with fewer employees.

The top five problem areas related to software maintenance were listed as:

1. Quality of application system documentation
2. Turnover of maintenance personnel

3. Availability of maintenance personnel
4. Internal competition for maintenance personnel
5. Quality of original programming of application system

2.5.9 *FITZGERALD (1999)*

Fitzgerald performed an investigation amongst UK organizations in 1999 [31]. The survey was distributed to organizations chosen from 1997 Software User Year Book, the Times Top 1000 Companies and the membership list of two separate UK organizations for system developers. The response rate was reported to be 20%, and after some of the answers were excluded for various reasons, the total amount of responses used in the analysis was 354.

The roles of the responding persons were quite diverse. 35% were IT managers, 19% were other business managers, 9% were team managers and the rest were analysts, programmers or had "other" roles.

In this investigation it was reported that only 44% of the IT department's efforts were spent on maintenance, and 56% were spent on development. These numbers stand out as very different from any other investigation we have looked at. The report indicates that this improvement may be real, and that organizations may have improved their practices through use of better methods and techniques.

2.5.10 *JAHR (2003)*

The investigation performed by Jahr in 2003, was another follow up study to the Krogstie investigation in 1993, and the Holgeid investigation in 1998 [32]. The survey had 54 participating organizations, and invitations were distributed through The Norwegian Computer Society.

In this investigation, the share of maintenance work is 66%, when only time spent on maintenance and development is compared. It is also reported that organizations that use pre-defined methods and tools when dealing with maintenance, spend less time on maintenance work than organizations that use no such tools or methods.

The five problem areas of most importance found in the investigation are listed below:

1. Quality of the product
2. Tight budgets
3. Quality of documentation
4. Availability of maintenance personnel

5. Turnover of maintenance personnel

2.5.11 SUMMARY

We will now summarize the investigations we have been looking into. First we will compare the share of maintenance in the different investigations, when we only look at development and maintenance.

Year	Investigation	Maintenance
1977	Lientz and Swanson [25]	51%
1990	Nosek and Palvia [26]	58%
1990	Dekleva [27]	66%
1990	Arfa [28]	49%
1993	Krogstie [29]	59%
1998	Holgeid [30]	73%
1999	Fitzgerald [31]	44%
2000	Capers Jones [2]	73%
2003	Jahr [32]	66%
2010	Capers Jones [2]	79% (estimated)

Table 2-1: Share of maintenance (isolated) from earlier investigations

We see from Table 2-1 that a majority of the investigations performed since 1990, conclude with a isolated maintenance share of approximately 60%. Only Arfa's investigation from 1990 and Fitzgerald's investigation from 1999 report of values significantly lower than 60%, while Holgeid's investigation from 1998 and Capers Jones' investigation from 2000 reports values that are clearly higher. These variations will be discussed in more detail later, but it is important to point out already to now that Capers Jones include user support in his definition of software maintenance [2]. This comparison is only done to get an overview of what earlier investigations have found, and we do not go into detail on which of these discoveries are more reliable.

When it comes to problem areas, there are some that is mentioned more often than others. "Quality of the product" is listed as one of the top five problem areas six out of seven times, in the investigations we have been looking at. Problems regarding either incomplete documentation or poor quality of documentation are also listed six times. Different problems regarding maintenance personnel are also listed in six of the top five lists. This can be problems with availability of maintenance staff, internal competition for maintenance staff or high turnover of maintenance staff.

3 RESEARCH METHOD

The research method used to collect and analyze the data will be presented in this chapter. Focus will be on describing the approach to the investigation, but also why the different approaches were chosen. We will not go into detail on different methodologies, but discuss them briefly as they are presented.

3.1 CONDITIONS OF METHODS

The first aspect of method design is usually whether the investigation shall be a quantitative one, or a qualitative one. A quantitative design is often preferred if your main objective is to get objective and descriptive data, that can be easily analyzed using statistics [33]. Qualitative methods are usually preferred if you want to do explorative investigations, and are not too concerned about generalizing your results [34].

When the investigation was designed, an important aspect was to follow the tradition of similar investigations performed earlier. Some decisions where consequently already made. A survey had to be conducted, and the form of the survey should be as similar to the earlier ones as possible.

However, we decided to extend the design used before, with case studies. This form of multimethodology, combining quantitative and qualitative research methods, helps us to neutralize some of the shortcomings of both methods. First we did a traditional survey investigation, and then we did a statistical analysis of the results. Later we carried out case studies of a few organizations, to get their thoughts on maintenance and development.

3.2 OVERVIEW OF THE RESEARCH METHOD

We started the whole process with a literature review, to get a good understanding of the "state of the art" in this field of study. Both papers describing development and maintenance frameworks were reviewed, as well as previous investigations in this problem area. The literature preview has also been performed parallel with the rest of the investigation, as more information and knowledge has eventually been necessary to get a good understanding of different topics.

Since our investigation was performed as a follow-up study to earlier works it was not only natural, but also necessary, to use the same methods as earlier. However, this time we decided to expand the investigation by adding case-studies, in hope of revealing new information.

The data was collected through a questionnaire, distributed to Norwegian organizations by e-mail, chosen randomly from The Norwegian Computer Society's members. Although Haug's investigation from 2008 shows that only organizations of a certain size have their own IT-department [35], we still find answers from smaller organizations relevant for some aspects of the investigation (e.g. questions related to outsourcing, use of consultants and IT-budget)

From previous surveys of this kind, the response rate has been around 20%. As we were hoping to get a minimum of 50 responses, we sent the form to 300 different organizations. With the relatively low response rate we were expecting, it is important that there is no common reason that some companies are responding, and some are not. If there is a system to it, it will hurt the generalization of the results [34]. There is no reason to believe this was the case, but since we do not have much information about the organizations that did not respond we cannot be certain. We need to take this into consideration when we analyze our results.

Of the 300 e-mail invitations sent out, 22 could not be delivered. Either because of invalid e-mail addresses or reservation against SurveyMonkey [36] surveys. After we sent the invitation to the survey, we sent out three more reminders by e-mail. This finally got us 53 complete and 22 partial responses. A total of 75 responses gave us a response rate of 25%. However, 10 of these responses were considered too incomplete. This left us with 65 answers usable for analysis.

After the results from the survey had been analyzed, we approached seven organizations with the request to do a short interview regarding their IT activity. The aim was to do a study, following a multiple-case design, which were meant to give us further details and descriptive results [37].Three of these organizations responded positively, and the interviews were arranged and completed. The analysis of the results from the interviews was purely qualitative. The main focus was to find similarities and differences in how these organizations handled maintenance and development issues, and what they found challenging in this matter. No statistical measures were used to find relations between the answers.

3.3 PRESENTATION OF THE SURVEY

Since this is a follow up study to similar investigations carried out before, it is in our interest to look at trends. For this reason, we have kept the questions we feel are still relevant. Some new questions have also been added, to explore new areas. Especially service-oriented-architecture and outsourcing have been chosen as fields of interest. Some questions were also removed, as they dealt with topics we did no longer find relevant.

The survey consists of mostly closed questions, with possibilities to comment on the answer given. The only open questions are the ones requiring the respondent to fill out numbers. After we decided on all the questions, they were entered into an on-demand internet service called SurveyMonkey [36].

Before the questionnaire was distributed, a pilot survey was done. We sent the pilot both to survey experts, who could support us on general quality aspects of the questionnaire, and to domain experts. Our main motivation was to be sure the questions were understandable, and not ambiguous.

3.3.1 THE QUESTIONNAIRE

This is just a summary of the questions in our survey. For the full questionnaire, see Appendix I

#1 - #3	are contact info on the participant.
#4 - #8	are facts about the participant, such as position, education and years of IT experience.
#9 - #13	are related to quick facts about the company. E.g. what business is the company in, how many employees does it have and how big is the IT budget.
#13 - #17	are about how IT-tasks are done in the organization, and how much resources are spent on each task.
#18- #22	are facts about the organizations IT-department. Such as how many employees the IT-department has, and what their background are.
#23 - #31	are questions related to application portfolio upkeep, evolution and usage of the IT-systems.
#32 - #36	are about use of technology. This contains what programming languages are in use, and also if the organization has deployed a service-oriented architecture.
#37 - #42	are questions about development of new systems. This contains questions about why the new systems are developed, and further questions on why systems are replaced.
#43 - #47	are related to usage of tools and methods when developing and maintaining IT-systems.
#48 - #49	are questions about what kind of problems the organization has encountered, in regards of system maintenance.

3.4 STATISTICAL ANALYSIS

For the statistical analysis we have used SPSS. Kolmogorov-Smirnov and Shapiro-Wilk tests were used to determine whether or not the maintenance variables were normally distributed. As most of them showed to not be normally distributed, the non-parametric Wilcoxon Signed Ranks tests were used to test the hypotheses where two different variables were compared.

For hypotheses where the correlation between organizational variables and the maintenance variables were tested, we used Spearman's rank correlation coefficient. This is also a non-parametric test.

In hypotheses where we tested the same variable in two different groupings against each other we used the T-test for independent samples. This is a statistical hypothesis test which is best used with normally distributed numbers. However, it can also be used when the numbers are not normally distributed, but it is then less powerful.

3.5 INTERVIEW GUIDE

Before the interviews were conducted, a interview guide were composed. This was to ensure the different answers would be somewhat comparable. We will now go through the 10 questions in this interview guide.

#1 **"How is the IT activity organized?"** – This very open question was used to get the interview started, and to give the interviewer a general perception of the IT organization. We also wanted to reveal what type of activity the organization did in-house, and what they outsourced.

#2 **"How are IT projects proposed, prioritized and conducted**?" – Here we want to find out whether or not the organization has any organization controls enabled for conduction of IT projects.

#3 **"What effect does the organization want from its IT investments? What is regarded a successful investment, and how is this measured?"** – We now want to investigate the organizations success criteria, and how well they measure them.

#4 **"Does the organization distinguish between different types of maintenance when planning?"** – Here we want to investigate the awareness of maintenance terms, and how the organization handles this when it comes to planning and budgeting.

#5 **"What factors are important to ensure that an IT project is successful?"** – Here we ask about what the IT leader considers important when carrying out projects.

#6 **"Does the organization have a plan to minimize application portfolio upkeep?"** – We now want to know if the organization regards it as a problem, that upkeep takes up the majority of their IT resources (if this is true for the organization). We also want to know if they actively seek to reduce the expenses related to upkeep.

#7 **"What is the general motivation for developing replacement systems?"** – Here want to know more about why replacement systems are made. What are the motivation, and does the IT leader feel that these goals are reached. We also wanted to know whether or not new functionality was implemented together with the replacement system, or if this was done later.

#8 **"Is re-use of requirements, specification and code important to the organization? Are there any guidelines for re-use? Do you achieve the amount of re-use you want?"** – We now want to know if re-use is prioritized by the organization, and if they achieve their goals in this area.

#9 **"Do you use, or plan to implement, a service oriented architecture? If so, how is it implemented?"** – Here we want to know the organizations plan for SOA, and how they have chosen to implement it.

#10 **"What is the organizations biggest challenge, when it comes to maintenance?"** – We here want to know what the IT leader considers the biggest problem related to software maintenance. We also want to know why this is challenging, and what the IT leader intends to do with it.

4 HYPOTHESES

In this chapter we will present the hypotheses. They are divided into categories only for convenience sake. The hypotheses H1 to H20 are regarding factors that have proved to be significantly connected with maintenance, in one or many earlier investigations. H20 to H24 are new hypotheses, aiming to investigate whether or not outsourcing and SOA has any impact on maintenance. H25 to H27 have also been part of earlier investigations. They consider how maintenance variables changes over time.

4.1 MAINTENANCE AND DEVELOPMENT

H1: There is no difference in the amount of time used on maintenance and development, when we only look at maintenance and development.

H2: There is no difference in the amount of time used on maintenance and development.

H3: There is no difference between the time used on application portfolio upkeep and traditional maintenance, when we look at development and maintenance only.

H4: There is no difference between the time used on application portfolio evolution and traditional development, when we look at development and maintenance only.

H5: There is no difference between the time used on application portfolio evolution and application portfolio upkeep.

4.2 TYPE OF ORGANIZATION

H6: There is no difference in the amount of maintenance-work between organizations with many employees and organizations with fewer employees.

H7: There is no difference in the amount of maintenance-work between organizations where maintenance is often performed by the people who developed the system, and organizations where maintenance is rarely performed by the people who developed the system.

4.3 IMPORTANCE OF IT

H8: There is no difference in the amount of maintenance-work between organizations in which the size of the IT-department compared to the total number of employees is large and the organizations where the size of the IT-department compared to the total number of employees is small.

H9: There is no difference in the amount of maintenance-work between organizations in which there are many system-developers in proportion to total number of internal users, and organizations with few system-developers in proportion to total number of internal users.

H10: There is no difference in the amount of maintenance-work between organizations in which there are many system-developers in proportion to total number of employees in the IT department, and organizations with few system-developers in proportion to total number of employees in the IT department.

4.4 Consultants and Employees

H11: There is no difference in the amount of maintenance-work between organizations with hired IT-consultants and organizations without any hired IT-consultants.

H12: There is no difference in the amount of maintenance work between organizations with higher turnover amongst developers, and organizations with lower turnover amongst developers.

4.5 Complexity of the Portfolio

H13: There is no difference in the amount of maintenance-work between organizations with many main systems and organizations with fewer main systems.

H14: There is no difference in the amount of maintenance-work between organizations with many internal users and organizations with fewer users.

H15: There is no difference in the amount of maintenance-work between organizations with main-systems with high age average, and organizations with main-systems with low age average.

H16: There is no difference in the amount of maintenance-work between organizations with many main-systems which are dependent on data from other systems, and organizations with few main-systems which are dependent on data from other systems.

H17: There is no difference in the amount of maintenance-work between organizations that use many different programming-languages, and organizations that use fewer different programming-languages.

4.6 USE OF METHODS AND TOOLS

H18: There is no difference in the amount of maintenance-work between organizations that use pre defined methods throughout the systems lifecycle, and the organizations that do not use this.

H19: There is no difference in the amount of maintenance-work between organizations that use system development tools, and the organizations that do not use this.

H20: There is no difference in the amount of maintenance-work between organizations that use defined organizational controls, and organizations that do not use this.

4.7 OUTSOURCING

H21: There is no difference in the amount of maintenance-work between organizations that outsources much of the total IT activity, and organizations that outsource less of the total IT activity.

H22: The use of outsourcing is not dependent on the size of the company.

4.8 SERVICE-ORIENTED ARCHITECTURE

H23: There is no difference in the amount of maintenance-work between organizations that has deployed service oriented architecture and organizations that has not deployed service oriented architecture.

H24: The use of service oriented architecture is not dependent on the size of the company.

4.9 TIME PERSPECTIVE

H25: There is no difference in the amount of maintenance-work between organizations that participated in this investigation, and the investigation performed in 2003.

H26: There is no difference in the share of total new systems that is classified as replacement systems in our survey and what were reported in 2003.

H27: The average age of a system that is being replaced, is the same in our survey and what were reported in 2003.

5 CONTRIBUTION

We will now outline generally what contributions an investigation like this can offer, but also what can be taken from this investigation in particular.

5.2 INVESTIGATIONS IN GENERAL

Today, it can be regarded as common knowledge among IT-managers, and others interested in IT operations, that maintenance is the most extensive of all the IT-activities. Over the years a number of investigations, independent of each other, have reported the same findings. This forms our perception of "state-of-practice", and proves the importance of doing such investigations in the first place.

Previous investigations summarized in Table 2-1 shows that 50%-80% of the IT budgets for maintenance and development are usually spent on maintenance. Often as much as ¾ of an investment in software is spent after the system has been deployed [38]. With this in mind, it seems obvious that there is a motivation to improve software maintenance performance. Discovering what affects the organizations efficiency is the first step towards proposing new routines.

In software development, changes are happening every year. New programming languages gain ground, new frameworks are developed, new principles are proposed and new methodologies implemented. It is of great interest to see whether or not these changes are able to affect the efficiency of development and maintenance, and the quality of the systems, the way they are meant to.

5.3 THIS INVESTIGATION

As number four in the series of an ongoing investigation series of development and maintenance in Norway, this study is absolutely of value. Even though the investigation is relatively small, and it therefore might be difficult to generalize the results, the results will still be a statement of what is going on in the Norwegian organizations. The fact that this is a series of investigations will also strengthened any conclusions drawn from these investigations, if the same results are reported time after time.

There are not too many of these investigations. In our background study, we could not find any relevant investigation carried out after 2003, when number three in this series took place. True enough there are investigations of software maintenance, in particular investigations dealing with different type of maintenance types. However, these investigations usually consider only one system, or one organization. The focus seems to be strictly on maintenance, and not how maintenance is a part of the application portfolio management.

This investigation is quite different, as it looks at many organizations, and examines how software maintenance is practiced by the IT departments. In other words, we are not so concerned about details of software maintenance, but rather the impact the organizations' IT strategy has on their own maintenance and development.

The job of mapping the development in this field does not seem to be done particularly well neither in Europe nor in the USA. A continuous documentation of how this field evolves in Norway is therefore important.

Stakeholders can be those with commercial interests in the IT-industry, looking for best practices. An investigation like this might propose what works, and what does not. Organizations may also be able to compare themselves to our results, and see whether they are better or worse than the average.

Finally the results are also of academic interest. New hypotheses can be formed from the results, and findings can be the basis of more investigations. Also, when developing and researching new methodologies, tools or frameworks, it is important to know what is status quo in the field of study.

6 DESCRIPTIVE RESULTS

In this chapter we will present descriptive results from the survey. We will not go into discussions or relate the results to our hypotheses, as this will come later. In this chapter there will be several references to the investigations this study is meant to follow up. Numbers from [29], [30] and [32] will be mentioned as the results from 1993, 1998 and 2003.

6.1 RESPONDENTS

The survey was addressed to the organization's contact person towards The Norwegian Computer Society or the IT-leader of the organization. It has been our intent that they complete the survey themselves, as this is how earlier investigations has been carried out.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Manager	56	86,2	86,2	86,2
	Project manager	7	10,8	10,8	96,9
	System developer	2	3,1	3,1	100,0
	Total	65	100,0	100,0	

Table 6-1: Respondents title

As we see from Table 6-1, as much as 96.9% of the respondents were either managers (86.2%), or project managers (10,8%). Compared to the investigations from earlier years, we see that the results are about the same. In 1993 the respondents were 94% managers, in 1998 there were 90.6% managers, and in 2003 there were 81.5% managers. As managers and system developers may have different opinions about questions asked in this investigation [39], it is an advantage to have the group homogenous in this matter. Also, since most investigations in this field is carried out amongst leaders, this is good for comparison.

We also asked about the respondent's employment status. Employers that are working for the organization temporarily or are hired as consultants are likely to know less about the organization than those who are permanently employed. This time all of the participants were permanently employed. In the earlier investigations from 1993, 1998 and 2003, the percentage of permanently employed respondents was 98.1%.

	N	Minimum	Maximum	Mean	Std. Deviation
Years	65	0	34	17,35	7,402
Valid N (listwise)	65				

Table 6-2: Years of IT-experience

In Table 6-2 we can see that the average respondent has 17,35 years of IT-experience. This is approximately the same as earlier (16,7 in 1993, 14,2 in 1998 and 14,5 in 2003). The standard deviation in 2003 was 7,5, and in 1998 7,9. We see from Table 6-2 that the standard deviation is still in the same range (7,40).

The experience of the respondents is relevant because it might affect the answers. IT-leaders with long experience will probably base their answers on problems they have experienced in their career, while newly educated leaders with less experience might base their answers on theory, common perception and what they have learned.

6.2 ORGANIZATIONS

We will now have a look at the different aspects of the organizations the respondents represent.

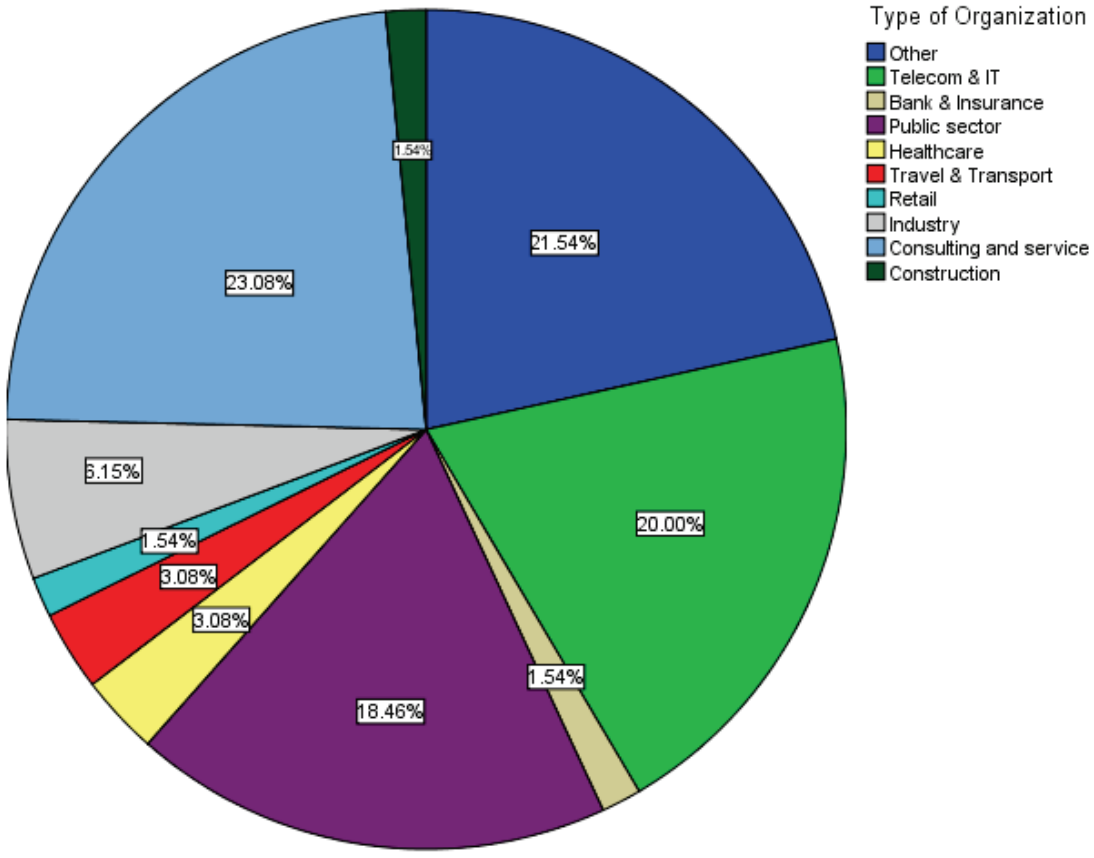

Figure 6-1: Type of Organization

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Other	14	21,5	21,5	21,5
	Telecom & IT	13	20,0	20,0	41,5
	Bank and insurance	1	1,5	1,5	43,1
	Public sector	12	18,5	18,5	61,5
	Healthcare	2	3,1	3,1	64,6
	Travel and transport	2	3,1	3,1	67,7
	Retail	1	1,5	1,5	69,2
	Industry	4	6,2	6,2	75,4
	Consulting and services	15	23,1	23,1	98,5
	Construction	1	1,5	1,5	100,0
	Total	65	100,0	100,0	

Table 6-3: Type of Organization

Compared to the investigation completed 5 years ago, the participating organizations are more equally divided amongst the different types of organizations this time. In 2003 the participants were 46,3% Telecom & IT companies, and 40,7% Consulting and other service companies. From Figure 6-1 and Table 6-3 we see that these numbers are reduced to 20,0% and 23,1%. An even distribution is good for generalization of the results among different lines of business. However, it may affect the basis for our comparison to the earlier investigations, where the participating organizations were not as equally divided among the categories.

Amongst the other investigations we will compare our results to, it is worth mentioning that Fitzgerald's investigation from 1999 had as much as 60% consultancy businesses and software houses participating [31].

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	No importance	1	1,5	1,5	1,5
	Little importance	1	1,5	1,5	3,1
	Some importance	8	12,3	12,3	15,4
	Severe importance	19	29,2	29,2	44,6
	Absolute importance	36	55,4	55,4	100,0
	Total	65	100,0	100,0	

Table 6-4: Strategic importance of IT

Since mostly IT-managers are responding to the survey, it is not surprising to find that the respondents consider IT to be of strategic importance to their organization. The majority of the respondents (84,6%) has answered that IT is of "Absolute" or "Severe" importance. However, this is lower than in the investigation done 5 years ago, where the number was 92,6%.

	N	Minimum	Maximum	Mean	Median	Std. Deviation
Number of employees	65	1	35000	1083,14	48	4520,841
Valid N (listwise)	65					

Table 6-5: Number of employees

From Table 6-5 we see that the mean number of employees in the participating organizations is 1083,14, and the median 48. The corresponding numbers in 2003 was a mean of 181 and median of 27, in 1998 the mean was 656 and median was 160 and 1993 had a mean of 2347 and a median of 555. Judging from these numbers, the participating organizations are generally bigger this time than in 2003. However, they are not as big as in 1998 and 1993.

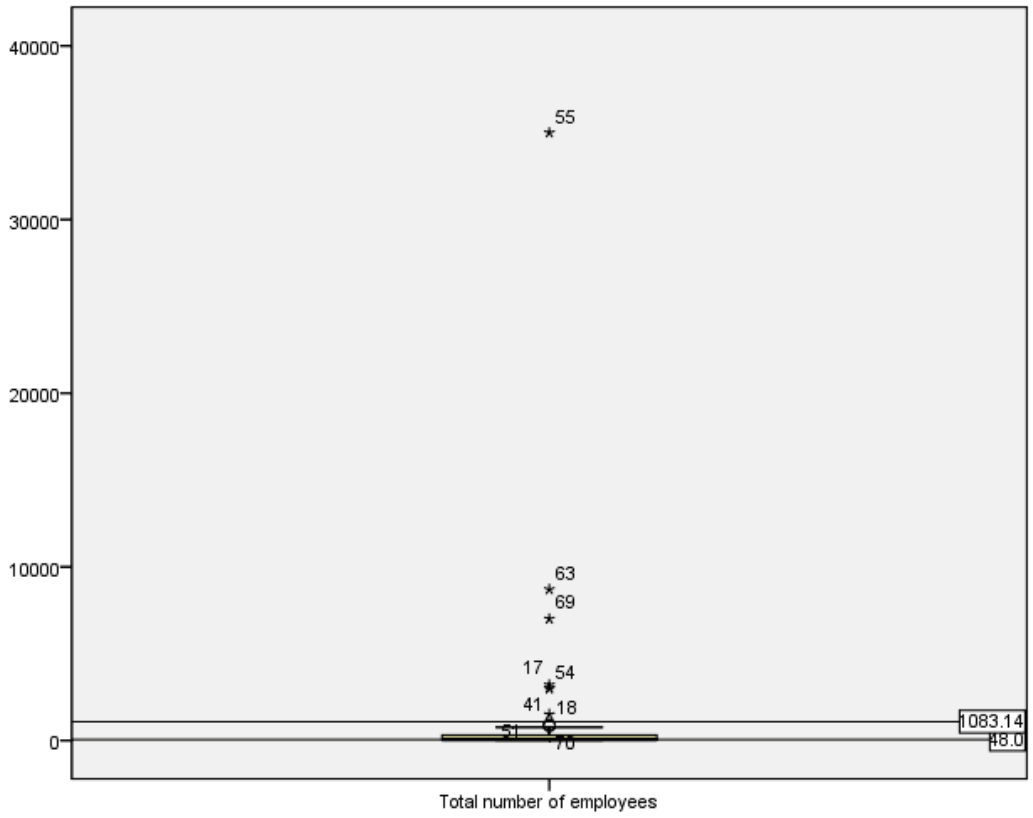

Figure 6-2: Total number of employees, boxplot

We have included a boxplot of the total number of employees, just in order to visualize how a few outliers affect the mean value drastically.

	IT-Budget 2008		IT-Budget 2003		IT-Budget 1998		IT-Budget 1993	
Million NOK	**N**	**Percent**	**N**	**Percent**	**N**	**Percent**	**N**	**Percent**
More than 50M	10	15,4 %	4	7,41 %	9	16,98 %	4	9,30 %
40M – 50M	3	4,6 %	2	3,70 %	2	3,77 %	1	2,33 %
30M – 40M	1	1,5 %	0	0,00 %	2	3,77 %	1	2,33 %
20M – 30M	3	4,6 %	1	1,85 %	2	3,77 %	2	4,65 %
10M – 20M	9	13,8 %	4	7,41 %	8	15,09 %	5	11,63 %
1 – 10M	18	27,7 %	28	51,85 %	18	33,96 %	13	30,23 %
Less than 1M	21	32,3 %	15	27,78 %	12	22,64 %	17	39,53 %
Total:	65	100,00 %	54	100,00 %	53	100,00 %	43	100,00 %

Table 6-6: IT-budget

The size of an organization's IT-budget is usually decided by two factors; how big is the organization, and how important is IT to the organization. When we compare budgets to the ones from earlier years in Table 6-6, it seems like we have more organizations with big budgets this year than before. This might seem unlikely, as the organizations typically have fewer employees, than the participants of the investigations from 1993 and 1998. This can imply that organizations generally have bigger IT budgets today, than 15 years ago. However, we need to take inflation into account before we make such assumptions.

We will not do any thorough analysis of this, but only do a couple of quick calculations. The inflation in Norway from 1993 to 2008 was 36%[40]. This changes our 1993 categories from 1, 10, 20, 30, 40 and 50 million to 1.36, 13.6, 27.2, 40.8, 54.4 and 68 million in 2008 currency. With this in mind, the difference is not necessarily that big. Still, it seems like the IT budgets increase, and it does not seem unlikely that IT has become more and more important to the organizations.

6.3 DISTRIBUTION OF LABOR

Now we will look at what kind of work is done by the organization itself, and what is outsourced. We will also look at mean values for what kind of work is taking up most of the organization's resources.

	N	Minimum	Maximum	Mean	Std. Deviation
Total IT-activity	65		100	29,72	32,029
Development	65		100	31,62	39,991
Maintenance	65		100	30,68	37,695
Operation	65		100	37,74	39,791
Support	65		100	26,18	36,301
IT-management	65		100	6,85	20,834
Valid N (listwise)	65				

Table 6-7: Outsourcing

Since this is the first time we have questions regarding outsourcing as a part of the questionnaire, we don't have any results to compare our numbers to. From the standard deviation numbers in Table 6-7, we see that there is not really any strong trend in how much of the activity is outsourced. However, if we look at whether the organizations use outsourcing or not, only 13,8% have answered that they don't

outsource anything at all. In other words, 86,2% of the participating organizations are outsourcing IT-activities to some extent.

This is number is even higher than the results from Eurostat 2007 survey, mentioned in Chapter 2.3, indicating that 72% of Norwegian companies outsource some of their IT-activities. When it comes to what activities are outsourced the most and the least, only 16,9% outsource IT-management, while 72,3% outsource IT-operation to some extent.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	High accuracy, based on data	20	30,8	30,8	30,8
	Some accuracy, estimated	27	41,5	41,5	72,3
	Low accuracy, "good guess"	18	27,7	27,7	100,0
	Total	65	100,0	100,0	

Table 6-8: Quality of numbers, Outsourcing

Table 6-8 shows the quality of the data. We can see that the accuracy of the data is varying from good accuracy to little accuracy.

	N	Minimum	Maximum	Mean	Std. Deviation
Correct errors	61		50%	7,41%	7,78
Adapt	61		20%	5,70%	5,08
Add new functionality	61		45%	9,87%	8,98
Enhance non-func. Properties	61		30%	8,15%	6,77
Develop replacement systems	61		35%	8,54%	8,35
Develop new systems	61		70%	10,39%	13,20
Operation	61		59%	21,18%	14,55
Management	61		60%	11,75%	11,14
Support	61		50%	17,00%	15,02
Other	61			,00%	,000
Total share maintenance	**61**		**60,00%**	**31,13%**	**15,93**
Total share development	**61**		**70,00%**	**18,93%**	**15,40**
Valid N (listwise)	61				

Table 6-9: Distribution of labor in IT-department

In Table 6-9, the first four categories sum to "Total share maintenance", and the next two categories sum to "Total share development". The mean value of maintenance is 31,13% and the mean value of development is 18,93%. Earlier maintenance has been reported to be 40,0% in 1993, 41,4% in 1998 and 35,9% in 2003. Development has been reported to be 29,6% in 1993, 17,1% in 1998 and 21,9% in 2003.

Even though the mean value of maintenance is considerably lower than before, we should be careful drawing any conclusions. This time we have included "Management" as one of the categories in distribution of labor, and therefore this might explain the big differences. Earlier, some respondents may have included management in the "Other" category, while others may have excluded it totally. If we remove the "Management" category, and normalize our result, we get 35,3% maintenance, and

21,5% development. This is still slightly lower than earlier years, but closer to the results from 2003.

When we look at development and maintenance isolated from the other activities, development comes to 37,8% while maintenance comes to 62,2%. In 2003 the share of maintenance was 65,9% and in 1998 the number was 72,9%. In 1993 the number was as low as 58,6%. The high share of maintenance in 1998 can be explained by issues regarding Y2K, and were believed to drop after year 2000[30]. It seems like this was correct, and both numbers from 2003 and 2008 are lower than the result from 1998. This will be discussed in further detail in Chapter 9.

Also the ratio of application portfolio evolution ("Add new functionality" and "Develop new systems") and application portfolio upkeep ("Correct errors", "Adapt", "Enhance non-func. Properties" and "Develop replacement systems") is very close to the ratio in 1998 and 2003. This time application portfolio evolution comes to 37,1%, while application portfolio upkeep comes to 62,9%. The numbers from the previous investigations are 44% upkeep in 1993, 62,3% upkeep in 1998 and 61,1% upkeep in 2003. We do not have information about application portfolio upkeep from the other investigations, as they did not operate with these categories.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	High accuracy, based on data	15	23,1	24,6	24,6
	Some accuracy, estimated	23	35,4	37,7	62,3
	Low accuracy, "good guess"	23	35,4	37,7	100,0
	Total	61	93,8	100,0	
Missing	System	4	6,2		
Total		65	100,0		

Table 6-10: Quality of Numbers, Distribution of Labor

Also here we asked the respondent for what the foundation for the answers were. In Table 6-10 we see that the participants found it even harder to give accurate answers to how the IT department distributed their work, than what parts of IT-activity they had outsourced.

6.4 IT-DEPARTMENT

In this chapter we will present the results regarding the organizations' IT-department.

	N	Minimum	Maximum	Mean	Std. Deviation
Employees in IT-dept.	62		300	14,11	40,297
System Developers	62		30	2,71	4,950
IT-Consultants	62		20	2,82	4,741
Valid N (listwise)	62				

Table 6-11: Employees in the IT-department

A measure for the size of an IT-department can be the number of people employed by the department. In Table 6-11 the mean number of employees, system developers and IT consultants is displayed. All calculated to the amount of full-time employees. Our

investigation shows that the average number of IT employees is 14,11. In 1993 this number was 24,3, in 1998 it was 10,9 and in 2003 it was 9,8.

For comparison we will have a look at other previous investigations as well. Lientz and Swanson reported an average of 45,4 IT employees in 1977[25] and Nosek and Palvia reported 178 in 1990 [26]. This is significantly bigger numbers than ours, and confirms that these investigations were carried out among big companies.

In our investigation the IT department amounts to 1,3% of the total number of employees. This is about the same as in 1993 when the number was 1,0%, and in 1998 when the number was 1,7%. In 2003 this number was much higher, with the IT department amounting to 5,4% of the total number of employees. This might not necessarily mean anything. Different type of organizations can have an impact on this, and similar organizations should be compared to see if the IT departments actually gets smaller.

With the numbers from Table 6-11 , we get that 19,2% of the employees in the IT-department are system developers. This is considerably lower than in the previous investigations, were the numbers were 39% in 1993, 42% in 1998 and 42% in 2003. Also other investigations have reported this number to be around 40%. Lientz and Swanson reported that developers took up 38% of the IT departments in 1977 [25], and Nosek and Palvia reported the share of developers to be 43% in 1990 [26].This may be linked to a higher share of outsourcing, but as outsourcing was not included in the earlier investigations, we have no basis for claiming this. Also, this might be related to bigger needs for user support. As organizations now have many external users, the need for user support increases.

When it comes to consultants, we see from Table 6-11 that the average number of consultants in the IT-department is 2,82 (recalculated to full-time employees). This is higher than in 2003 (0,7), but close to the average found in 1998 (2,7). When it comes to the low numbers from 2003, this can be related to the “dot-com bubble”. The market for consultants were extremely low in the years after 2001.

45,2% of the respondents answered that their organization does not use consultants at all. This is lower than in 2003, where 56% did not use consultants, but higher than in 1998 where 30,2% did not use consultants.

The results regarding consultants may have been affected by the fact that we now ask about outsourcing. It is not always trivial to distinguish outsourcing from hiring consultants, and some may have included outsourced work as work done by consultants in earlier investigations. As we in this survey ask about both consultants and outsourcing, it is reason to believe that this is no longer a problem.

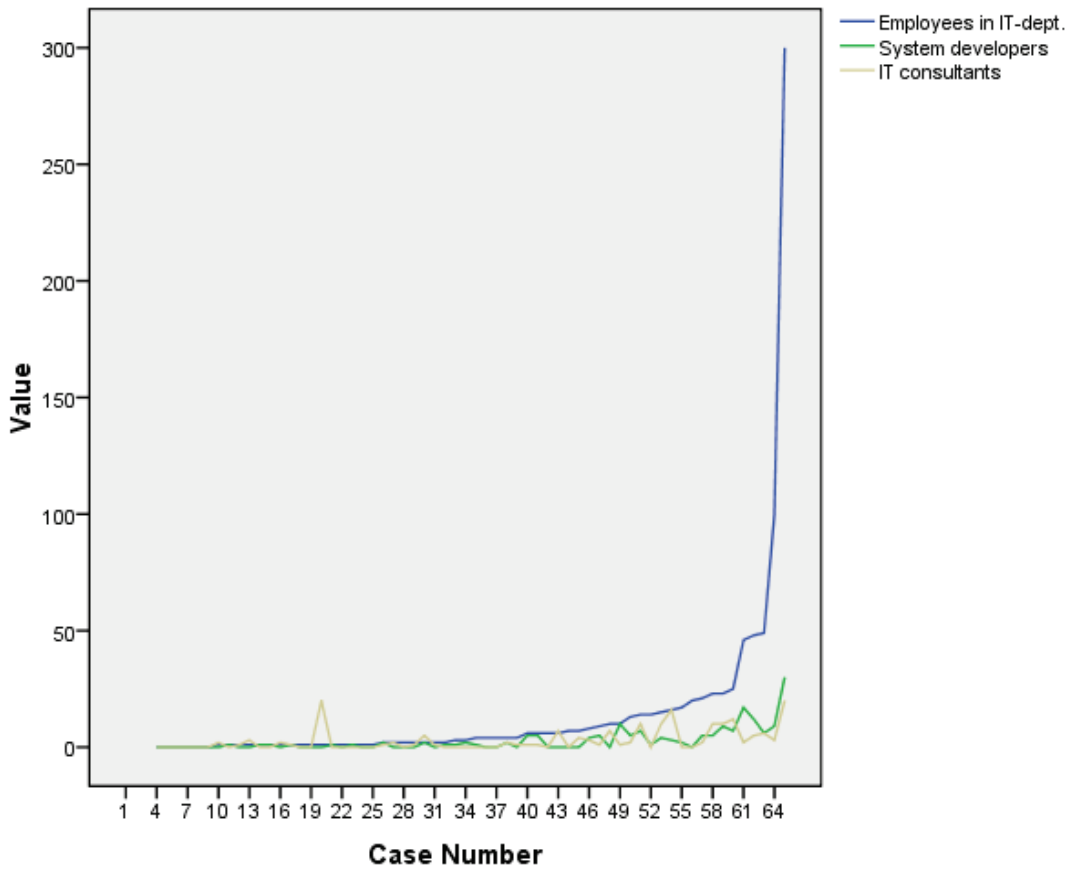

Figure 6-3: Employees in the IT-department

From Figure 6-3 we can see how the number of employees, system developers and consultants does not co-relate in any obvious way. The figure also shows how diverse the numbers are for the different cases, and reminds us that average numbers does not necessarily reveal the whole truth. Also, out of the 62 respondents that answered the question regarding their IT-department, as many as 29 of them had no system developers employed. This means 46,8% had no system developers in their IT-department at all. This is a high number, compared to the result from 2003. Of all the participating organizations in that investigation, only 25% had no system-developers on staff.

	N	Minimum	Maximum	Sum	Percent	Mean	Std. Deviation
0-1 year	33		5	22	13,1%	,67	1,137
1-3 years	33		10	41	24,4%	1,24	1,821
3-6 years	33		10	41	24,4%	1,24	2,463
6-10 years	33		7	38	22,6%	1,15	1,679
More than 10 years	33		9	26	15,5%	,81	1,768
Valid N (listwise)	33						

Table 6-12: Years of experience from the IT-department

The system-developers experience from the organizations IT-department is presented in Table 6-12. Only the 33 organizations that actually had system-developers are

counted. When we compare to the results from 2003, the most noticeable changes are a distinct decrease in the "3-6 years" category, from 45,7% to 24,4%. This seems to lead to an increase in the "6-10 years" category, and the "More than 10 years" category. When we calculate the mean value[1], the average employee has 5,8 years of experience from the IT-department. The previous investigations reported an average of 7 years in 1993, 8,8 years in 1998 and 5,4 years in 2003. It seems like IT personnel switch jobs more often after 2000, than before. This may affect both the quality and the efficiency of development and maintenance, and in hence make an impact on the time spent on these activities.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Don't know	4	6,2	6,3	6,3
	Never	2	3,1	3,2	9,5
	Seldom	7	10,8	11,1	20,6
	Sometimes	13	20,0	20,6	41,3
	Often	28	43,1	44,4	85,7
	Always	9	13,8	14,3	100,0
	Total	63	96,9	100,0	
Missing	System	2	3,1		
Total		65	100,0		

Table 6-13: Maintenance performed by the original developers of the system

In Table 6-13 we see the distribution of how often maintenance is done by the same employees, who developed the system. The results are not too different from what was found in 2003, with the majority then also stating that this is "Often true" (37,3%).

6.5 SYSTEM PORTFOLIO

We will now have a look at results from the part of the survey regarding the organization's system portfolio.

	N	Minimum	Maximum	Sum	Mean	Std. Deviation
Systems	60		90	471	7,85	13,824
Valid N (listwise)	60					

Table 6-14: Running main systems

From Table 6-14 we can see that the average organization has 7,9 main systems running. This is higher than in 2003 when the average was 4,5, but lower than the results from 1998 where the average was 9,6 and 1993 when the average was 10,3.

	N	Minimum	Maximum	Sum	Mean	Std. Deviation
Internal users	60		15000	33511	558,52	2000,305
Valid N (listwise)	60					

Table 6-15: Number of internal users

[1] When we calculate the mean value based on these categories, we weight all the categories with their mean value. The "More than 10 years" category is weighted 15. If A…E denotes the sum of the difference categories: (0,5*A + 2*B + 4,5*C + 8*D + 15*E) / (A + B + C + D + E)

The average number of internal users of the main systems is 559, and can be read from Table 6-15. The corresponding numbers from previous investigations are 541 in 1993, 498 in 1998 and 115 in 2003. However, looking at users alone is only so interesting. If we look at internal users in proportion to total number of employees, our investigation shows that 52% of the employees are users of the organizations' IT systems on average. In 1993 this number was 23%, in 1998 it was 76% and in 2003 the average was 64%.

The low value, 52%, from this investigation is a little surprising. It seems unlikely that the share of employees using IT systems is on the way down. However, these results may be because of outliers, and different types of organizations.

Even though there numbers are quite diverse, it seems safe to say that more employees use the organizations IT systems today, than what was the fact in 1993. Past that, we cannot really conclude on any increasing trend because of the varying results from 1998 and 2003.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	High accuracy, based on data	46	70,8	76,7	76,7
	Some accuracy, "estimated"	12	18,5	20,0	96,7
	Low accuracy, "good guess"	2	3,1	3,3	100,0
	Total	60	92,3	100,0	
Missing	System	5	7,7		
Total		65	100,0		

Table 6-16: Quality of numbers, Internal users

From Table 6-16 we see that the respondents found it relatively easy to give good answers to this question.

	N	Minimum	Maximum	Sum	Mean	Std. Deviation
External users	59		120000	225342	3819,36	15939,788
Valid N (listwise)	59					

Table 6-17: Number of external users

From Table 6-17 we can see that the average organization has 3819 external users of their main systems. These numbers have increased drastically since 2003 when the mean value was 198. However, this should not be surprising. It is a strong trend in almost any branch, to let customers interact directly with the organizations internal systems, through defined interfaces. Web shops and internet banking are examples of such solutions, offering functionality to external users.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	High accuracy, based on data	43	66,2	71,7	71,7
	Some accuracy, "estimated"	13	20,0	21,7	93,3
	Low accuracy, "good guess"	4	6,2	6,7	100,0
	Total	60	92,3	100,0	
Missing	System	5	7,7		
Total		65	100,0		

Table 6-18: Quality of numbers, External users

Table 6-18 shows that also the question about external users was something the respondents found fairly simple to answer precisely.

	N	Minimum	Maximum	Sum	Percent	Mean	Std. Deviation
0-1 year	59		10	50	11,4%	,85	1,627
1-3 years	59		15	112	25,5%	1,90	2,998
3-6 years	59		45	141	32,0%	2,39	6,178
6-10 years	59		25	108	24,5%	1,83	4,500
More than 10 years	59		5	29	6,6%	,49	1,073
Valid N (listwise)	59						

Table 6-19: Main systems age distribution

The age distribution we see in Table 6-19 is somewhat different from the distribution from 2003. In our investigation the most systems are between 3-6 years old (32,0%), while in 2003 the most systems were between 1-3 years old (37,4%). However, in 1998 the most systems were between 3-6 years old. The reason for all the new systems in 2003 is probably Y2K related. Many new replacement systems where deployed between 1998 and 2000.

If we calculate the average age of the systems, we get a mean value of 5 years. Both in 1993 and 1998 the average of the systems were also 5 years. In 2003 the systems were actually a little younger, with an average of 3,9 years. Also here we see how Y2K makes an impact.

	2008		2003		1998		1993	
	Sum	Percent	Sum	Percent	Sum	Percent	Sum	Percent
Dev. by the IT-dept.	53	12,0%	47	22,6%	132	26,8%		59%
Dev. by user group, intern	10	2,3%	4	1,9%	131	26,6%		1%
Dev. by external company	176	39,8%	73	35,1%	108	22%		12%
COTS, with many changes	100	22,6%	25	12,0%	47	9,6%		11%
COTS, with few changes	78	17,6%	57	27,4%	72	14,6%		17%
Webservice / Component[2]	25	5,7%	2	1,0%	2	0,4%		
Valid N (listwise)								

Table 6-20: The main systems' development process

From Table 6-20 we see that most of the running main systems are developed specifically for the organization, by an external company. It seems to be a trend that the IT department does not develop the systems, and instead buy COTS or have a external company develop the systems for them. How this affects development and maintenance depends on how the IT departments implements a change like this, and is not trivial. We will discuss this trend further in Chapter 9.

[2] In 1998 and 2003 this alternative was "Component based development". In 2008 we changed this to "Solutions using external Webservices".

	N	Minimum	Maximum	Mean	Sum	Std. Deviation
Dependent systems	60		90	5,98	359	13,830
Valid N (listwise)	60					

Table 6-21: Main systems' dependence on data from other systems

The average number of dependent systems in an organization is 5,98. This is severely higher than in 2003, where the number was 2,3. However, it might be more interesting to look at how many percent of an organizations main systems, are dependent on data from other systems. In 1993, the average percentage of dependent systems was 74%, in 1998 it was 60% and in 2003 the number was 50%. In this investigation, the percentage was 81,6%.

These variations is hard to explain, as the obvious theory would be that after time more systems are added to the application portfolio, and the dependency increases. However, the explanation might be that many systems were replaced during the nineties. Partially, or wholly, due to the problems connected with the Y2K. These are merely speculations, as we have no empirical data to back this up. However, one can imagine that when replacing systems at a big scale, the results would be a tidier application portfolio, and thus less dependency.

6.6 USE OF TECHNOLOGY

In this chapter we will look at what technology is in use in the different organizations. Programming languages in use, types of databases and whether the organization deploys a service-oriented architecture is investigated.

Language	N	# Org. that use the lang.	% that use	# Systems	% of systems
COBOL	60	3	5,0 %	23	4,5 %
Assembly	60	2	3,3 %	2	0,4 %
C	60	5	8,3 %	12	2,4 %
C++	60	22	36,7 %	89	17,5 %
C#	60	13	21,7 %	25	4, 9 %
Java	60	24	40,0 %	115	22,6 %
Script	60	19	31,7 %	34	6,7 %
4GL	60	12	20,0 %	61	12,0 %
Other	60	29	48,3 %	147	28,9 %
Valid N (listwise)	60				

Table 6-22: Use of programming languages

From Table 6-22 we see what programming languages are in use in the organizations. Java, C++, Java, Scripting-languages and C# are in use in many of the organizations, while COBOL, Assembly and C is obviously retreating, compared to results from 2003, 1998 and 1993.

To give a quick impression we have summed the systems using either COBOL or Assembly in previous investigations. In 1998 it was 48,0%, in 2003 the number was 13,0% and now we have come down to 4,9%. Lientz and Swanson reported in 1977 that 76,6% of the systems were written in either COBOL or Assembly.

As many as 48,3% uses languages not listed among the alternatives, but when we look at their comments, most of them have answered that they don't know exactly what programming languages are in use. This is probably because COTS packages are in use, and the respondent does not have insight to the technology of the software.

	N	Minimum	Maximum	Mean	Std. Deviation
# of languages	54	1,00	7,00	2,3889	1,61849
Valid N (listwise)	54				

Table 6-23: Number of different languages in use

From Table 6-23 we see that the average organization has systems developed in 2,4 different languages. In 1993 this number was 2,7, in 1998 it was 2,5 and in 2003 it was 2,0. It seems like this mean value stays relatively constant, regardless of what programming languages are in use.

DB Type	N	# Org. that use the DB	% that uses it	# Systems using the DB	% of systems
Hierarchical DB	59	4	6,8 %	9	1,6 %
Network DB	59	10	16,9 %	119	21,8 %
Relational DB	59	41	59,4 %	289	52,8 %
Object-oriented DB	59	18	30,5 %	55	10,1 %
Other DB	59	7	11,9 %	75	13,7 %
Valid N (listwise)	59				

Table 6-24: Use of database types

We see from Table 6-24 that relational databases are the most common type of database. 59,4% of the organizations use this type of databases. In 1998 the number was 61,3% and in 2003 it was 54,5%. When it comes to network databases, as many has 21,8% of the systems use this type. The number was 40,6% in 1998, but only 10,7% in 2003. We think it is safe to say that these outcomes are pretty random, and based on a minority of the participants, with very many systems using this kind of databases.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Don't know	6	9,2	10,2	10,2
	Not used	16	24,6	27,1	37,3
	Seldom used	18	27,7	30,5	67,8
	Sometimes used	11	16,9	18,6	86,4
	Mainly used	4	6,2	6,8	93,2
	Always used	4	6,2	6,8	100,0
	Total	59	90,8	100,0	
Missing	System	6	9,2		
Total		65	100,0		

Table 6-25: Use of SOA in the organization

From Table 6-25 we see that most of the organizations don't use SOA (24,6%), or seldom use SOA (27,7%). This is the first time questions regarding SOA is a part of these surveys, hence we do not have any data to compare our results to.

6.7 DEVELOPING NEW SYSTEMS

In this chapter we will look at reasons for developing new systems, methods and tools used in the development process and to what extent organizations are able to re-use their specifications, code and design.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	No plan	22	33,8	38,6	38,6
	A wish/intent	18	27,7	31,6	70,2
	A structured plan	6	9,2	10,5	80,7
	Already implementing	11	16,9	19,3	100,0
	Total	57	87,7	100,0	
Missing	System	8	12,3		
Total		65	100,0		

Table 6-26: Plan to deploy SOA

We see from Table 6-26 that the two largest categories are "No plan to implement SOA" and "A wish to implement SOA". Together these two categories consist of 70,2 % of the responding organizations.

The Gartner reported, mentioned under Chapter 2.2, indicates that the share of companies wanting to implement SOA fell from 53% in 2007, to 25% in 2008. Our number is clearly higher, and comes to 42,1 % if we sum together those that have a "wish/intent" and those that have a "structured plan" to implement SOA.

	N	Minimum	Maximum	Sum	Mean	Std. Deviation
Total systems	57		10	87	1,53	1,681
Replacement systems	57		6	57	1,00	1,134
Valid N (listwise)	57					

Table 6-27: Systems under development

Table 6-27 tells us how many new systems are being developed, and how many of these are replacement systems. The numbers differ slightly from earlier years, but the most interesting aspect is the replacement systems versus total systems ratio. In our investigation 65,5% of the systems being developed, are regarded replacement systems. In 1993 the percentage was 48%, in 1998 it was 56% and in 2003 the percentage was 60%.

It seems to be a increasing trend here, which seems fair. After many years of using information systems, more and more functionality will be covered by already existing systems. Development of new systems may therefore more often be categorized as replacement systems.

	N	Minimum	Maximum	Sum	Percent	Mean	Std. Deviation
0-1 year	57					,00	,000
1-3 years	57		2	10	12,8 %	,18	,428
3-6 years	57		2	15	19,2 %	,26	,583
6-10 years	57		5	26	33,3 %	,46	,888
More than 10 years	57		2	7	9,0 %	,12	,426
Valid N (listwise)	57						

Table 6-28: Age distribution of replaced systems

From Table 6-28 we see that most systems being replaced, are between 6 and 10 years old average (33,3%). When we look at numbers from 2003, most systems being replaced was only between 3 and 6 years old average (50,0%). These numbers give us a average age of 6,9 years for the replaced systems. In 1993 the average age of systems being replaced was 8,5 years, in 1998 it was 10,5 years and in 2003 it was 5,5 years.

These numbers may implicate that the lifetime of information systems is decreasing. Our investigations shows a higher average than the numbers from 2003, but results from our investigation and the ones from 2003 are both lower than the results from 1998 and 1993.

	2008		2003		1998		1993
Reason	N	Mean	N	Mean	N	Mean	Mean
Difficult to maintain existing system	43	3,7	48	2,9	30	3,1	3,7
Difficult to operate existing system	42	3,3	48	2,6	30	2,3	3,7
Difficult to use existing system	41	2,9	48	2,6	30	2,1	3,0
Alternative software packages exist	42	2,6	48	2,8	30	2,1	2,4
Alternative application generator exist	41	1,9	48	1,9	30	1,6	1,8
Transition to SOA	41	2,4					
Transition to new architecture (not SOA)	41	2,6	48	3,0	30	2,9	3,7
Standardization	43	3,0	48	3,3	30	3,4	3,0
Integration with other systems	42	3,7	48	3,4	30	3,2	3,9
Other	41	2,1	10	2,2			
Valid N (listwise)	27						

Table 6-29: Reasons for replacing systems

Respondents were asked to grade the reasons for replacing systems seen in Table 6-29, after importance. (1 is lowest importance, 5 is highest) The table present mean values for all the four investigations. This time, the difficulty with maintaining the existing systems has been ranked the most important reason, together with the need to integrate with other systems.

From the table we can see the results from the other investigations. There does not seem to be any significant change in the reasons for replacing systems. However, it seems like our addition to the list, “Transition to SOA”, might have taken some importance away from the “Transition to new architecture” category. This seems only natural.

		Specification	Design	Code
N	Valid	57	57	57
	Missing	8	8	8
	Mean	2,58	1,98	1,65

Table 6-30: Comparing re-use of specification, code and design

We will now look at the organizations ability to re-use specification, design and code. The respondents were asked to grade their own re-use, from 1 to 5 within the three categories. From Table 6-30 we can see that re-use of specification is most common on a general basis, but that all the mean values are quite low. In Table 6-30, Table 6-31 and Table 6-32 the frequencies of the different answers are given.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	No	17	26,2	29,8	29,8
	Little	10	15,4	17,5	47,4
	Some	13	20,0	22,8	70,2
	Considerably	14	21,5	24,6	94,7
	Much	3	4,6	5,3	100,0
	Total	57	87,7	100,0	
Missing	System	8	12,3		
Total		65	100,0		

Table 6-31: Re-use of specifications

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	No	26	40,0	45,6	45,6
	Little	16	24,6	28,1	73,7
	Some	8	12,3	14,0	87,7
	Considerably	4	6,2	7,0	94,7
	Much	3	4,6	5,3	100,0
	Total	57	87,7	100,0	
Missing	System	8	12,3		
Total		65	100,0		

Table 6-32: Re-use of design

In our investigation 47,4% say they have no or little re-use of specification, and as much as 73,7% say they have no or little re-use of system design. In the previous investigations re-use of design and specifications was not separated, but asked as one question. It is therefore a little difficult to compare our results directly, but we will try.

In 1993 52% reported no or little re-use of specifications and design, in 1998 the number was 53% and in 2003 it was 65,9%. Our number is 47,4%, but if we have asked about specifications and design together, we might have gotten a higher number. We are therefore content saying the results seem to be approximately the same as earlier years.

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	No	39	60,0	68,4	68,4
	Little	9	13,8	15,8	84,2
	Some	2	3,1	3,5	87,7
	Considerably	4	6,2	7,0	94,7
	Much	3	4,6	5,3	100,0
	Total	57	87,7	100,0	
Missing	System	8	12,3		
Total		65	100,0		

Table 6-33: Re-use of code

As much as 68,4% of the respondents, who answered the question, feel like their organization is not re-using code at all. 84,2% re-use no or little of their code. In 1993 86% re-used no or little code, in 1998 the number was 74% and in 2003 the number was 36,9%.

Even though our result is much higher than in 2003, both the two latter investigations show more re-use of code than the investigations from 1993 and 1998.

6.8 METHODS AND TOOLS

In this chapter we will present the results from the part of the survey, related to the use of methods, tools and organization controls[3] to develop and maintain the organization's IT-systems.

	2008		2003		1998	
Phase	Yes	Percent	Yes	Percent	Yes	Percent
Planning	17	31,5 %	20	43,5%	18	34,0%
Analysis	13	24,1 %	11	23,9%	16	30,2%
Requirement Specification	*26*	*48,2 %*	*26*	*56,5%*	*27*	*50,9%*
Design	18	33,3 %	21	45,7%	21	39,6%
Implementation	21	38,9 %	24	52,2%	23	43,4%
Testing	24	44,4 %	25	54,3%	18	34,0%
Deployment	18	33,3 %	15	32,6%	14	26,4%
Operation	22	40,7 %	17	37,0%	17	32,1%
Maintenance	16	29,6 %	13	28,3%	16	30,2%
Project management	20	37,0 %	16	34,8%	22	41,5%

Table 6-34: Use of pre-defined methods in the systems lifecycle

From Table 6-34 we see that Requirement Specification is the phase were most of the organizations use a pre-defined method (48,15%). Although the percentages differ some, this is the same result as in 1998 (50,9%) and 2003 (56,5%). Worth mentioning is also that use of pre-defined methods in maintenance has been relatively low in all the investigations. Will discuss how this may affect maintenance efficiency in Chapter 9.

[3] By organizational controls we mean different procedures and routines established to control different aspects of the software lifecycle.

	2008		2003		1998	
Phase	Yes	Percent	Yes	Percent	Yes	Percent
Planning	13	24,07 %	15	65,2%	2	28,6%
Analysis	10	18,52 %	9	39,1%	4	57,1%
Requirement Specification	13	24,07 %	17	73,9%	5	71,4%
Design	17	31,48 %	15	65,2%	7	100,0%
Implementation	20	37,04 %	15	65,2%	6	85,7%
Testing	*23*	*42,59 %*	*16*	*69,6%*	*2*	*28,6%*
Deployment	13	24,07 %	10	43,5%	2	28,6%
Operation	10	18,52 %	11	47,8%	1	14,3%
Maintenance	13	24,07 %	7	30,4%	2	28,6%
Project management	17	31,48 %	12	52,2%	2	28,6%

Table 6-35: Use of development tools in the systems lifecycle

We see from Table 6-35 that Testing is the phase where most of the organizations use development tools, followed by Implementation. This is slightly different from 2003, when Requirement Specification came first, followed by Testing and Implementation. Analysis and Operation is found at the bottom, while Maintenance and Analysis where lowest on the ranking in 2003. Another interesting aspect is that the total use of system development tools seems to have dropped noticeably. If we take the average use of tools from all the different phases, we get 27,59%. In 2003 the number was 55,21% and in 1998 it was 47,15%.

	N	Minimum	Maximum	Sum	Mean	Std. Deviation
Years of experience	41		20	130	3,17	3,584
Valid N (listwise)	41					

Table 6-36: Experience with system development tools

Table 6-36 tells us that the average organization has 3,17 years of experience with the system development tools in use. In 2003 this average was 4,2 and in 1998 it was 3,1.

	N	Minimum	Maximum	Sum	Mean	Std. Deviation
Systems	40		20	98	2,45	3,883
Valid N (listwise)	40					

Table 6-37: Number of main-systems that are supported by development tools

From Table 6-37 we see that the average organization has 2,45 systems that are supported by system development tools. In 2003 the number was 1,9 and in 1998 it was 2,0.

Organizational controls	N	Use	%	08	03	98	93
Same routines for change requests if they come from the IT dept. and the user groups	54	32	59,3 %	1	7	7	12
All changes are tested before the system goes into production	54	31	57,4 %	2	1	3	4
Change requests are classified by type and priority	54	30	55,6 %	3	2	2	9
Personnel costs related to operation and maintenance are charged to the user groups	54	27	50,0 %	4	10	12	8
All change requests goes through an analysis of consequence and cost	54	22	40,8 %	5	4	8	2
Except from critical errors all changes are gathered and periodically implemented	54	22	40,8 %	6	12	5	5
Users requesting changes are notified both if the request is accepted or denied	54	22	40,8 %	7	5	6	11
All user requirements are logged	54	20	37,0 %	8	6	1	1
Equipment related to operation and maintenance are charged to the user groups	54	20	37,0 %	9	11	11	7
All changes to the information systems are documented	54	14	26,0 %	10	3	4	3
A formal control of the information systems is performed periodically	54	7	13,0 %	11	8	10	6
When changes are accepted, related documents are reviewed and updated	54	7	13,0 %	12	9	9	10

Table 6-38: Use of organizational controls

In Table 6-38 we see the share of organizations using the different organizational controls. We have also listed the ranking from the previous investigations. When we compare the results from our investigations to ranks from earlier years, we see some important changes. It seems like the users are more heavily involved now than before, and the IT department's status has changed some. We will now see how we can claim this.

"Same routines for change requests if they come from the IT department and the user groups" is now the routine with highest implementation rate. Earlier years this has not even been in the top five. Also it seems like organizing the IT department as a supplier, and the other departments as customers, has become way more usual. "Personnel costs related to operation and maintenance are charged to the user groups" has moved from the bottom of the list in earlier years, and is now the fourth most implemented routine. Letting the departments act as "customers" also includes the users more in the IT activities.

Another noteworthy result is the importance of documentation. In previous investigations routines regarding documentation have been ranked very high. "All changes to the information systems are documented" has been ranked as three and four. In our investigation this routine has dropped to 10. Also "All user requirements are documented" has dropped from third, first and sixth down to eight on our list.

6.9 PROBLEM AREAS WITHIN MAINTENANCE

We will now present results from the last question in the survey, regarding what problem areas related to maintenance, the respondents feel are important for their organization.

Problem Area	1	2	3	4	5	N	Mean
Quality of system documentation	5,7 %	9,4 %	35,8 %	32,1 %	17,0 %	53	3,45
Quality of the original system	1,9 %	20,8 %	37,7 %	35,8 %	3,8 %	53	3,19
Tight budgets	3,8 %	28,3 %	30,2 %	24,5 %	13,2 %	53	3,15
Unrealistic user expectations	1,9 %	26,9 %	32,7 %	32,7 %	5,8 %	52	3,13
Turnover in maintenance staff	11,3 %	20,8 %	26,4 %	28,3 %	13,2 %	53	3,11
Available maintenance staff	3,8 %	26,4 %	32,1 %	32,1 %	5,7 %	53	3,09
Competing demand for maintenance staff	5,8 %	26,9 %	26,9 %	36,5 %	3,8 %	52	3,06
Untrained users	5,8 %	28,8 %	34,6 %	23,1 %	7,7 %	52	2,98
Skills of the maintenance staff	0,0 %	34,0 %	39,6 %	24,5 %	1,9 %	53	2,94
Users don't understand the system	7,7 %	38,5 %	23,1 %	26,9 %	3,8 %	52	2,81
User demand for enhancements	5,7 %	34,0 %	39,6 %	20,8 %	0,0 %	53	2,75
Turnover in usergroup	7,7 %	42,3 %	25,0 %	23,1 %	1,9 %	52	2,69
Changes in hardware and systemsoftware	9,4 %	32,1 %	41,5 %	15,1 %	1,9 %	53	2,68
System fails during runtime	11,3 %	41,5 %	20,8 %	20,8 %	5,7 %	53	2,68
No use of programming standards	15,4 %	32,7 %	32,7 %	17,3 %	1,9 %	52	2,58
Missing interest from users	17,6 %	27,5 %	39,2 %	11,8 %	3,9 %	51	2,57
No support from management	18,9 %	41,5 %	17,0 %	15,1 %	7,5 %	53	2,51
Integrity between data in applications	15,4 %	38,5 %	30,8 %	13,5 %	1,9 %	52	2,48
Producticity of the maintenance staff	11,8 %	52,9 %	27,5 %	5,9 %	2,0 %	51	2,33
Reliablity of hardware and systemsoftware	19,2 %	50,0 %	21,2 %	7,7 %	1,9 %	52	2,23
Motivation of the maintenance staff	26,9 %	36,5 %	26,9 %	7,7 %	1,9 %	52	2,21
Computerperformance	29,4 %	39,2 %	23,5 %	7,8 %	0,0 %	51	2,10
Requirements regarding datastorage	34,6 %	42,3 %	17,3 %	5,8 %	0,0 %	52	1,94
Other	87,0 %	4,3 %	8,7 %	0,0 %	0,0 %	23	1,22
Valid N (listwise)						23	

Table 6-39: Problem areas, sorted by mean value

From Table 6-39 we can see what are regarded as the most important problem areas on an average basis, considering all the participating organizations. Quality of documentation, quality of the original system and tight budgets are regarded the

biggest problems. Problems regarding performance, data storage and motivation of the staff are considered to be of less importance.

Lientz and Swanson reported in 1977 that the knowledge of the users, efficiency of the programmers and the quality of the original system was the most important problem areas [25]. In 1990 Nosek and Palvia reported that availability and efficiency of the programmers were the biggest problems [26], and in 1993 we found that turnover in maintenance personnel and quality of documentation were concerned most problematic. In 1998 quality of documentation, turnover in personnel and availability of the personnel was named as the most important problems. In 2003 the quality of the original system, tight budgets and quality of system documentation was considered to be the biggest problems.

Quality of system documentation	**2008**	**1**
	2003	3
	1998	1
	1993	2
	1977	2
Quality of the original system	**2008**	**2**
	2003	1
	1998	5
	1993	1
	1990	5
	1977	4
Tight budgets	**2008**	**3**
	2003	2
Unrealistic user expectations	**2008**	**4**
Turnover in maintenance personnel	**2008**	**5**
	2003	5
	1998	2
	1993	3

Table 6-40: Problem areas, this and previous investigations

In Table 6-40 we see the top five problem areas from this investigation, and compare their ranks from other investigations.

It is interesting to see how quality of system documentation is mentioned as one of the biggest problems, year after year. Still as little as 25% of the organizations in our investigations say they have established organizational routines for documenting all changes to the systems (Table 6-38).

Tight budgets also seem to be big problem in 2003 as well as in our investigation. Quality of the original system has been regarded a big problem in many of the investigations, and is also the second most important problem of our study.

7 HYPOTHESIS-TESTING

In this chapter we will be testing the hypotheses, defined in Chapter 4. Before we begin testing our hypotheses, we will remove some outliers from our data. Organizations that only do development, and no maintenance is regarded to be outside our scope. The same goes for organizations that do only maintenance, and no development.

7.1 NORMALITY TESTS

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistic	df	Sig.	Statistic	df	Sig.
Total share of maintenance	,107	59	,090	,975	59	,269
Total share of development	,126	59	,020	,925	59	,001
Share of maintenance (isolated)	,108	59	,084	,942	59	,007
Share of development (isolated)	,108	59	,084	,942	59	,007
Share of application portfolio evolution.	,107	59	,091	,938	59	,005
Share of application portfolio upkeep	,107	59	,091	,938	59	,005

a. Lilliefors Significance Correction

Table 7-1: Test of normality

Table 7-1 shows normality tests for different variables. The only variable that may be normally distributed is "Total share of maintenance", with significance levels of ,090 and ,269 in the Kolmogorov-Smirnov and Shapiro-Wilk tests.

7.2 MAINTENANCE AND DEVELOPMENT

H1: There is no difference in the amount of time used on maintenance and development, when we only look at maintenance and development.

Ranks

		N	Mean Rank	Sum of Ranks
Development - Maintenance	Negative Ranks	46[a]	28,88	1328,50
	Positive Ranks	9[b]	23,50	211,50
	Ties	4[c]		
	Total	59		

a. Development < Maintenance

b. Development > Maintenance

c. Development = Maintenance

Test Statistics[b]

	Development - Maintenance
Z	-4.684[a]
Asymp. Sig. (2-tailed)	,000

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 7-2: Maintenance vs. development (Isolated)

H1 is rejected. We see from Table 7-2 that more time is spent on maintenance than on development in 46 of 59 cases.

H2: There is no difference in the amount of time used on maintenance and development.

Ranks

		N	Mean Rank	Sum of Ranks
Development - Maintenance	Negative Ranks	46[a]	27,71	1274,50
	Positive Ranks	9[b]	29,50	265,50
	Ties	6[c]		
	Total	61		

a. Development < Maintenance

b. Development > Maintenance

c. Development = Maintenance

Test Statistics[b]

	utvpros - vedpros
Z	-4.234[a]
Asymp. Sig. (2-tailed)	,000

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 7-3: Maintenance vs development

H2 is rejected. We see from Table 7-3 that in 46 of 61 cases, more time is spent on maintenance than on development.

H3: There is no difference between the time used on application portfolio upkeep and traditional maintenance, when we look at development and maintenance only.

Ranks

		N	Mean Rank	Sum of Ranks
Upkeep - Maintenance	Negative Ranks	25[a]	25,30	632,50
	Positive Ranks	20[b]	20,13	402,50
	Ties	14[c]		
	Total	59		

a. Upkeep < Maintenance

b. Upkeep > Maintenance

c. Upkeep = Maintenance

Test Statistics[b]

	funkved - vedprost
Z	-1.299[a]
Asymp. Sig. (2-tailed)	,194

a. Based on positive ranks.

b. Wilcoxon Signed Ranks Test

Table 7-4: Application portfolio upkeep vs maintenance (isolated)

H3 is not rejected. The share of maintenance is bigger than upkeep in 25 of 59 cases, while upkeep is bigger in 20 cases. We see from Table 7-4 that the significance level of the difference is 0,194, therefore we cannot reject H3.

H4: There is no difference between the time used on application portfolio evolution and traditional development, when we look at development and maintenance only.

Ranks

		N	Mean Rank	Sum of Ranks
Evolution - Development	Negative Ranks	20[a]	20,13	402,50
	Positive Ranks	25[b]	25,30	632,50
	Ties	14[c]		
	Total	59		

a. Evolution < Development

b. Evolution > Development

c. Evolution = Development

Test Statistics[b]

	funktutv - utvprost
Z	-1.299[a]
Asymp. Sig. (2-tailed)	,194

a. Based on negative ranks.

b. Wilcoxon Signed Ranks Test

Table 7-5: Application portfolio evolution vs development (Isolated)

H4 is not rejected. We see from Table 7-5 that even though evolution is more time consuming in more cases than development, 25 vs 20, the significance level is 0,194, and we cannot reject H4.

H5: There is no difference between the time used on application portfolio evolution and application portfolio upkeep.

Ranks

		N	Mean Rank	Sum of Ranks
Upkeep - Evolution	Negative Ranks	10[a]	27,50	275,00
	Positive Ranks	47[b]	29,32	1378,00
	Ties	2[c]		
	Total	59		

a. Upkeep < Evolution

b. Upkeep > Evolution

c. Upkeep = Evolution

Test Statistics[b]

	funkved - funktutv
Z	-4.384[a]
Asymp. Sig. (2-tailed)	,000

a. Based on negative ranks.

b. Wilcoxon Signed Ranks Test

Table 7-6: Application portfolio upkeep vs application portfolio evolution

H5 is rejected. We see from Table 7-6 that more time is spent on application portfolio upkeep, than on application portfolio evolution in 47 of 59 cases.

7.3 TYPE OF ORGANIZATION

H6: There is no difference in the amount of maintenance-work between organizations with many employees and organizations with fewer employees.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Employees	Correlation Coefficient	-,219	,112	,095
		Sig. (2-tailed)	,090	,397	,473
		N	59	59	59

Table 7-7: Maintenance, based on numbers of employees

H6 is not rejected. In Table 7-7 we see the results from a Spearman test, to decide if any of the maintenance variables correlate with the number of employees. We see that there are no significant correlations.

H7: There is no difference in the amount of maintenance-work between organizations where maintenance is often performed by the people who developed the system, and organizations where maintenance is rarely performed by the people who developed the system.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Maintenance done by dev.	Correlation Coefficient	-,123	,082	,117
		Sig. (2-tailed)	,353	,543	,385
		N	59	59	59

Table 7-8: Maintenance, based on maintenance done by developers

H7 is not rejected. We see from Table 7-8 that none of the maintenance variables correlates with how often maintenance is done by the original developers of the system.

7.4 IMPORTANCE OF IT

H8: There is no difference in the amount of maintenance-work between organizations in which the size of the IT-department compared to the total number of employees is large and the organizations where the size of the IT-department compared to the total number of employees is small.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Employees IT / Total emp.	Correlation Coefficient	,182	-,206	-,072
		Sig. (2-tailed)	,169	,123	,594
		N	59	57	57

Table 7-9: Maintenance, based on employees in IT and in total

H8 is not rejected. We see from Table 7-9 that there is no significant correlation between maintenance and the proportion between IT-employees and total employees.

H9: There is no difference in the amount of maintenance-work between organizations in which there are many system-developers in proportion to total number of internal users, and organizations with few system-developers in proportion to total number of internal users.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Developers / Users	Correlation Coefficient	,252	-,264	-.453**
		Sig. (2-tailed)	,059	,051	,001
		N	57	55	55

**. Correlation is significant at the 0.01 level (2-tailed).

Table 7-10: Maintenance, based on developers and internal users

H9 is rejected. We see from Table 7-10 that the "developers per user" ratio correlates with time spent on application portfolio upkeep. The correlation coefficient is negative, which means that the more developers per user the organization has, less time is spent on upkeep.

H10: There is no difference in the amount of maintenance-work between organizations in which there are many system-developers in proportion to total number of employees in the IT department, and organizations with few system-developers in proportion to total number of employees in the IT department.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Developers / IT-employees	Correlation Coefficient	.272*	-,233	-.473**
		Sig. (2-tailed)	,045	,093	,000
		N	55	53	53

*. Correlation is significant at the 0.05 level (2-tailed).

**. Correlation is significant at the 0.01 level (2-tailed).

Table 7-11: Maintenance, based on developers and IT-employees in total

H10 is rejected. We see from Table 7-11 that there is clearly a relation between how much time is spent on upkeep, and how many of the organizations IT-employees are actually developers. The correlation between Maintenance and this ratio is significant at the 0.05 level, but here the correlation coefficient is actually positive. This indicates that organizations with many developers per IT-employee spend more time on traditional maintenance, but less time on application portfolio upkeep. In other words, they are in a greater degree able to spend their resources on introducing new functionality.

7.5 CONSULTANTS AND EMPLOYEES

H11: There is no difference in the amount of maintenance-work between organizations with hired IT-consultants and organizations without any hired IT-consultants.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Consultants	Correlation Coefficient	,136	-,204	-,109
		Sig. (2-tailed)	,305	,129	,418
		N	59	59	59

Table 7-12: Maintenance, based on number of consultants

H11 is not rejected. We see from Table 7-12 that the number of consultants does not correlate with any of the maintenance variables.

H12: There is no difference in the amount of maintenance work between organizations with higher turnover amongst developers, and organizations with lower turnover amongst developers.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Average exp. from IT-dept.	Correlation Coefficient	,135	,284	-,057
		Sig. (2-tailed)	,468	,121	,763
		N	31	31	31

Table 7-13: Maintenance, based on developers experience from the IT-dept.

H12 is not rejected. We see from Table 7-13 that there is no significant correlation between the developers' average experience from the IT department, and any of the maintenance variables.

7.6 COMPLEXITY OF THE PORTFOLIO

H13: There is no difference in the amount of maintenance-work between organizations with many main systems and organizations with fewer main systems.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Main systems	Correlation Coefficient	-,201	-,101	,158
		Sig. (2-tailed)	,133	,462	,249
		N	57	55	55

Table 7-14: Maintenance, based on the number of main systems

H13 is not rejected. From Table 7-14 we see that the number of main systems does not correlate with any of the maintenance variables.

H14: There is no difference in the amount of maintenance-work between organizations with many internal users and organizations with fewer users.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Internal users	Correlation Coefficient	-,142	,140	,045
		Sig. (2-tailed)	,290	,306	,745
		N	57	55	55

Table 7-15: Maintenance, based on number of internal users

H14 is not rejected. We see from Table 7-15 that there is no significant correlation between the number of internal users and any of the maintenance variables.

H15: There is no difference in the amount of maintenance-work between organizations with main-systems with high age average, and organizations with main-systems with low age average.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Age of systems	Correlation Coefficient	-,038	,175	-,045
		Sig. (2-tailed)	,783	,205	,748
		N	56	54	54

Table 7-16: Maintenance, based on average age of main systems

H15 is not rejected. From Table 7-16 we see that there is no consequent influence on share of maintenance, from average age of systems.

H16: There is no difference in the amount of maintenance-work between organizations with many main-systems which are dependent on data from other systems, and organizations with few main-systems which are dependent on data from other systems.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Dependent systems	Correlation Coefficient	-,193	-,142	,141
		Sig. (2-tailed)	,150	,300	,304
		N	57	55	55

Table 7-17: Maintenance, based on the number of dependent systems

H16 is not rejected. We see from Table 7-17 that the number of dependent systems does not influence the share of maintenance in the organizations.

H17: There is no difference in the amount of maintenance-work between organizations that use many different programming-languages, and organizations that use fewer different programming-languages.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	No. of languages	Correlation Coefficient	,049	-,128	-,015
		Sig. (2-tailed)	,733	,370	,918
		N	52	51	51

Table 7-18: Maintenance, based on total number of languages in use

H17 is not rejected. We see from Table 7-18 that there is no correlation between the amount of programming languages in use, and the time spent on maintenance.

7.7 USE OF METHODS AND TOOLS

H18: There is no difference in the amount of maintenance-work between organizations that use pre defined methods throughout the systems lifecycle, and the organizations that do not use this.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Use methods	Correlation Coefficient	,095	-,121	,129
		Sig. (2-tailed)	,508	,407	,376
		N	51	49	49

Table 7-19: Maintenance, based on use of methods

H18 is not rejected. We see from Table 7-19 that there is no correlation between the organizations' use of methods and their maintenance. The "Use of methods" is represented by counting how many phases of the software lifecycle are supported by pre defined methods.

H19: There is no difference in the amount of maintenance-work between organizations that use system development tools, and the organizations that do not use this.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Use of tools	Correlation Coefficient	,268	-,180	-,167
		Sig. (2-tailed)	,057	,216	,251
		N	51	49	49

Table 7-20: Maintenance, based on use of tools

H19 is not rejected. We see from Table 7-20 that there is no correlation between the organizations' use of tools and their maintenance. The "Use of tools" is represented by counting how many phases of the software lifecycle are supported by tools.

H20: There is no difference in the amount of maintenance-work between organizations that use defined organizational controls, and organizations that do not use this.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Org. controls	Correlation Coefficient	,161	-,108	-,003
		Sig. (2-tailed)	,258	,461	,986
		N	51	49	49

Table 7-21: Maintenance, based on use of org. controls

H20 is not rejected. We see from Table 7-21 that there is no correlation between the organizations' use of organizational controls and their maintenance.

7.8 OUTSOURCING

H21: There is no difference in the amount of maintenance-work between organizations that outsources much of the total IT activity, and organizations that outsource less of the total IT activity.

			Maintenance	Maintenance (Isolated)	App. Portfolio upkeep
Spearman's	Outsourcing totalt	Correlation Coefficient	-,043	,211	,191
		Sig. (2-tailed)	,740	,108	,147
		N	61	59	59

Table 7-22: Maintenance, based on total share of outsourcing

H21 is not rejected. We see from Table 7-22 that no significant effect from outsourcing can be seen on the maintenance variables.

H22: The use of outsourcing is not dependent on the size of the company.

			Total no. of employees
Spearman's	Outsourcing in total	Correlation Coefficient	,022
		Sig. (2-tailed)	,863
		N	65
	outsourcing development	Correlation Coefficient	,005
		Sig. (2-tailed)	,971
		N	65
	Outsourcing maintenance	Correlation Coefficient	,050
		Sig. (2-tailed)	,691
		N	65
	Outsourcing operation	Correlation Coefficient	-,060
		Sig. (2-tailed)	,634
		N	65
	Outsourcing support	Correlation Coefficient	,048
		Sig. (2-tailed)	,705
		N	65
	Outsourcing management	Correlation Coefficient	-,073
		Sig. (2-tailed)	,561
		N	65

Table 7-23: Outsourcing, based on total number of employees

H22 is not rejected. We see from Table 7-23Table 7-23 that there is no correlation between total number of employees in the organizations and the different outsourcing variables.

7.9 SERVICE ORIENTED ARCHITECTURE

H23: There is no difference in the amount of maintenance-work between organizations that has deployed service oriented architecture and organizations that has not deployed service oriented architecture.

	>=3			**<3**				
	N	Mean	SD	N	Mean	SD	Mean diff.	Sig. (2-tailed)
Maintenance	38	33,11	13,25	18	29,68	17,29	3,43	0,461
Maintenance (isolated)	36	64,84	15,26	18	64,33	25,10	0,52	0,936
App. portfolio upkeep	36	66,23	16,64	18	61,14	23,68	5,09	0,419

Table 7-24: Maintenance, based on deployment of SOA

H23 is not rejected. The respondents were asked to what extent their organization was using service oriented architecture, on a scale from 1 to 5. We see from **Feil! Fant ikke referansekilden.** that whether the organization answered 3 or above, or below 3, does not have any effect on their maintenance.

H24: The use of service oriented architecture is not dependent on the size of the company.

			Use of SOA	Plan to implement SOA
Spearman's	Total no. of employees	Correlation Coefficient	,188	,176
		Sig. (2-tailed)	,154	,189
		N	59	57

Table 7-25: SOA, based on total number of employees

H24 is not rejected. We see from Table 7-25 that the total number of employees does not significantly correlate with either use of SOA in the present, or the plan to implement SOA in the future.

7.10 TIME PERSPECTIVE

H25: There is no difference in the amount of maintenance-work between organizations that participated in this investigation, and the investigation performed in 2003.

	2003			**2008**				
	N	Mean	SD	N	Mean	SD	Mean. diff	Sig. (2-tailed)
Maintenance	52	35,01	13,791	61	31,13	15,928	3,888	0,172
Maintenance (isolated)	52	65,88	21,404	59	65,75	21,856	0,129	0,975
App. portfolio upkeep	52	61,25	20,038	59	62,92	21,141	-1,663	0,672

Table 7-26: Maintenance, based on year

H25 is not rejected. We see from Table 7-26 that there is no significant difference in the amount of maintenance work reported in the two investigations.

H26: There is no difference in the share of total new systems that is classified as replacement systems in our survey and what were reported in 2003.

	2003			**2008**				
	N	Mean	SD	N	Mean	SD	Mean. diff	Sig. (2-tailed)
Replacement / New systems	25	0,57	0,56	42	0,6770	15,38	-0,11	0,354

Table 7-27: Share of replacement systems, based on year

H26 is not rejected. We see from Table 7-27that there is no significant difference in the share of replacement systems between the two investigations.

H27: The average age of a system that is being replaced, is the same in our survey and what were reported in 2003.

	2003			2008				
	N	Mean	SD	N	Mean	SD	Mean. diff	Sig. (2-tailed)
Avg. age of systems being replaced	16	5,38	4,27	37	6,79	3,85	-1,41	0,241

Table 7-28: Average age og systems being replaced, based on year

H27 is not rejected. Even though the mean value from 2008 is clearly higher than the mean value from 2003, the values are too scattered to make the difference significant.

8 CASE RESULTS

In this chapter we will present summaries of the different interviews performed in our case study. The full version of the interviews can be found in the appendix.

8.1 INTERVIEW A

The participating organization in this interview is a public institution. It has 4500 employees, and an IT department has total number of 120 employees, corresponding to 95 man-labor years.

13% of the organizations resources are spent on maintenance, 6% is spent on development. Isolated this gives 68% of traditional maintenance. Application portfolio upkeep sums to 79%. The organization outsources 5% of its total IT activity.

8.1.1 IT ACTIVITY, IT GOVERNANCE AND IT STRATEGY

This organization has a centralized IT department, with around 60 employees working 100%. In addition there are some part-time workers connected to the IT department. Furthermore, the other departments have their own IT employees, which perform IT tasks to various extents. Some of the departments only have "customer competence" and buy all IT-services from the centralized IT department. Other departments operate their own systems, together with buying services from the IT department. Every department has a service contract with the centralized IT department. This organizational form has been decided by the board, and the organization is still working on implementing it.

The IT director admits there has been a lack of IT governance in the organization, and that they are still working to improve this. In 2005 a committee was formed, to investigate and improve IT processes in the organization. One of the most important changes suggested by this committee was to introduce a user panel consisting of "non-IT personnel" from the different departments. This panel shall be included in all strategic decisions regarding central information systems in the organization. The goal is to include the users more in these decisions, and the IT director welcomes this as an improvement.

When it comes to IT strategy, the organization has no overall IT strategy. It has been said that the IT strategy is to "support the overall strategy". The only IT specific strategy that is formulated is the strategy of the IT department. The IT director describes this as "nothing fancy", just a formulated statement of what their customers and owners expects from them. This strategy is meant to ease the decision-making, when projects have to be prioritized.

The organization has 12 system developers. 50% of their time is spent on developing, and the other 50% are spent on application administration.

8.1.2 SUCCESS CRITERIA AND CHALLENGES

Since the organization has many employees, in addition to other users of their systems, operation is the most important of all their tasks. The IT director states that operating their existing software and hardware consumes most of their resources, and their overall goal is to get "more out of the budget". All their investments are considered with regards to how it will affect the economical bottom line. Success is defined as "getting the work done, with the resources available".

The IT director considers thoroughly thinking through organizational effects of changes to the IT infrastructure to be the biggest challenge, related to development and maintenance. Although they are well aware of these aspects of software development, you sometimes lose focus of it, and get more concerned with technical problems.

When the different departments sign service contracts with the IT department, they distinguish between operation and maintenance. If any of the departments need any development done, this is seen as a new service, and is charged to the department. The developers also report what time they spend of maintenance, development and administration. This way you can say that the organization distinguishes between development and maintenance when doing economic planning.

The IT director agrees that the high amount of resources spent on maintenance is a problem, and explains that a contributing factor is the amount of integration work. The old systems constantly need to adapt as new tools and applications are introduced, and this is time-consuming work. The organization also experiences problems with keeping their maintenance personnel, and often new employees must obtain competence of legacy systems. This makes the maintenance work less efficient.

When it comes to replacement systems, the main motivation to do these projects is to renew the technology. This is related to the problem with getting maintenance personnel with the required competence, to maintain legacy systems. The organization does not usually have any ambitions to introduce any new functionality in these replacement systems, and the goal is mainly to not lose any functionality. However, when COTS products are chosen over in-house developed systems, the organization has to alter their way of doing things, to adjust to the new software. This is a challenge, but the director feels that it is usually manageable to adapt.

8.1.3 FUTURE PLANS AND IMPROVEMENT PROPOSALS

The IT director expresses concerns about the current system architecture. He says that it's not too bad, but not good enough either. This is a consequence from many years with introducing new systems, and keeping the old ones. Making them work together has caused many less than good modifications, and they now hope to reverse this. The department has started planning a service-oriented architecture, but they are still not sure how they will implement it. The IT director hopes this will ease the reuse of old components, without modifying them every time.

Regarding the problems related to losing personnel, the IT director says they should probably consider more outsourcing. So far they have very little outsourcing, only a few complete systems are outsourced, in areas where the organization does not want

to be involved. This is especially in systems related to “HMS” (Health and Safety). The organization has no direct plan to outsource more of their IT activity, but the director admits that they should probably look into the possibilities of this. He believes there might be benefits to this, if there are special areas where there is a technical environment available.

8.2 INTERVIEW B

The participating organization in this interview is a public enterprise, fully owned by the Norwegian state. It has 45 employees, and an IT department of corresponding to 3 man-labor years.

35% of the organizations resources are spent on maintenance, 20% is spent on development. Isolated this gives 64% of traditional maintenance. Application portfolio upkeep sums to 55%. The organization outsources 90% of its total IT activity.

8.2.1 IT ACTIVITY, IT GOVERNANCE AND IT STRATEGY

This organization has a department called "IT & Archives", which handles the IT activity. This department consists of an IT coordinator, and two more employees. The IT coordinator is hired as a consultant, and not employed by the organization. The organization owns the IT infrastructure, and has servers running on-site. They have a contract with a supplier of IT services to do all of the organizations operational work. In addition, they have a general agreement with five more contractors, regarding development and maintenance work.

A plan of action is worked out every year, and states what projects and activities shall be prioritized. This plan is worked out according to the organizations IT budget. If urgent needs occur, there might be necessary to change the plan of action during the year. If the projects are big, they decisions regarding the projects are handled by the executive group.

8.2.2 SUCCESS CRITERIA AND CHALLENGES

The organization defines result measures, and effect measures together with every project. However, there is not necessarily any "return on investment" analysis involved. The goals are often on a more qualitative level, related either to enhance non-functional aspects of existing systems, or to offer new functionality to the users of the system, enabling new methods of work.

Since the organization outsources, or hires consultants, for all their development and maintenance work, they find it fairly easy to distinguish between development and maintenance in their budgets. However, when a contractor does work on a existing system, they do not distinguish between upkeep and evolution of the applications. The IT coordinator admits they could have followed some of the projects closer, but also feels like it is more effective to let contracts cover both types of work.

For any given project to be certified as a success, the IT coordinator states that delivery on time and budget is the most important aspect. Some of the projects are regarded critical on time, while in other cases the quality of the application is more important. In the cases where these measures are not fully satisfied, the IT coordinator does entirely blame the contractors. Often the employees of the organization do not have the time, or does not prioritize, the projects carried out by the contractors. When the IT employees are not fairly involved in the projects, it is difficult to challenge the contractor and demand more from them. This might lead to projects running late.

The organization has a wish to reduce the share of non productive maintenance work, and mentions their replacement system projects as a measure taken to do so. The IT coordinator states that things happens fast when it comes to IT systems, and the people doing the maintenance see advantages in changing the technology of older systems. New functionality is not usually something they plan for when they do replacement system projects. However, since the systems being replaced are changing continually, the goal is to have all the functionality present in the old system in the new one when it is introduced.

The organization does not prioritize re-use of requirements, design or code. The IT coordinator hopes that this might come more in to focus in time to come, as they are planning on implementing a service-oriented architecture.

8.2.3 FUTURE PLANS AND IMPROVEMENT PROPOSALS

In today's situation the IT coordinator has a couple of proposals to what would improve the development and maintenance work. Since all projects are carried out by contractors and consultants, there is a need for better reporting tools. This would ease the organizations involvements in the projects, and change requests and bug reporting would always come through to the developers.

One of the reasons mentioned why there are lot of resources is spent on maintenance, is lack of competence with the contractors. When the original developers of a system quit, there might not be any others that knows the technology used. The extra time spent to learn new technology, is a cost the organization partially have to carry. The IT coordinator says it might be possible to protect the organization against this in future contracts.

He also believes the organizations "ad-hoc" approach to maintenance work might be ineffective. Especially since none of this work is done in-house, he feels a stronger maintenance regime might be better for the organization. He proposes a system where non critical bugs and change requests are collected in the organization, and the changes to the systems are released in newer version or patches. He says he has no empirical proof that this will improve the efficiency, but following the feedback he gets from employees and consultants this might be an improvement.

In regards of future work, the organization has had a study done about implementing a service-oriented architecture. This study concluded that the organization should move against a service-oriented architecture, and they are now in the middle of a process to decide how this best can be done.

They have not decided whether they will keep legacy systems under the service layer, or whether all systems shall be replaced in order to fulfill the new architecture. The IT coordinator says that this is always part of the consideration when new projects are proposed, and that replacement systems under development right now are prepared to be a part of a service-oriented architecture.

Also, the IT coordinator sees advantages with using COTS instead of developing their own systems. They are constantly on the lookout for software that applies to their organization structure. The main motivation for using COTS, is the costs related to maintenance of their in-house developed systems.

8.3 INTERVIEW C

The participating organization in this interview is a public institution. It has 350 employees, and an IT department has total number of 23 employees.

20% of the organizations resources are spent on maintenance, 50% is spent on development. Isolated this gives 29% of traditional maintenance. Application portfolio upkeep sums to 79%. It shall be noted, that the organization is in the middle of a very big project, to replace nearly all their systems. The organization outsources 30% of its total IT activity.

8.3.1 IT ACTIVITY, IT GOVERNANCE AND IT STRATEGY

The organization is in the middle of a restructuring of their IT department. Up until now, they have had a traditional IT department with around 40 employees. 20 of these were developers, and the rest were leaders, personnel with responsibility for functionality and infrastructure personnel. The IT department did all the development and maintenance internally, and the systems ran on a mainframe operated by an external contractor.

The organization is in the middle of a modernization process that includes both the information systems and their architecture, and also the organization of the IT department itself. The aim is to not have any in-house technology specialists, except for customer competence. In the new IT department knowledge about the organizations core business and functional requirements will be central. While technology matters will be outsourced, or solved by contractors.

When it comes to propose and prioritize new projects, often this comes from political decisions. However, a small part of the budget is set aside to enhancements the organization itself can decide. In these situations, the management makes the decisions. It is a trend the organization has seen for a while, and also wishes to continue, that the IT department becomes a less powerful institution. The interviewee also mentions a plan to implement expert users at different levels of the organization, who will assist in decisions regarding the IT strategy.

8.3.2 SUCCESS CRITERIA AND CHALLENGES

The interviewee mentions alignment across the organization as the most important part of a IT project. The different leaders involved needs to believe in the project, and they all need to pull in the same direction. And obviously, the organization has to supply enough competence and resources to the project.

After everything is in the right place regarding the organization itself, the success of the project also depends on the contractors. One aspect is how skilled the consultants are, another aspect is the contract itself. Often problems with the contractors can be traced back to a sloppy follow-up of the competition for the contract, and the agreement of the contract. For the project to be a success, it is important that the organization has full control of the contract. An aspect of the contract that has proven to be as important as the requirements is the determination of what methods shall be used. Good methods are a prerequisite, and so is high awareness of the projects' completion model.

Over the years the organization has had many challenges regarding rapid changes in the IT systems, related to new laws and regulations from the government. It has not been easy to rapidly implement changes in the legacy systems of the organization, and this is part of the motivation for the new system architecture. As new systems are implemented, features as a robust and flexible system are highly prioritized.

The organization distinguishes clearly between what they call maintenance, which is error correcting, and the maintenance that really is further development of the systems without being a project. This line is clear all the way from the budget, and to the execution of the tasks.

Regarding maintenance, the interviewee does not think they spend too many resources on it. When it comes to whether or not the investment in maintenance gives any real value, he believes preventative maintenance to save money at a later point in time, or in another budget. He feels as if they maybe should spend even more money on maintenance and application portfolio upkeep.

Guidelines for re-use of specifications, design and code were established in 2001. However, 70% of their systems were made before that. For that reason, they struggle with badly documented code, and feels that this is almost impossible to re-use.

8.3.3 Future Plans and Improvement Proposals

As the organization is in the middle of a modernization process, they have a good overview of how their future will be. They have just started the implementations of a service-oriented architecture, and the first delivery of this project will be in couple of months.

The motivation for the ongoing replacement system project, which in time will replace all the information systems in use, is mainly to get rid of old code. Since the systems are not well documented, and the availability of developers with the relevant competence is low, they are not able to develop the systems as efficient as they have to. In the first phase of the replacement system project, they have tried to implement new functionality together with the functionality from the old systems. This has not been easy, and the interviewee feels they have learned from this that new functionality might be better to add later. However, in some cases, the work processes are meant to change with the new system. In these cases, the functionality cannot be directly copied. To avoid problems related to implementing this new functionality, they will do pilots before the implementation.

When designing the new system, they think functionality first. This is easily transferred to a service-oriented reasoning, and the IT department designs the services. This is later given to the contractor as part of the requirement specification, and they do the technical decisions at large. This is something the interviewee feels has worked very well. It makes it easy to track the functionality in the technical solutions, and hopefully this will ease further development later. The tidiness makes the service-oriented architecture is to work with, and this increases the involved personnel's understanding of the system. However, it has been more expensive than first foreseen.

The organization has not spent a lot of resources on maintenance earlier. The interviewee thinks the reason is that even though their old applications are big, they are somewhat controllable. Everything is "hardcoded", and there are very few interfaces, and not too much communication.

He fears that they will spend more resources on maintaining the new architecture, regardless of others opinions that service-oriented systems shall be easier to maintain. His experience tells him that when more interfaces are involved, many different technologies are involved, and many persons with different competence work on the project, the system becomes more complex.

With these new service-oriented systems, it is easy to generate new interfaces if you need information from somewhere in the system. This makes "everything possible" in terms of dataflow, and code is generated at a high rate, to please increasing expectations from users of the system, internally and externally. This may in turn lead to more complex systems, which actually might become more difficult to maintain.

In the days of the old systems, each developer had its field of responsibility. If this field needed maintenance, the dedicated developer worked on it. In the new system, everything will be more complex. One of the main reasons the interviewee believes the share of maintenance will go up, is that more complex systems require more communication. He believes that when maintaining a complex system, as much as 60% of the time might be spent on communication between the involved personnel.

9 DISCUSSION

We will now look at our results on a higher level, and place them in the context of similar investigations performed earlier. In this chapter there will be several references to the investigations this study is meant to follow up. Findings from [29], [30] and [32] will be mentioned as the results from 1993, 1998 and 2003.

9.1 SHARE OF MAINTENANCE

First we will discuss the results we received from the survey investigation, and also the outcome of our hypotheses testing, on a higher level than before. First we will look at traditional maintenance, and the application portfolio upkeep.

9.1.1 TRADITIONAL MAINTENANCE

The share of maintenance, when we only look at maintenance and development, is 62,2%. When we compare this to previous investigations in Table 9-1, we see that the number is a little lower than the investigations from 2003, 2000 and 1998. This might indicate a negative trend in share of maintenance, especially if we look the Norwegian investigations. Still we were not able to reject the hypothesis that the maintenance level is the same as in 2003. We must therefore be careful drawing conclusions, as these numbers are not significantly different, statistically speaking.

Year	Investigation	Maintenance
1977	Lientz and Swanson [25]	51%
1990	Nosek and Palvia [26]	58%
1990	Dekleva [27]	66%
1990	Arfa [28]	49%
1993	Krogstie [29]	59%
1998	Holgeid [30]	73%
1999	Fitzgerald [31]	44%
2000	Capers Jones [2]	73%
2003	Jahr [32]	66%
2008	***This investigation***	***62%***
2010	Capers Jones [2]	79% (estimated)

Table 9-1: Share of maintenance, previous investigations

When we look at the results of other investigations in Table 9-1, they also show a fairly continuous trend of increasing maintenance work up until Capers Jones investigation from 2000, where the share of maintenance was reported to be 73% (including user support)[2]. In 1990 we have looked at three different investigations, with results from 49% to 66%. If we take the average of these investigations, we get 58% maintenance

work. This fits well with the theory that the share of maintenance work is increasing. In 1999 Fitzgerald reported that share of maintenance was as low as 44%, which is a drastic deviation from the trend [31]. However, we should keep in mind that in this investigation, 60% of the participants represented consultant agencies and software houses. We can imagine why these types of businesses do more development than maintenance, and this might have affected the final results.

If we look away from Fitzgerald's investigation, it still seems as the trend might have been reverted. The share of maintenance went from 58% in 1990, to 73% in 1998 and 2000, and then dropped to 66% in 2003, and now 62% in 2008. The increase in maintenance work in the nineties has before been connected to Y2K related issues [30] [2], which lead to more maintenance work. This also fits with the dropping share of maintenance after 2000.

Capers Jones claims that issues as Y2K and the introduction of the Euro will keep appearing, and because of this the share of maintenance will keep increasing until it reaches a steady state at 77%-78% maintenance [2]. This cannot be confirmed by our investigation series, showing a clear decrease in share of maintenance work from 1998 to 2008. However, we must not forget Capers Jones includes user support in his definition of maintenance. If we include support we get 72% maintenance, and only 28% development. This is about the same as Capers Jones found in 2000, but not in line with his 79% estimate for 2010.

9.1.2 APPLICATION PORTFOLIO UPKEEP

When it comes to application portfolio upkeep, we do not have that many investigations to compare our results to. However, we will look at our results in light of the results from 1993, 1998 and 2003.

Year	Investigation	App. Portfolio upkeep
1993	Krogstie [29]	44,0%
1998	Holgeid [30]	62,3%
2003	Jahr [32]	61,1%
2008	***This investigation***	***62,9%***

Table 9-2: Application portfolio upkeep, previous investigations

It has been argued that application portfolio upkeep gives a better impression of whether or not the organization has a successful portfolio management than traditional maintenance [29]. The traditional terms of maintenance and development only considers whether the work done is performed on an existing system, or if a new system is being built. The terms of application portfolio upkeep and application portfolio evolution however are concerned with functionality, and are thus more related to the business processes of the organization. If we want to know how much work is done, without directly fulfilling functionality needs of the organization, it is more helpful to look at application portfolio upkeep.

Whereas the share of traditional maintenance seems to vary over the years, the share of application portfolio upkeep seems to stay very much constant. If we look away from the investigation in 1993, the other results vary with less than 2%. This implies

that decisions regarding whether one should make replacement systems or maintain old systems may vary with time. The same goes for decisions regarding whether one should enhance old systems or develop new ones. What seems to stay constant is the proportion between the need for IT systems to evolve functionally, and the need to keep the existing functionality intact. While the proportion of traditional maintenance and development may vary with the waves of new technology and time related issues, the upkeep/evolution ratio has stayed the same for 10 years. This need to enhance functionality in systems seems to confirm two of Lehman's Laws, which we reviewed in our background study. Law number one, about continuing change in the functionality, and law number six, about continuing growth in the functional content, are both confirmed by the fact that the functional evolution of software systems stays constant [14].

The only hypotheses regarding application portfolio upkeep that was rejected, was H5, H9 and H10. The rejection of H5 means significantly more time was spent on application portfolio upkeep than on application portfolio evolution. The rejection of H9 and H10 means that more developers per user and IT employee equals less time spent on application portfolio upkeep. These results may imply that IT departments that hold its own technical environment for developers are able to spend more of their time supporting the organizations' core businesses. Although we were not able to reject any of the hypotheses regarding outsourcing or use of consultants, the rejections of H9 and H10 implies that IT departments that have more than just customer competence are more efficient when it comes to producing functionality in the IT systems.

9.2 VARIABLES AFFECTING MAINTENANCE

We will now discuss different variables that may affect the share of maintenance, and look at these variables in the light of this investigations, as well as investigations performed earlier.

9.2.1 INCREASING COMPLEXITY OF THE APPLICATION PORTFOLIO

In this investigation we could not reject any of the hypotheses, dealing with complexity and maintenance. We did correlation tests with the maintenance variables and total number of main systems, number of users, average age of systems and number of different programming languages in use. None of these signs of complexity showed a correlation with any of the maintenance variables.

However, two of the three interviewees express concern about complexity of the application portfolio. Interviewee A says the current complexity, in form of many systems and different programming languages, is the reason for maintenance being less than efficient. Interviewee C expresses concern that there new service-oriented architecture may actually lead to more difficult maintenance work. Although no legacy systems will be a part of the new service-oriented architecture, he fears that many different interfaces and the several different technologies involved might lead to a higher share of maintenance work. Interviewee B did not speak on this issue, one way or the other.

Although we are not able to prove any relation between share of maintenance and complexity of systems, our case study tells us that there might be a connection here after all. At least the IT leaders seem to feel this way.

In the investigation from 2003, it was found that increasing complexity in the portfolio leads to increased share of maintenance. Worth mentioning is also that in 1998 complexity were actually found to correlate negatively with traditional maintenance, that is higher complexity meant a lower share of maintenance work.

From the results from 2003, and our results from the case studies, we can confirm Lehman's Law #2 from our background study. This law claims that the complexity of a system increases with time, unless work is done to maintain or reduce it [14]. This seems probable, even though we were not able to prove it statistically in our investigation.

9.2.2 LIFETIME OF SYSTEMS

We were not able to reject the hypotheses that organizations with older systems on an average had higher share of maintenance. However, it is interesting to take a look at the average age of systems over the years, and see how this relates to the share of maintenance.

In 1993 and 1998 the average age of systems was 5 years, the same as in our investigation. In 2003 the average age was 3,9 years. We cannot say that these numbers in any way coincide with the share of maintenance.

Even though the average numbers did not give us any proof of whether or not the lifetime of systems has anything to do with maintenance share, all three interviewees had something to say about this matter. They all agreed that old systems were difficult to maintain, due to difficulties in getting personnel with the right competence. Furthermore Interviewee C uttered that limitations in the legacy systems itself, made it difficult or impossible to react fast enough to process changes in the organization.

Even though no statistical evidence can be shown to prove that age of systems affects maintenance in our survey, we think this is because the organizations constantly make moves to keep their application portfolio from getting too old.

9.2.3 TURNOVER IN PERSONNEL

We were not able to reject the hypothesis that organizations with higher turnover amongst developers have the same amount of maintenance work as organizations with lower turnover. Nor can we see any patterns when it comes to the developers experience and maintenance in the previous investigations.

In our investigation the average developer has 5,8 years of experience from the IT-department. The previous investigations reported an average of 7 years in 1993, 8,8 years in 1998 and 5,4 years in 2003. If anything, all these numbers tell us is that developers change jobs more often today, than 10 and 15 years ago.

Still, "Turnover in maintenance personnel" is listed as one of the top five problem areas, in all the investigations in this series. Also all three interviewees in our case study mentions losing competence when developers quit as very problematic.

It seems like this is a genuine problem in our field of study, as it has been mentioned for so long. Perhaps is the high awareness of the problem contributing to minimize the consequences of it, for example by documentation of old systems, or replacing systems when the needed competence gets difficult to procure.

9.2.4 *Higher User Expectations*

Questions regarding this are not part of our survey, as this is something we did not think of before one of our interviewees mentioned it. However, the increasing amount of user support, and the increase in the amount of external users, can very well be reasons for higher user expectations.

In a way, higher user expectations might just be seen as a reason for systems becoming more complex, which in turn might be a reason for higher amounts of maintenance work. Still, we find the relevant enough to dwell upon before we move on.

Interviewee C's assertion is that the users, both internal and external ones, demand more and more from the information systems. We also see from the results regarding organizational controls, that users are more heavily involved in the development and maintenance process of software. Because of this, a trend towards a service-oriented architecture seems to be a fact.

With this architecture in place, every system can use information or functionality from all other systems or data stores in the organization. If the user wants it, and it is available, the interfaces will be generated. Interviewee C fears that these will lead to an over saturated service bus, which will be very messy to maintain. Of course actions can be taken to avoid this; the interesting part is whether or not these actions will be taken or not.

Already in 1998, Holgeid proposed that increasing user expectations and complexity of technology might become a problem for maintenance. He concludes that with loads of change requests coming from the users, good routines must be in place to prioritize and control the development.

9.2.5 *Use of consultants and outsourcing in development*

In our investigation, we could not find any direct correlation between maintenance, and the use of consultants and outsourcing. However, it seems that organizations with few developers in proportion to internal users and IT employees, spends more of their resources on maintenance. We rejected the hypotheses H9 and H10 because of this correlation.

We must be careful not take any shortcuts here, and we cannot claim that organizations with few system developers automatically outsource more, or use more consultants. However, it is a trend to outsource the IT expertise, and maintain better

customer competence around IT and core business [23] [22]. Further, two of the three organizations from our case studies are heading towards a strategy were they have a minimum of in-house technical expertise. The third organization is not using outsourcing in a big scale, but the interviewee admits that they probably should look into the possibilities of outsourcing more.

It is interesting however, that with this trend going on; it is still the organizations with many developers per user that minimizes both traditional maintenance and application portfolio upkeep. It will be interesting to see how this trend continues, and whether or not it can be determined in the future if this strategy pays off.

Another important aspect when discussing outsourcing was pointed out to us by Interviewee B; it always comes down to the contract. As an example he tells us about a contract where his organization were held economically responsible when the contractor could not hold on to their competence. This was obviously a circumstance that raised the price of the maintenance project, as the contractor was paid to teach new personnel the technology in question. The interviewee pointed out that these are the types of costs you want to protect yourself against by outsourcing.

Also in the 1993 investigation the number of system developers is found to correlate negatively with the share of maintenance variables.

9.2.6 MAINTENANCE REGIMES

In the investigation from 1998 it is reported that organizations were IT is of high strategic importance, and where the IT department is big in proportion to the rest of the organization, application portfolio upkeep is low. We do not have the same correlation in our investigation. However, we have results from our case studies supporting the theory that well defined maintenance regimes are of absolute importance to a well managed application portfolio.

Use of methods was mentioned both by Interviewee A and Interviewee C as important success criteria. Interviewee C went as far as saying the method itself was often more important, than the requirements itself. This was mentioned when talking about contracts with external companies.

Interviewee B also speaks of how they want to implement a stronger maintenance regime. They have not decided what measure should be taken, but he thinks a routine where change requests are gathered and implemented simultaneously would increase the maintenance efficiency. Also, better tools for following up change requests and bug reports are about to be implemented.

Both Interviewee A and Interviewee B agreed that minimizing application portfolio upkeep would be of interest, and said they could be better in terms of controlling their maintenance. Interviewee C however did not feel the same way. He uttered that both the upkeep and the traditional maintenance was necessary. He also said that they were talking about spending more time on preventive maintenance, as they had experienced that this drastically decreased other types of maintenance. It shall be mentioned that Organization C has a very small share of traditional maintenance, with only 29% when we look at maintenance and development isolated.

All the interviewees mention the development of replacement systems as a step towards easier and hopefully cheaper maintenance. Still, Interviewee C hesitated when being further asked about this. He corrected himself, and said that adding functionality to the new systems would be easier, but he was not so sure the actual maintenance would become more efficient.

9.3 Outsourcing and Service-oriented Architecture

Both outsourcing and service-oriented architecture are new fields of interest in this investigation series. This means we don't have any real grounds for comparison. Still, because of the case studies, we have some interesting observations.

As we have mentioned, two out of the three organizations included in our case studies are already implementing a strategy involving outsourcing on a big scale. Both these interviewees were happy with the strategy, but said it demands more from the organization as customers than first expected. Interviewee B even states that when problems occurred, they mostly had to take the blame themselves, for not dedicating enough of their own time to the projects. When it comes to the survey results, we were not able to reject any of our hypotheses regarding outsourcing. In our investigation outsourcing does not significantly affect the share of maintenance, and the use of outsourcing is not determined by the size of the organization.

Of the three organizations in our case study, all of them wanted to implement a service-oriented architecture. Organization C has already started the implementation, while Organization A and Organization B had yet to decide how they wanted to implement this new architecture. Both Interviewee A and Interviewee B were of the opinion that a service-oriented architecture would make maintaining the system easier. Interviewee C however said that the maintenance work might actually become more difficult, as the system would become more complex, and have many more interfaces than the old system had.

Implementation of a service-oriented architecture does not significantly affect the share of maintenance, and there is no correlation between size of the organizations and whether or not they deploy, or wish to implement, a service-oriented architecture.

10 EVALUATION

In this chapter we will evaluate our own work, and look at the different limitations of our investigation.

10.1 POPULATION

The participating organizations were chosen randomly from the members of The Norwegian Computer Society. This was both because of a belief that these organizations had a active IT department, and because it would ease the invitation of the organizations. Looking at our respondents, some of them are really too small to be of value for an investigation like this. Looking back, we should probably decide on a minimum of employees for the participating organizations.

Since all our respondents are members of The Norwegian Computer Society, this can also put limitations to our generalization. We have not checked if there are any common factors of the participating organizations.

10.2 RESPONDENTS

The vast majority of the respondents are IT leaders. This can by some be viewed as a limitation, and that we should also interviewee developers, as they may have different views. However, most of the similar investigations have been carried out amongst IT leaders. Therefore, it was natural to ask only for the IT leaders when inviting organizations to participate in our survey.

10.3 RESPONSE RATE

We got a response rate of 25%. This is not optimal when it comes to generalizing the results, but it is still better than many of the similar investigations. The other investigations we have used as backgrounds for our study have all response rates around 20%. Because of this, it should be ok to compare our results to the results from these investigations.

10.4 UNDERSTANDING OF CONCEPTS

It may be a problem that the respondents have their own definition of different concepts in the survey. We have tried to guard ourselves against this by defining even well known concepts as “Outsourcing” and “Service-oriented architecture” in the beginning of the survey. Hopefully this contributes to the respondents talking about the same things. However, there might still be some misinterpretations, and we cannot say for sure that every respondent operates with the same definitions on all concepts.

10.5 Biased Questions

Biased questions can often be a problem in survey investigations. However, we believe that we avoided this as best we could. Before we sent out the survey, we did a pilot that was reviewed both by IT personnel and researchers with experience in the field of survey investigations.

One exception may be the question regarding strategic importance of IT. In all the surveys of this investigations series, the answer has been unnaturally high. This is probably because IT managers have problems to admit otherwise. However, we promised and effectuated full confidentiality to all our respondents.

10.6 Hypothesis Testing

Most of our hypotheses are tested by correlation. These tests do not say anything about causes, or why the two variables correlate. The reasons that variables correlate may be external to our data, and we cannot conclude that one situation causes the other.

Our discussion chapter is therefore only an attempt in defining what may be the reason for different values of maintenance and development, and it is important to emphasize that we do not present any reliable or proven results.

10.7 Research Method

Our combination of quantitative and qualitative method is an attempt to neutralize the drawbacks of these methods. The survey allowed us to reach many organizations efficiently, and the case studies allowed us to follow up on a smaller number of the respondents. In the survey we asked about the accuracy of the most important data. This allows us to easily evaluate the accuracy of the answers to different questions. In most cases the answers were distributed evenly amongst the three grades of accuracy; low, some or high accuracy.

To optimize the method, we should probably have done some more interviews, as only three may put some limitations on our discoveries. Still we managed to find some interesting agreements and disagreements from the three interviewees.

11 CONCLUSION AND FURTHER WORK

With this final chapter, we will conclude our investigation, and suggest further work related to this field of study.

11.1 CONCLUSION

The investigations from 1993, 1998 and 2003 have reported that maintenance makes up 59%, 73% and 66% of the work, respectively. In our investigation this sums to 62%, when we look at maintenance and development alone. With these numbers we can conclude that our series of investigations does not seem to follow the increasing trend of maintenance work, proposed by Capers Jones [2]. Even when we include support, we only get 72% of maintenance when we look at maintenance and development alone. Rather, it seems like the maintenance decreases after Y2K, as Holgeid suggested should happen [30].

When we look at application portfolio upkeep, the investigations from 1993, 1998 and 2003 have reported that this makes up 44%, 62% and 61%. In our investigation the average application portfolio upkeep amounts to 63%. It seems that this number is very stable, regardless of the maintenance share.

From our hypothesis-testing we cannot conclude that either type of organization, number of consultants, complexity of the portfolio, degree of outsourcing or deployment of a service-oriented architecture affects the share of maintenance. This is different from previous investigations, where especially complexity of portfolio and type of organization has proven to affect the organizations share of maintenance. In 2003 the number of main systems correlated with traditional maintenance, and use of methodology where found to correlate negatively with application portfolio [32]. Neither of these variables where found to correlate with maintenance or upkeep in our investigation.

However, we were able to correlate the number of system developers in proportion to users and total number of IT employees to the maintenance variable. It seems that many system developers per user correlates with both share of maintenance, and application portfolio upkeep.

11.2 FURTHER WORK

We have during this study revealed new fields of interest, which may be further investigated. Some of this was revealed in the case studies, and some are questions we were not able to answer.

First of all, further work can be done with regards to service-oriented architecture and outsourcing, and how it affects maintenance and development. This was the first time these aspects were included in the survey, and we will need more data to see how this trend of IT departments doing less programming in-house, affects efficiency.

Also, the questions related to user expectations and system complexity deserves an answer. There might be an option to include questions regarding these aspects in the next investigation, alternatively include system developers when discussing complexity. It seems natural that developers have a better understanding of whether an application portfolio is complex or not. This is still difficult to evaluate for survey data, and a more comprehensive form of case study may have to be performed in order to get good answers.

The last thing we would like to propose is an investigation in how awareness of different issues related to maintenance and development affects the ratio itself. Our case studies revealed that some organization has stronger regimes when it comes to development and maintenance, and how they handle the challenges that might occur. Information like this is not trivial to attain through surveys, and case studies might have to support the study if this is to be investigated further.

REFERENCES

1. Krogstie, J., A. Jahr, and D. Sjøberg, *A longitudinal study of development and maintenance in Norway: Report from the 2003 investigation.* Information and Software Technology, 2006. **vol 48**: p. 993-1005.
2. Jones, C., *The Economics of Software Maintenance in the twenty first century*. 2006, Software Productivity Research, Inc: Hendersonville, North Carolina.
3. Brooks, F.P., *No silver bullet: Essence and accidents of software engineering.* IEEE Computer, 1987. **20**(4): p. 10-19.
4. Swanson, E.B. *The Dimensions of Maintenance*. in *Proceedings of the 2nd International Conference on Software Engineering*. 1976. San Francisco.
5. ISO, *ISO/IEC 14764: Software Engineering, Software Life Cycle Processes, Maintenance*. 2006.
6. Chapin, N., *Software Maintenance Types - A Fresh View.* Proceedings of the 2000 International Conference on Software Maintenance (ICSM-00), 2000: p. 247-252.
7. Grubb, P. and A.A. Takang, *Software Maintenance: Concepts and Practice*. 1996, World Scientific Publishing Company.
8. Royce, W.W., *Managing the development of large software systems.* Proceedings, IEEE WESCON August, 1970.
9. Larman, C. and V.R. Basili, *Iterative and Incremental Development: A Brief History.* IEEE Computer, 2003. **36**(6): p. 47-56.
10. Boehm, B.W., *A Spiral Model of Software Development and Enhancement.* IEEE Computer, 1988: p. 61-72.
11. Cunningham, W. *Manifesto for Agile Software Development*. [cited 2009 May 6th]; Available from: http://www.agilemanifesto.org/.
12. Stafford, J.A., *Software Maintenance As Part of the Software Life Cycle*. 2003, Tufts University Information Technology: Somerville, MA.
13. Bennett, K. and V. Rajlich, *Software maintenane and evolution: a roadmap.* Proceedings of the Conference on The Future of Software Engineering, 2000: p. 73-87.
14. Lehman, M.M., *Laws of Software Evolution Revisited.* Lecture Notes In Computer Science, 1996. **Vol. 1149**: p. 108-124.
15. Godfrey, M.W. and Q. Tu, *Evolution in Open Source Software: A Case Study.* Proceedings of the 2000 International Conference on Software Maintenance (ICSM-00), 2000.
16. Dong, Y. and S. Mohsen, *Does Firefox obey Lehman's Laws of Software Evolution?* 2007, Univeristy of Waterloo, Department of Management Sciences: Waterloo, Canada.
17. Fujitsu, *The Key to SOA Governance*, in *Technical White Paper*. 2008: Tokyo.

18. Lewis, C. and D. Smith, *Service-Oriented Architecture and its Implications for Software Maintenance and Evolution.* Frontiers of Software Maintenance, 2008: p. 1-10.
19. Banker, R.D., G.B. Davis, and S.A. Slaughter, *Software Development Practices, Software Complexity, and Software Maintenance Performance: A Field Study.* Mangement Science, 1998. **Vol. 44**(No. 4): p. 433-450.
20. Oreld, M., *IT-folket dumper SOA*, in http://www.idg.no/computerworld/article111189.ece. 2008, IDG.no: Oslo.
21. NucleusResearch, *Benchmarking: Service Oriented Architecture*. 2007, Nucleus Research Inc.: Boston.
22. Bardhan, A.D. and C. Kroll, *The New Wave of Outsourcing*, in *Fisher Center Research Reports*. 2003, University of California, Berkeley: Berkeley.
23. Ahmed, R.E., *Software maintenance outsourcing: Issues and strategies.* Computers and Electrical Engineering, 2006(32): p. 449-453.
24. Pilskog, G., *Mest utflytting av IT-tenester frå små land*, in http://www.ssb.no/magasinet/norge_verden/art-2008-02-05-01.html. 2008, SSB: Oslo.
25. Lientz, B. and E.B. Swanson, *Characteristics of Application Software Maintenance.* Communications of the ACM, 1978. **vol 21**(#6): p. 466-471.
26. Nosek, J.T. and P. Palvia, *Software maintenance management: Changes in the last decade.* Journal of Software Maintenance, 1990(2): p. 157-174.
27. Dekleva, S., *Delphi study of Software Maintenance Problems*. 1992, DePaul University: Chicago.
28. Arfa, L.B., A. Mili, and L. Sekhri, *An Empirical Study of Software Maintenance*. 1991, University of Tunis II, Department of Informatics: Belvedere, Tunisia.
29. Krogstie, J., *On the Distinction between Functional Development and Functional Maintenance.* Journal of Software Maintenance, 1995. **vol 7**: p. 383-403.
30. Krogstie, J., K. Holgeid, and D. Sjøberg, *A study of development and maintenance in Norway.* Information and Software Technology 2000. **vol 42**: p. 687-700.
31. Fitzgerald, G., A. Philippides, and S. Probert, *Information systems development, maintenance and enhancement: findings from a UK study.* International Journal of Information Management, 1999(19): p. 319-328.
32. Jahr, A., *Development and maintenance of IT-systems in Norwegian organizations*, in *IFI*. 2005, UIO: Oslo.
33. Rea, L.M. and R.A. Parker, *Designing and Conducting Survey Reserarch*. 1997, San Francisco: Jossey-Bass.

34. Ringdal, K., *Enhet og Mangfold - Samfunnsvitenskapelig forskning og kvantitativ metode*. 2nd. ed. 2001, Trondheim: Fagbokforlaget.
35. Haug, P., *Den skjulte IKT-bransjen i Norge*, in *IDI*. 2008, NTNU: Trondheim.
36. SurveyMonkey. *http://www.surveymonkey.com*. Survey Monkey Survel Tool [cited 2009 June 2nd].
37. Yin, R.K., *Applications of Case Study Research*. Second Edition ed. Applied Social Research Methods Series, ed. D.J.R. Leonard Bickman. Vol. 34. 2003, Thousand Oaks (California): Sage Publications. 174.
38. Arthur, L.J., *Software Evolution: A Software Maintenance Challenge*. 1988: John Wiley and Sons.
39. Jørgensen, M., *Empirical Studies of Software Maintenance*. 1994, PhD Thesis, University of Oslo: Oslo.
40. SSB. *Konsumprisindeksen fra 1865*. [cited 2009 May 28th]; Available from: http://www.ssb.no/kpi/tab-01.html.

APPENDIX I: THE QUESTIONNAIRE

Utvikling og vedlikehold av IT systemer i norske bedrifter

1. Veiledning for utfylling

* 1. Virksomhetens navn:

* 2. Ditt navn:

* 3. E-post:

Info og veiledning

Spørreskjemaet vil enklest kunne besvares av en IT-sjef eller en som innehar tilsvarende stilling i virksomheten. Svarene skal være basert på de rutiner og den praksis som virkomheten har i dag.

Relevansen til noen av spørsmålene vil være avhengig av svar på tidligere spørsmål. Hvis enkelte spørsmål ikke er relevante, fyll ut med antall 0 eller en blank.

De som besvarer undersøkelsen vil motta 500 kroner skattefritt. For å følge opp dette vil vi i etterkant av undersøkelsen benytte e-post adressen du har lagt inn over.

Utvikling og vedlikehold av IT systemer i norske bedrifter

2. Informasjon om deg

* 4. Din stilling:

- Leder
- Prosjektleder
- Systemutvikler

* 5. Ansettelsesforhold:

- Fast
- Midlertidig
- Innleid konsulent
- Annet (Spesifiser)

* 6. Formell utdannelse:

* 7. Antall års IT-erfaring:

8. Kort beskrivelse av type erfaring, arbeidsoppgaver, ansvar m.m. i nåværende jobb:

Utvikling og vedlikehold av IT systemer i norske bedrifter

3. Informasjon om virksomheten

* 9. Type virksomhet:

- Telekommunikasjon og data
- Bank og forsikring
- Offentlig forvaltning
- Helsevesen
- Reise og transport
- Varehandel
- Industri
- Tjenesteyting/konsulentvirksomhet
- Bygg og anlegg
- Annet (Spesifiser)

* 10. Er IT av strategisk betydning for bedriften?

	Ikke av strategisk betydning				Absolutt av strategisk betydning
Hvor stor betydning?					

Kommentarer

* 11. Hvor mange ansatte har bedriften?

* 12. Hva er det årlige budsjettet for dataavdelingen inklusive maskinvare, programvare, personell og outsourced oppgaver? (oppgitt i millioner kroner, og uten avskrivninger)

- mer en 50
- mellom 40 og 50
- mellom 30 og 40
- mellom 20 og 30
- mellom 10 og 20
- mellom 1 og 10
- mindre enn 1

Utvikling og vedlikehold av IT systemer i norske bedrifter

4.

* 13. Hvor stor andel av følgende aktiviteter gjøres av andre virksomheter, ved utsetting av aktiviteten? (Outsourcing, ikke innleide konsulenter. Anslå i forhold til tidsbruken på aktiviteten)

Utvikling av nye applikasjoner (%)	
Vedlikehold/forvaltning av eksisterende applikasjoner (%)	
Drift (%)	
Brukerstøtte (%)	
Ledelse av IT-virksomheten (%)	
Den totale IT-aktiviteten(%)	

* 14. Svaret ovenfor er:

- Rimelig nøyaktig, basert på gode data
- Et grovt estimat, basert på minimale date
- En best mulig gjetning, ikke basert på noen data

* 15. På bakgrunn av de totale utførte timeverk internt i dataavdelingen i løpet av et år, hvor mye (i prosent) brukes til:

Rette feil i systemer som er i produksjon	
Tilpasse systemer til endret teknisk arkitektur	
Utvikle ny funksjonalitet i eksisterende system	
Forbedre ikke-funksjonelle egenskaper (f.eks. ytelse og sikkerhet) i eksisterende systemer	
Utvikle nye system som overlapper/erstatter gamle systemer funksjonelt sett	
Utvikle nye system for å dekke nye funksjonsområder	
Drift	
Ledelse	
Brukerstøtte	
Annet	

16. Spesifiser "Annet" i forrige spørsmål:

* 17. Svaret ovenfor er:

- Rimelig nøyaktig, basert på gode data
- Et grovt estimat, basert på minimale date
- En best mulig gjetning, ikke basert på noen data

Begrepet vedlikehold omfatter oppgaver som de fire første alternativene i spørsmål 15.

Utvikling og vedlikehold av IT systemer i norske bedrifter

5. Spørsmål om IT-avdelingen

* 18. Hvor mange personer er ansatt i dataavdelingen (omregnet til fulltidsansatte)?

* 19. Hvor mange av disse er systemutviklere (omregnet til fulltidsansatte)?

* 20. Hva er fordelingen av systemutviklerne med hensyn til hvor lenge de har arbeidet i avdelingen? (antall personer)

0-1 år	
1-3 år	
3-6 år	
6-10 år	
Mer enn 10 år	

* 21. Hvor mange innleide konsulenter innen systemutvikling og vedlikehold har avdelingen i gjennomsnitt over et år (omregnet til fulltidsansatte)?

Utvikling og vedlikehold av IT systemer i norske bedrifter

6. Virksomhetens applikasjonsportefølje

* 22. Blir vedlikehold av informasjonssystemer utført av de som opprinnelig laget systemet?

	Aldri				Alltid	Vet ikke
Hvor ofte:						

Kommentar:

* 23. Hvor mange større systemer (hovedsystemer) er i produksjon i virksomheten?

* 24. Hvilke områder dekker disse hovedsystemene (f.eks. lønn, lagerstyring, regnskap osv.)?

* 25. Hvor mange sluttbrukere innen virksomheten har disse systemene?

* 26. Svaret ovenfor er:

- Rimelig nøyaktig, basert på gode data
- Et grovt estimat, basert på minimale data
- En best mulig gjetning, ikke basert på noen data

* 27. Hvor mange sluttbrukere utenfor virksomheten har disse systemene?

* 28. Svaret ovenfor er:

- Rimelig nøyaktig, basert på gode data
- Et grovt estimat, basert på minimale data
- En best mulig gjetning, ikke basert på noen data

* 29. Hva er aldersfordelingen til eksisterende hovedsystemer (regnet i år etter første installasjon)?

0-1 år	
1-3 år	
3-6 år	
6-10 år	
Mer enn 10 år	

Utvikling og vedlikehold av IT systemer i norske bedrifter

* 30. Hvordan er de ulike hovedsystemene utviklet? (antall systemer)

Utviklet av dataavdelingen	
Utviklet i brukeravdelingen i bedriften	
Utviklet av et eksternt selskap	
Pakkeløsning, med store interne tilpasninger	
Pakkeløsning, med små interne tilpasninger	
Løsning som bruker Web services utviklet eksternt	

* 31. Hvor mange av hovedsystemene er avhengig av data fra andre systemer?

* 32. Svaret ovenfor er:

- Rimelig nøyaktig, basert på gode data
- Et grovt estimat, basert på minimale data
- En best mulig gjetning, ikke basert på noen data

Utvikling og vedlikehold av IT systemer i norske bedrifter

7. Teknologibruk

* 33. Hvilke programmeringsspråk er i bruk? (Angi antall hovedsystemer pr språk)

COBOL	
Assembler	
C	
C++	
C#	
Java	
Scriptspråk (PHP, Perl osv.)	
4 GL språk	
Andre	

34. Spesifiser "Andre" i forrige spørsmål:

* 35. Hvilke typer databasesystemer er i bruk? (Angi antall databaser pr type)

Hierarkiske databaser	
Nettverksdatabaser	
Relasjonsdatabaser	
Objektorienterte databaser	
Annet	

Utvikling og vedlikehold av IT systemer i norske bedrifter

8. Utvikling av nye systemer

* 36. Har bedriften en plan for innføring av SOA i fremtidige IT systemer?

- Ingen plan om å innføre SOA
- Et ønske om å innføre SOA
- En klar plan på å innføre SOA
- Allerede i gang med å implementere SOA

* 37. Hvor mange nye systemer er for tiden under utvikling?

* 38. I hvor stor grad brukes en tjeneste-orientert arkitektur (SOA) for dagens hovedsystemer?

	Ingen grad				Stor grad	Vet ikke
I hvilken grad?						

* 39. Av totalt antall nye systemer, hvor mange av disse er "erstatningssystemer"? (Systemer som hovedsaklig dekker funksjonalitet som alt er dekket i eksisterende systemer)

* 40. Hva er aldersfordelingen på de systemene som eventuelt erstattes?

0-1 år	
1-3 år	
3-6 år	
6-10 år	
Mer enn 10 år	

* 41. Ved utvikling av erstatningssystem, hva er de viktigste grunnene for at de blir erstattet (gi score fra 1-5 på alle punktene nedenfor)

	1 (ikke viktig/relevant)	2 (lite viktig)	3 (noe viktig)	4 (viktig)	5 (svært viktig)
Svært vanskelig å vedlikeholde eksisterende system					
Svært vanskelig å drifte eksisterende system					
Svært vanskelig å bruke eksisterende system					
Finnes alternativ pakkeløsning					
Finnes alternativ applikasjonsgenerator					
Overgang til service orientert arkitektur (SOA)					
Overgang til ny teknisk arkitektur (ikke SOA)					
Standardisering med resten av organisasjonen					
Integrering med andre nye eller eksisterende systemer					
Annet					

Utvikling og vedlikehold av IT systemer i norske bedrifter

* 42. Ved utvikling av erstatningssystemer, og nye systemer med overlappende funksjonalitet, i hvor stor grad er man i stand til å gjenbruke kode, spesifikasjoner og design?

	Svært lite				Svært mye
Spesifikasjoner og design					
Kode					

Utvikling og vedlikehold av IT systemer i norske bedrifter

9. Metoder og verktøy ved utvikling og vedlikehold

43. I hvilke deler av livssyklusen til IT-systemene anvendes en på forhånd definert metode (sett ett eller flere kryss)?

- Planlegging
- Analyse
- Kravspesifikasjon
- Design
- Implementasjon
- Testing
- Utrulling
- Drift
- Vedlikehold
- Prosjektledelse

Hvilke metoder er i bruk på de ulike feltene:

44. Hvilke deler av livssyklusen støttes i dag gjennom anvendelse av systemutviklingsverktøy?

- Planlegging
- Analyse
- Kravspesifikasjon
- Design
- Implementasjon
- Testing
- Utrulling
- Drift
- Vedlikehold
- Prosjektledelse

Hvilke verktøy er i bruk på de ulike feltene:

* 45. Hvor lenge har man brukt disse systemutviklingsverktøyene i organisasjonen? (antall år)

* 46. Hvor mange av de eksisterende hovedsystemene (oppgitt under spørsmål 23) understøttes av disse systemutviklingsverktøyene?

* 47. Svaret ovenfor er:

- Rimelig nøyaktig, basert på gode data
- Et grovt estimat, basert på minimale data
- En best mulig gjetning, ikke basert på noen data

48. Hvilke rutiner er etablert for vedlikehold av informasjonssystemer?

- Man bruker samme rutiner for endringsforslag som kommer fra dataavdelingen som for endringsforslag som kommer fra brukergrupper
- Økonomiske utstyrskostnader som er forbundet med drift og vedlikehold av informasjonssystemet belastes brukergruppene
- Endringsforslag blir klassifisert etter type og viktighet
- Det gjennomføres en formell gjennomgang av systemet periodisk
- Alle endringer av informasjonssystemet blir testet før systemet settes i produksjon
- Personellkostnader forbundet med drift og vedlikehold av informasjonssystemet belastes brukergruppene
- Ved akseptansetest av endringer, sjekkes også at den tilliggende dokumentasjon er oppdatert
- Alle endringsforslag gjennomgår en konsekvensanalyse og kostnadsestimering
- Alle brukerkrav som kommer inn blir dokumentert
- Brukere som etterspør endringer får beskjed både hvis endringsforslaget gjennomføres eller underkjennes
- Med unntak av driftstruende feil blir alle endringer samlet opp for periodisk implementasjon
- Alle endringer av programvaren blir dokumentert

Kommentar:

Utvikling og vedlikehold av IT systemer i norske bedrifter

10. Problemer innen vedlikehold

* 49. I hvilken grad er følgende områder et problem ved vedlikehold av informasjonssystemer slik du bedømmer det (gi score fra 1-5 på alle punktene nedenfor)?

	1 Intet problem	2	3	4	5 Alvorlig problem
Utskifting av personell					
Kvaliteten av systemdokumentasjonen					
Endringer av maskinvare og systemprogramvare					
Brukerkrav for utvidelser og forbedringer					
Ferdigheter til vedlikeholdspersonell					
Kvaliteten til det originale programmet					
Tilgjengelighet på vedlikeholdspersonell					
Konkurrerende behov om vedlikeholdspersonell					
Manglende interesse fra brukere					
Systemet feiler under operativ drift					
Manglende brukerforståelse av systemet					
Datalagringskrav					
Maskinhastighet					
Motivasjonen til vedlikeholdspersonell					
Vedlikeholdspersonellets produktivitet					
Pålitelighet til maskin og systemprogramvare					
Dataintegritet i applikasjonen					
Urealistiske brukerforventninger					
Programmeringsstandarder ikke brukt					
Trange budsjetter					
Mangelfull opplæring av brukere					
Utskiftninger i brukerorganisasjonen					
Ledelsen støtter ikke bruk av applikasjonen					
Annet					

APPENDIX II: COVER LETTER

Til IT-leder i Organisasjon X eller X X, virksomhetens kontaktperson mot Dataforeningen

I samarbeid med Dataforeningen gjennomfører vi en undersøkelse blant norske virksomheter rundt utvikling og vedlikehold av virksomhetens egne IT-systemer.

Undersøkelsen følger opp tilsvarende undersøkelser foretatt i Dataforeningens regi i 1993,1998 og 2003.

Da spørreskjemaet er relativt omfattende (anslått tid for utfylling 30-45 minutter) vil deltakelse i spørreundersøkelsen kompenseres med 500 kr (skattefritt gitt at man ikke har annen inntekt fra NTNU).

Spørreskjema kan besvares på nettside: https://www.surveymonkey.com/s.aspx
Svarfrist for undersøkelsen er 23/11-2008.

En papirutgave av spørreskjemaet kan lastes ned fra http://folk.ntnu.no/magnekd/dataforeningen_survey.pdf, og kan brukes for å skaffe seg oversikt over det det spørres etter før man fyller inn skjemaet på nett

Magne Davidsen, hovedfagsstudent ved Institutt for datateknikk og informasjonsvitenskap ved NTNU i Trondheim følger opp den praktiske gjennomføringen av undersøkelsen. Spørsmål angående selve spørreskjemaet, eller generelt om undersøkelsen kan rettes til Magne på e-post: magnekd@stud.ntnu.no

Dataforeningen ønsker å støtte akademiske arbeider som tar for seg aktuelle praktiske problemstillinger og oppfordrer alle utvalgte bedrifter til å besvare spørreskjemaet.

Med vennlig hilsen
John Krogstie
krogstie@idi.ntnu.no
På vegne av faggruppe for Applikasjonsintegrasjon - Metoder og arkitektur
Den Norske Dataforening

APPENDIX III: INTERVIEWS

Intervju A

Jeg tenkte å starte med en gjennomgang av spørreskjemaet dere leverte, og spørre om et par ting. Jeg ser blant annet at dere har relativt lite outsourcing, 5% av den totale IT-aktiviteten?

Ja, det er drift av enkeltsystemer hvor vi ikke har kompetansen, og ikke har noe ønske om å drifte selv. Typisk HMS og personopplysninger, og det ønsker vi ikke å håndtere. Så det er outsourcet til noen som er proffesjonelle på det. Er også et par andre mindre systemer.

Så da outsources alt som har med systemet å gjøre? Utvikling, vedlikehold og drift?

Ja.

Ellers ser jeg at dere har 95 årsverk totalt, hvordan er disse organisert? Inkluderer det IT-ansatte i forskjellige underavdelinger?

Nei, det vi (IT-avdelingen) har er hovedsaklig ansvar for alle *KUNDENE* og drift av deres systemer. Også har vi også de sentrale administrative systemer. Alle fellessystemer har vi i stor grad. Vi og andre (avdelinger) har vært sitt e-postsystem, så det er veldig distribuert. Ellers har vi drift av trådløst og fast nett. Men totalt er vi rundt 170 IT-ansatte, inkludert de som sitter rundt omkring i avdelingene.

Også har dere 6 utviklere i IT-avdelingen?

Nja, 6 årsverk utviklere. Totalt 11-12 som driver med utvikling, men de driver også med forvaltning og annet. Det vi lager selv er typisk tilpasninger til de store systemene, og mye integrasjon.

Og dere har 7 hovedsystemer, hva legger dere i hovedsystem?

Jeg tok de viktigste systemene, de som er kritiske for oss. Uten og inkludere de standardsystemene for e-post osv. Vi har også noen sentrale systemer, som er sektorbasert, men vi drifter de allikevel her hos oss. Totalt forvalter vi cirka 40 systemer, hvis vi inkluderer alt. Men hovedsystemene er 7, også bygges det mye rundt. For eksempel har vi ett hovedsystem for økonomi, men så har vi innkjøpssystem, fakturasystem osv, så økonomiavdelingen har kanskje 15 systemer alene.

Er det noe spesielt med hvordan IT-virksomheten er organsiert, som ikke har kommet frem i spørreskjemaet?

Det er en beslutning i styret om at IT skal være skilt fra resten av avdelingene, så vi har en ”leverandørhatt” på oss vi i IT-avdelingen. Også burde det vært på plass ”bestillere” i de andre avdelingene. Det har blitt en utfordring for oss, for når vi skal være leverandør og selge noe, så må vi jo ha noen som vil kjøpe. En kunde ikke sant, men de må jo også læres opp. Så vi har jobbet ganske mye med å få på plass tjenesteavtaler, og vi har også opprettet et internt kundesenter.

Akkurat. Dette henger jo da litt sammen med neste spørsmål, som er "Hvordan foreslås, prioriteres og gjennomføres nye IT-prosjekter i organisasjon?

I 2005 ble det nedsatt et utvalg, som skulle se på IT-avdelingens funksjon. Og da ble det foreslått en del tiltak for å få på plass en bedre styringsmodell, eller IT governance, som har vært litt fraværende her. Altså det har vært veldig styrt fra IT-avdelingen, og fra IT-folkene. Men vi må få brukerne på banen ellers i organisasjonen, og det skulle disse tiltakene adressere. Og det gjorde de forsåvidt ganske bra, men vi holder ennå på med og implementere disse forslagene, og det er jo ikke alle som kommer til å bli implementert. Vi har blant annet fått etablert noe som heter et prosessesutvalg, og det består av folk fra de forskjellige avdelingene, ikke IT-folk. Jeg (IT-sjef) sitter der som såkalt observatør, og skal være med på strategiske avgjørelser og være med å prioritere, men det er veldig brukerfokusert. Også har vi det som heter driftsutvalg, som jeg leder, og er mer teknisk fokusert, og er på en måte et underutvalg av prossessutvalget, som formulerer saken til en beslutning. Og disse to utvalgene er oppnevnt av øverste leder, ikke av avdelingene selv, og er altså en ganske god representasjon. Og det er disse to styringsenhetene som skal gi råd til beslutningene. Så alle nye prosjekter av en viss størrelse, skal gjennom de her to.

Hvor lenge har dere hatt denne styringsmodellen?

Prossessutvalget kom tidlig i 2008, så det er relativt nytt. Og det krever sitt, for det er jo opptatte mennesker dette her, så sakene må være klare til disse møtene. Så dette gjelder alle nye prosjekter, beslutningene skal ut av sentrale avdelinger og forplantes i kjernevirksomheten, det er viktig.

Hvilke effekt ønsker man å oppnå ved IT-investeringer, på generell basis?

Vi på IT-avdelingen har en intern strategi, som sier noe om dette. Men det finnes ikke pr dags dato en IT-strategi ved organisasjonen, det er sagt at dette bare er en del av den totale strategien. Men det har man fått signaler på at man må være litt mer tydelig på det her, så nå skal det utarbeides en strategi. Og det er prosessutvalget, som jeg nevnte i sted, som har et ansvar der. Men IT-avdelingen har og valgt å lage en egen strategi, hva skal vi prioritere og hvorfor skal vi prioritere "sånn og sånn og sånn", og da har vi satt en del målsetning på papir. Men det er veldige vanlige ting, en modell som går på hva eierne forventer og hva kundene forventer. Og på bakgrunn av de behovene og de forventningene vi tror er der, så må vi ha en organisasjon som kan levere. Og for at vi skal ha en organisasjon som kan levere, så må vi ha ansatte som har riktig kompetanse og riktig motivasjon osv. Og under hvert punkt så har vi et sånt hovedmål, som vi bruker når vi prioriterer. Så vi bruker dette som et beslutningshjelpemiddel når vi setter igang prosjekter.

Bruker dere det til og måle også?

Ja vi har noen målbare ting, som for eksempel at vi skal ha en turnover på maks 10%, det er jo målbart. Og innen 2010 eller 2011 skal vi ha definert to nye kjernesystemer, definert i forhold til kjernevirksomheten. Så vi følger opp, og måler det vi kan.

Akkurat, og hva er det dere karakteriserer som en "suksessfull IT-investering"?

Altså, vi er en driftsorganisasjon og det legger vi ikke skjul på. Vi har ansvar for drift og utvikling, men det blir veldig mye drift da. Og vi har som mål at vi skal bli bedre på drift, og

relativt sett levere mer pr krone. Så det forholder vi oss til når vi prioriterer, og velger hvilke prosjekter vi kan gjennomføre og hvilke vi ikke kan. For til slutt så er det jo bunnlinjen vi måles etter, at vi får gjort det vi skal for de kronene vi har. Eller aller helst få gjort mer enda mer enn det som er påkrevd, og hver investering vurderes etter dette.

Skiller dere mellom forskjellige typer vedlikehold når dere planlegger og budsjetterer?

Nå er jeg ikke helt sikker på hva du mener med forskjellige typer vedlikehold, men vi har vedlikeholdsavtaler med forskjellige typer leverandører, så det er jo lett og skille mellom dem i budsjettene. Men angående vedlikehold i systemene vi har utvikliket, og forvalter, så tenker jeg på vedlikehold som at det har kommet en liten feil i systemet, eller at man har en forandring man vil gjøre.

Ja nemlig, det kan jo være en form for vedlikehold.

Ja, faktisk det vi har i avtalene våre (med avdelingene) er at det er delt opp i "Drift" og "Vedlikehold". Så der prøver vi å skille, men det vanskelig i forhold til budsjetter. I år kan det kanskje være nesten ingenting (vedlikehold), neste år kan det være tre ganger så mye.

Hva med integrering og tilleggsfunksjonalitet som du snakket om tidligere, går det innunder vedlikehold?

Ofte så blir det litt utenom avtalene våre, at de andre avdelingene kjøper det som en tilleggstjeneste. Men noe vi begynte med for 4-5 år siden, var at folk skulle si ifra hva de holdt på med. Om de holdt på med forvaltning, eller om de holdt på med drift eller om de holdt på med utvikling. Så bruker vi erfaringstall for å beregne priser, og så justerer vi. Så vi gjør det, men det blir jo litt "best guess", men jeg mener det er ganske kvalifisert.

Hva mener du, og IT-avdelingen, er viktige faktorer for at et IT-prosjekt skal bli vellykket?

Der har jeg ganske mange meninger. Det som ofte er et problem, og det gjelder ikke spesifikt vår organisasjon men veldig generelt, og det er at man ikke alltid er like flink til å tenke på at et IT-prosjekt har alltid en organsisasjonsmessig konsekvens, som man glemmer litt bort underveis. Og man gjør jo et IT-prosjekt forhåpentligvis for å få mer funksjonalitet, bedre kvalitet, bedre økonomi osv. Men man vet jo det at prosjektet i seg selv kan aldri få noe gevinst i seg selv, det er det organisasjonen i etterkant som må få. Det kan jo være at folk i organisasjonen faktisk må endre arbeidsprossessene sine. Og det har vi litt lett for å glemme, så det er viktig å henge fast i det organisasjonsmessige aspektene midt oppi dette her. Også er det de vanlige faktorene, at det skal være målbart, det skal være akseptert i organisasjonen osv. Vi har jo noe som heter SMART (Specific, Measurable, Attainable, Relevant, Time-bound), som beskriver viktige aspekter du må tenke over før du starter et prosjekt. Og det tror jeg er viktig at man legger mye fokus på.

Hva med verktøy og metoder, tror du er viktig?

Ja, vi har jo med hell de siste par årene brukt agile metoder. Ikke bare når det gjelder utvikling, men også på infrastruktur ting ser vi at det har en verdi for oss. Så det har vi ganske god erfaring med. Ellers prøver vi å bruke "best-practice" der det er etablert sånne.

Har organisasjonen en plan for å minimere andelen "ikke produktivt IT-arbeid", i form av arbeid som ikke tilfører ny funksjonalitet?

Det er klart at vi har en diskusjon akkurat nå, og har hatt den diskusjonen en stund, på at vi må kanskje begynne å tenke litt annereldes. IT-systemene har utviklet seg over tide, og vi trenger kanskje og tenke på en ny arkitektur i fremtida. Vi ser blant annet at vi bruker veldig mye tid på integrasjon og tilpasninger med gamle system, og vi har ikke noen god system arktitektur. Ikke god nok i hvert fall. Og det blir mye uproduktivt arbeid sånn sett. Vi har startet planlegging, og ser for oss at vi gjør noe konkret med det fremover.

Det fører til et annet spørsmål, nemlig "Benyttes, eller er det planer om å benytte en SOA-arkitektur? Hva legger organisasjonen i SOA, og hvordan implementeres eventuelt SOA i organisasjonen?"

Nei, altså SOA er jo ingen teknologi men mer et tankesett og en arkitektur. Og hvordan det skal implementeres, da er jeg egentlig på grensen av hva jeg kan snakke om sånn rent kompetansemessig. Men jeg forstår såpass at vi har jo tenkt på en måte tjenesteorientert lenge da, vi har det vi kaller et tjenestelag faktisk, som er gammelt. Men det leverer tjenester på et nivå, og andre systemer kan hente disse standardtjenestene. Men teknologien i bunn er nok ikke slik som man vil ha, når man tenker på SOA idag. Så vi må begynne å tenke nytt. Og et punkt som vi ser er en kjempeutfordring hos oss er personopplysningslagring, vi har ansatte, kunder, vikarer med og uten norsk personnummer som ikke får tilgang til systemene, fordi systemene krever dette. Og disse har kanskje ikke tilgang på flere uker, fordi vi må "hacke" systemene først. Og da ser vi med en gang, skal vi gjøre en forbedring her, vil vi med en gang tenke tjenesteorientert.

Så dere håper på en glidende overgang etterhvert som dere gjør store forandringer?

Ja, det tror jeg er nødvendig. Men vi må først bli enige om internt først hvordan ting skal gjøres, og hvilke kompetanse vi trenger og hvordan vi skal få minimert det gapet fra der vi er nå til der vi vil være. Så håper vi etterhvert og få mindre kompleksitet i systemene våre. Når vi har cirka 40 systemer, og 12 stykker som jobber med dette her, så ser vi at det blir sårbart.

Dere har tre erstatningssystemer under utvikling nå. I den forbindelse; hva er motivasjonen for å lage disse erstatningssystemene?

Her har det vært at det er gamle system som ikke tilfredstiller dagens behov, og gjerne bygget på gammel teknologi, som det igjen er vanskelig å finne kompetanse til å vedlikeholde. Men først og fremst at de ikke lever opp til dagens krav?

Så de mangler rett og slett funksjonalitet?

Ja, det også. Og på disse systemene koster det ofte mye og flikke på, og legge til funksjonalitet. For eksempel intranettet vårt, det er skrevet i PHP og per idag har ikke vi så skrekkelig mye kompetanse på det. Vi er hovedsak JAVA-utviklere. Så det er krevende og ha flere plattformer, så selv om ikke PHP er utdødd så må vi velge noen teknologier.

Har du inntrykk av at man får gjennomført alle de målene man gjerne har når man går igang med et erstatningssystem prosjekt?

Nå har ikke jeg gjennomført så voldsomt mange slike prosjekter, men jeg tror man rett og slett må godta at man ikke kan få med alt som var før når man bytter system. At man mister noe av det man hadde, og det er litt tungt. Derfor har systemene en tendens til å leve litt lenger enn de kanskje bør. Så jeg tror man må akspetere at man får ikke med alt når man bytter system, selv med noe så enkelt som et epostsystem eller en kalenderløsning kan det være forskjeller fra en leverandør til en annen. Så jeg tror det er vanskelig å erstatte alt, pluss at man skal få ny funksjonalitet i tillegg.

Men du mener man får lagt til den nye funksjonaliteten man ønsker?

Tja, det kan du si men ofte skal jo disse nye systemene leve i mer enn ti år. Så man kan jo legge til noe av den nye funksjonaliteten etterhvert, det viktigste er å først få på plass det som må på plass.

Når det gjelder gjenbruk har du svart at dere bruker dette i "Svært liten grad" både når det gjelder spesifikasjon, design og kode.

Ja, det stemmer. Ofte er dette svært gamle system, og spesifikasjon og design er kanskje ikke å oppspore. Og koden er gjerne den man vil bytte ut, så da er det ikke så aktuelt med gjenbruk her heller.

Men har dere som mål at dere vil ha gjenbruk?

Nei, vi har egentlig ikke det. Når vi snakker om SOA tenker vi på gjenbruk av hele systemer, men ikke ellers.

Siste spørsmål er kanskje generelt; hva oppleves som den største utfordringen når det kommer til vedlikehold av IT-systemer?

Nei hos oss, konkret, så er det faktisk kompetanse. Det går tid, og tilpasninger blir gjort, og så forsvinner de som har gjort tilpasninger og kan systemene. Så jeg tror faktisk det er kompetanse, som forsvinner over tid. Så her ser vi at vi kanskje burde vært smart da, og satt ut (outsourcet) mer på spesielle ting, på områder hvor det faktisk finnes miljøer som er i stand til å ta seg av enkelte systemer. Det er jo andre som har gjort det, og fått mindre problemer av det. Vet ikke om andre innenfor våre sektor, men vet ihvertfall at Kommunen outsourcer en del med suksess. Men det er også viktig at du ikke setter ut for mye, at du fortsatt sitter igjen med nok kompetanse til å styre. At man i det minste har en bestillerkompetanse igjen i organisasjonen.

Intervju B

Første spørsmål; hvordan er IT-virksomheten organisert hos dere?

Hos oss så har vi en avdeling som heter ”Finans og Stab”, som er en stabsfunksjon, som dekker personal, økonomi, regnskap og det som vi har kalt for ”IT og Arkiv”. ”IT og Arkiv” er et område i avdelingen ”Finans og Stab”. I ”IT og Arkiv” sitter det en IT-koordinator, som da er undertegnede nå, innleid. Også er det to faste og en innleid, primært arkivressurser, selv om den ene av dem også har en IT vinkling. Så det er organisasjonens faste ansatte på IT-området. Også kjøper vi jo veldig mye tjenester, vi har en driftsavtale med Atea, som er todelt. De har både et ansvar for å holde infrastrukturen ved like, og så har de en on-site support.

Så dere kjører deres egne systemer her?

Ja, vi eier infrastrukturen og vi har maskinparken stående her, og applikasjonene kjører her. Så har vi jo ytterligere rammeavtaler innefor områdene systemutvikling, og IT-rådgivning. Og der har vi fem leverandører på den rammeavtalen, også har vi kjøp av programvare og konsulenttjenester som ikke faller inn under rammeavtalen, mer på sporadisk basis. Altså, vi kjøper mye tjenester, men nå er det en liten endring der. IT-koordinatoren som nå er en innleid rolle, skal gå over til å bli en fast ansettelse. Så det blir en IT-leder, om et par måneder. Det er organiseringen som sådan.

Neste spørsmål; hvordan foreslås, prioriteres og gjennomføres nye IT-prosjekter?

Vi har jo et verktøy som vi kaller for handlingsplan, det er jo en årsplan hvor det ligger inn et sett med prosjekter. Som inisielt oppstår enten som behov i organisasjonen, nede i produksjonsenhetene, i tillegg har vi i ”IT og Arkiv” en IT-strategi, vi har relativt nylig gjennomført en arkitektur studie som har gitt oss noen anbefalinger som vi prøver å gjennomføre. Dette er over en relativt lang tidshorisont, men allikevel, dette gir oss føringer i forhold til aktiviter og prosjekter som skal gjennomføres, innenfor det handlingsplanen dekker, men også innenfor en enda lenger tidshorisont. Også gjelder det da og plassere disse prosjektene som er avhengig, i en fornuftig rekkefølge. Også er det ofte sånn at dukker det opp virksomhetskritiske behov eller at det kommer krav fra myndighetene, vi har jo akkurat vært gjennom en litt hektisk periode hvor det kom da nye friske midler til organisasjonen som skulle fordeles. Også kalt ”tiltakspakken”. Den gjorde at vi på IT-siden måtte gjøre noen grep, for å støtte det tiltaket. Og det fikk da en naturlig prioritering, siden det var veldig tidskritisk. Men i det store og hele så prøver vi å legge en plan for året, som inneholder aktiviter og prosjekter per kvartal.

Så en del av prosjektene foreslås av avdelingene, også prioriteres de av dere i ”IT og Arkiv”?

Ja, det kan du si. Men større prosjekter går gjerne opp til ledergruppen for prioritering. For knytta til en handlingsplan, så ligger det også et budsjett. Der har man gjort noen estimater for hva et prosjekt vil komme på, og sådan dekket opp for budsjettet. Så hvis det da dukker opp større aktiviteter som det ikke er tatt høyde for i handlingsplanen, eller da i budsjettet, så må

man jo gjøre en omprioritering, eller se om det finnes tilgang til finansielle kilder som ikke er innenfor det budsjettet vi har.

Du var såvidt inne på IT-strategi. Har dere i organisasjonen formulert hva slags effekt de ønsker å oppnå av IT-investeringer generelt?

Til en viss grad ligger det både resultat og effektmål til hvert prosjekt, det gjør det. Men vi gjennomfører ikke noen kost/nytte vurdering, eller gjør noen kalkyle på pengemessig gevinst. Det går mer på kvalitative mål. Vi kunne ha gjort mer på det området, men jeg føler at alle prosjektene har en gitt intensjon og de er relevante og det er bevissthet rundt de.

Så hva vil dere anse som suksess for en IT-investering, altså hva kan et slikt mål være?

Det kunne være å gjøre systemet mer fleksibelt, til å imøtekomme varierende behov og krav. Det kan være å gjøre et system mer robust. Det kan være å øke ytelsen i infrastrukturen. Det kan være at vi mulgjør nye arbeidsformer. Hvis vi kjøper en programpakke som vi kan se for oss, helt konkret, så ser vi jo litt på Office Communicator Server nå, som vil gjøre det mulig å jobbe på en litt annerledes måte enn vi vanligvis gjør. Så det er nok et sett av forskjellige mål vi kan sette.

Angående vedlikehold, skiller dere på forskjellige typer vedlikehold når dere planlegger og budsjetterer?

Ja, det blir jo litt systemorientert. Du kan si, generelt vedlikehold på infrastruktur, så er jo det noe vi vanligvis håndterer innunder Atea sin driftsavtale. Hvis vi må oppgradere hardware. På applikasjonsvedlikehold har vi separate avtaler. Enten med en programvareleverandør hvor vedlikehold er relativt enkelt, enten ved support eller oppgraderinger, patcher og etcetera. Det tyngste systemet som vi har på vedlikeholdssiden er jo sakbehandlingssystemet, som er et skreddersydd system utviklet av ErgoGroup. Dette krever en del vedlikehold. Både i forhold til at det er kontinuerlige mengder feilsituasjoner knyttet til det, samtidig som vi har stadige mindre endringsønsker knyttet til det. Da har vi en spesiell vedlikeholdskontrakt med Ergo på det.

Ser dere forskjell på om det arbeidet ErgoGroup utfører er rent vedlikehold, eller evolusjon av systemene, altså mer videreutvikling? Dette blir jo gjerne kalt vedlikehold begge deler, så det er interessant om organisasjoner i det hele tatt tenker over forskjellen.

Vi burde nok kanskje bry oss mer om det. Men de er veldig beslekta, og det ville vært mye administrasjon om vi skulle hatt et tungt regime på å skille mellom feilretting og funksjonsjusteringer. Så for vår del er det mest effektivt og la kontrakten dekke begge deler.

Ja, når begge deler er 100% outsourcet, så er det vel kanskje ikke noe stort poeng å skille på det.

Nei, men det er klart, vi kan jobbe mer på det med oppfølging av saker. Og vi har snakket mye med de om å få bedre rutiner, og systemstøtte, på ren feiloppfølging. Eller problemhåndtering. Så der skal vi kanskje gå igang med et nytt system som gjør det enklere å følge opp. Forsåvidt også varsle, og følge opp feilsituasjoner.

Hva er de viktigste faktorene for at et IT-prosjekt hos dere skal bli vellykket?

Det er jo de veldig tradisjonelle da, at man klarer å levere innenfor tid og budsjett. Noen prosjekter er tidskritiske, mens i andre er det viktigere at vi får kvaliteten på plass, og at det blir ordentlig. Så det vil nok være det overliggende suksesskriteriet. Også er det som sagt viktig å prøve å holde prosjektplaner, tidsplaner og kostnadsplaner.

Og er det noen spesielle faktorer dere tror er viktig for at man klarer å overholde disse målene?

Altså sånn som vi gjennomfører prosjektene, så bruker vi jo Statens avtaleverk som kontraktregime, for å styre prosjektene. Og det fungerer bra, leverandørene er med på det. Og det er ikke mye rundt det. Det som viser seg ofte å være, altså leverandørene blir jo valgt ut basert på en konkurranse mellom de leverandørene som er på rammeavtale. Og da ligger det jo estimater, og kostnadsrammer, i avtalen når vi inngår den. Det som ofte viser seg å være en utfordring. Det går litt på vår egeninnsats, det å prioritere nok tid til å engasjere seg i prosjektet. Og det blir veldig vanskelig å utfordre leverandørene, i forhold til vår reduserte egeninnsats. Og det kan være en årsak til at prosjektene sklir på tid, det har vi nok noen erfaringer med.

Har organisasjonen et ønske, eller mål, om å minimere andelen vedlikeholdsarbeid? Tenker da på "ikke-produktivt" vedlikeholdsarbeid, som ikke tilfører ny funksjonalitet eller lignende.

Ja, forsåvidt er jo det et ønske. Det vi gjør på det saksbehandlingssystemet nå, er at vi programmerer om en gammel løsning. Det som har vært tidligere kun ASP kode, skal nå over på .NET teknologi. Som jeg har forstått skal være langt enklere å vedlikeholde. Vi får også bedre systemdokumentasjon. Dette er jo det viktigste systemet vi har, og har blitt utviklet utifra organisasjonens opprinnelse. Det er derfor viktig at dette er robust, og et enkelt system og vedlikeholde. Og vi har en veldig dyr vedlikeholdskontrakt idag, som kanskje kan forbedres fremover.

Nemlig, og da har du egentlig begynt å svare på neste spørsmål som er; hvorfor ønsker man å lage erstaningssystemer, hva er motivasjonen?

Det er for å lage de mer robust, enklere å videreutvikle og enklere å vedlikeholde. For det er jo gammel..., eller gammel er jo et feil ord og bruke, men ting skjer så fort. Ihvertfall de som forestår vedlikeholdet de ser betydelige gevinster, ved å gjøre omprogrammering. Men vi har jo også en langsiktig plan rundt det saksbehandlingssystemet, hvor vi er inne i en analyserunde nå, og ser på hvilke systemtyper kan erstatte dette systemet på lengre sikt. Nå har vi et pågående omprogrammeringsprosjekt, også ser vi for oss at på et to-tre års perspektiv burde vi kanskje ha vurdert, og muligens også implementert et nytt system, som ikke nødvendigvis er en skreddersømsløsning fra før av. Kanskje finnes det en mer standarverktøy, applikasjoner, hyllevare som dekker noe av behovet. Og da vil vi sannsynligvis, hvis vi velger den varianten, så vil vi nok klare å redusere vedlikeholdet betydelig.

Når dere lager dette erstatningssystemet, ønsker dere da kun en kopi av det gamle systemet? Eller ønsker dere også å implementere ny funksjonalitet samtidig?

Akkurat der har vi valgt og ikke tenke ny funksjonalitet i denne omgang. Altså vi har ny funksjonalitet kontinuerlig i det gamle systemet, og det vil jo måtte tas inn i den nye løsningen.

Slik at når vi skal erstatte den gamle løsningen, så har vi fått med oss alle endringer som har skjedd mens man har utviklet den nye løsningen. Sånn at vi skal ikke gå tilbake på funksjonalitet i det nye systemet, det er en ambisjon. Men det ligger ikke noen plan om å legge til noen ny funksjonalitet i erstatningsperioden.

Gjelder dette generelt, eller kun for dette prosjektet?

Nei, man kunne sett for seg at man kan gjøre andre ting samtidig. Men grunnen er at vi rent konkurransemessig, det vil si det er litt begrenset hvor mye nye ting vi kan legge inn i systemet, uten å kjøre en konkurranse på det. Så nå som vi lager et rent erstatningssystem, så er det den eksisterende leverandøren som gjør jobben.

Er gjenbruk av spesifikasjon, design og kode et mål for dere?

Lite prioritert. Det er et tema som vi ikke har hatt mye fokus på. Det kan jo bli et større fokus på dette under en tjenesteorientert arkitektur, som vi definitivt planlegger å bevege oss mot.

Riktig. Det er faktisk neste spørsmål; benyttes, eller er det planer om å benytte, en SOA-arktitektur?

Det kan jeg forsåvidt si ja til. Vi har hatt et analyseprosjekt som har konkludert med et målbilde, som definitivt er tjenesteorientert.

Og hva legge dere da i "tjenesteorientert arkitektur"?

Det ligger i det å gjøre et arbeid, hvor vi designer et sett av tjenester, med god innsikt i de systemene, de dataene og de prosessene som er viktige for organisasjonen. Se om det går an å identifisere fellestjenester. Og det kan jo være alt fra brukerhåndtering og sikkerhet på et lavt nivå, og på mer funksjonorienterte tjenester. Og det å jobbe med de i første omgang, før man da klarer å utnytte tjenestene i en større funksjon da. Og så å tilby tjenestene i en brukerportal. Det er litt sekvensen i det, etterhvert også bygge på med rapportering. Vi har gjort en type tankearbeid rundt i hvilken sekvens er det naturlig å implementere denne tjenesteorienteringen.

Ser man for seg at man da vil beholde gamle applikasjoner, under dette tjenestelaget?

Vi er inne i en diskusjon rundt det saksbehanldingssystemet, som er vår viktigste applikasjon. Der har vi to eller tre alternative løsningstyper på sikt. Første alternativ er hyllevare, finnes det standardverktøy. Eller skal vi videreutvikle, og tjenesteorientere den .NET varianten som vi nå er i feil med å implementere. Eller kanskje en kombinasjon kan være et alternativ. Så der har vi ikke inkludert enda. Men har en studie ute hos leverandører, som skal komme med innspill til oss, i forhold til hvordan vi bør gå frem i forhold til det konkrete systemet.

Hva oppleves som de største utfordringene i forbindelse med vedlikehold?

Nei, det blir nok litt tilfeldig hvordan vedlikeholdet foregår. Det finnes sikkert både gode og dårlige måter å vedlikeholde på, det finnes jo rammeverk for å stramme opp vedlikeholdsrutiner. Og vi har fortsatt en veldig ad-hoc tilnærming til vedlikehold. Så det vi kunne vurdert er jo å implementere for eksempel noen av ITIL sine prosesser rundt vedlikeholdsarbeid. Så et strammere regime rundt vedlikehold er nok noe vi burde jobbe med.

Dere bruker vel mer ressurser på vedlikehold enn på utvikling. Anser dere det som et problem?

Ja, altså det ligger nok et kostnadspotensiale... altså vi har litt for mye egenutviklede systemer som fordrer mye vedlikehold. Så å bevege seg mer mot standardverktøy og hyllevare der det er mulig, det er en ambisjon. Og da vil vedlikeholdsomfanget forhåpentligvis reduseres. Men da er du samtidig prisgitt hyllevarens utviklingsløp. Du har mindre muligheter for å gjøre tilpasninger til det. Så det er denne balansen da. Har du et system som er veldig orientert mot vår egen virksomhet, så krever det kanskje mer skreddersøm, og da følger tydelig vedlikehold i kjølvannet.

Hvis vi kun snakker om skreddersydde systemer, kan du se noen faktorer som fører til at det blir så mye vedlikeholdsarbeid?

Jeg tror det er delvis teknologiorientert. Mangel på kompetanse hos leverandøren. Det store systemet ble utviklet av en leverandør, og av et fåtall personer hos leverandøren. Når enkeltpersoner der slutta, så måtte de bygge opp mer kompetanse på det. Og da var jo det delvis en greie som kostet oss noe, i den forstand at det var vanskelig å gardere seg avtalemessig mot at noe sånt skjedde. Men det har vi delvis løst opp i nå da. Så det er sånne forhold som kan spille inn på de konkrete prosjektene.

Akkurat. For det vi har prøvd å se på er jo hvorfor det stadig er sånn at vedlikeholdsarbeid dominerer ressursbruken i forhold til utvikling. Utifra spørreundersøkelsen vi har gjort klarer vi ikke å finne noen tydelige faktorer som påvirker dette forholdet. Har du noen forslag til hva som kanskje kunne redusert vedlikeholdsarbeidet hos dere?

En annen ting vi ikke har vært inne på, er at enkelt feil og enkelt behov, løses enkeltvis og releases enkeltvis. Vi har ikke noe regime i forhold til releasehåndtering, og versjonhåndtering. Det kunne man se for seg, at man samlet opp alle vedlikeholdsproblemer i litt større bolker, og lanserte i nye versjoner. Det er det.

Du ser for deg at det kanskje kunne effektivisert vedlikeholdsarbeidet?

Ja, jeg ser ikke bort ifra det. Men det er bare... jeg ingen empiri på det. Det er litt i forhold til responsen jeg får. Altså, er det kritiske feil, da har man ikke anledning til det. Men de som er mindre alvorlige, og kosmetiske feil/endringsønksker, at det blir samlet opp i større bolker. Vi har en utfordring her, og det er at vi har ikke hatt et veldig godt testmiljø. Nå har vi gjort noen grep, og fått opp et testmiljø, slik at ved ny funksjonalitet gjennomfører vi testing i et testmiljø. Dette krever kanskje mer av oss, sånn i tid, men det kan hende at vi unngår følgefeil, som vi ofte erfarer når vi lanserer enkeltløsninger på enkeltfeil, uten at det er samlet opp i større bolker og gjennomtestet. Vi overlater veldig mye av testingen til leverandør, og kunne kanskje tatt en større del av det ansvaret selv.

Du nevner at dere overlater kanskje litt for mye til leverandøren. Er det annen problematikk som går igjen med tanke på at dere outsourcer mye, eller synes du dere stort sett håndterer det bra?

Vi kunne nok hatt mer kompetanse in-house, vi er nok litt tynt besatt. Men jeg tror vi har gradvis blitt flinkere, for vi har gjennomført en del prosjekter. Og gradvis får vi en større

forståelse av de utfordringene som ligger i et IT-prosjekt, og sånn sett kan være en mer moden sparring partner ovenfor leverandørern.

Intervju C

Hvordan er IT-virksomheten organisert hos dere?

I utgangspunktet er vi jo akkurat i en brytningstid, og derfor var det også litt vanskelig å fylle ut spørreskjemaet deres. For vi er jo midt mellom det å ha en egen tradisjonell IT-avdeling, og det å outsource det aller meste av IT. Fram til nå nylig har vi gjort det aller, aller meste selv. Organisasjonens fagsystem per i dag er en gedigen mastodont av et gammelt stormaskinmiljø, som ligger på Gjøvik og vedlikeholdes av Ergo. Og den inneholder det aller meste av det organisasjonen trenger av fagsystemer. Men, så har vi oppigjennom årene gjort noen tapre forsøk på å modernisere, og da har vi bygget til en del systemer rundt det. For eksempel portal, og standard systemer som økonomisystem og timeregistreringssystem og sånne ting. Så det er på en måte et grensesnitt inn mot hovedsystemet vårt. Så når vi hadde det her var vi organisert som en helt tradisjonell IT-avdeling, og var oppe i 40 personer den gangen, hvor vi da hadde en gruppe på 5-6 funksjonelt ansvarlige, og 15-20 utviklere. I tillegg noen ledere og noen som hadde med infrastrukturen å gjøre. En forholdsvis tradisjonell strukturering på 90-tallet. Også ble vi pålagt av Stortinget, med rette for så vidt, at vi skulle modernisere. Dvs, vi ba jo om det selv, men vi gehør og vi fikk penger, men forutsetningen var at det skulle skje gjennom en utkontraktering av tjenestene. Dvs at organisasjonens IT-avdeling skal legges ned, og erstattes av noe nytt. En del av den omstillingen går jo kompetansemessig på å bygge ned teknisk kompetanse, eller kanskje kan du si spisse innenfor en del nyere teknologi. Men også bygge opp større funksjonell kompetanse. Det er den veien jeg opplever at det går. De profesjonelle får i større grad ta hånd om det tekniske, for det setter vi ut, også må vi ivareta våre egne kjerneprosesser. Og det er det vi må ha inngående kunnskap om.

Så dere ønsker kun å sitte igjen med bestillerkompetansen?

Ja.

Akkurat. Intervjuet handler jo både om erfaringer du har gjort deg, og hvordan dere ser for dere fremtiden. Så da får du snakke om "gamle dager" når det gjelder erfaringer, og så kan du jo snakke om den nye organiseringen der hvor det passer seg.

Neste spørsmål er da, hvordan forelåes, prioriteres og gjennomføres nye IT-prosjekter?

I veldig stor grad er jo IT-prosjekter styrt av politisk myndighet. Fordi Stortinget vedtar nye regler og lover som vi må implementere. Det er den aller, aller største biten av det. Så har vi en liten egen pot, som på en måte har med egenutvikling å gjøre. Ting som vi ønsker å gjøre, fordi vi vil bli bedre, og de prioriteringene der tas på ledernivå. Altså i dag er det ikke IT-avdelingen som gjør det, men det er ledernivået som gjør det. Det har ikke alltid vært sånn, for på 80 og 90-tallet var IT-avdelingene svært mektige. Og det har vært en helt klar trend hos oss, at IT-avdelings maktposisjon har blitt svekket, til fordel for at ledelsen skal prioritere. Og det er en trend vi kommer til å fortsette på, og i fremtiden er det en målsettings at vi skal etablere endringshåndteringsstrukturer som er i tråd med en litt mer moderne tankegang.

Har dere ekspertbrukere eller lignende, som er med avgjørelser angående IT?

Det prøver vi å etablere. Hvis vi ser på dagens løsning, så er kompetansen altfor sterkt knyttet til IT-avdelingen. Vi ønsker å lage ekspertbrukere rundt omkring i de forskjellige fagavdelingene, som har sine områder. Og måten vi gjør det på, det er jo at vi involverer dem i det store omstruktureringsprosjektet vi har nå. Nå sitter jeg sammen med folk fra alle kanter av organisasjonen, og de har hver sine ansvarsområder.

Neste spørsmål; hvilken effekt ønsker man å oppnå med IT-investeringer?

Vi har erfart veldig tydelig, i det offentlig i hvert fall, mulig det er sånn i det private også, så er kravet til endringshastighet så stor innimellom. Fordi politikerne kommer på ting. Du kan se for deg at den 23. Desember vedtas statsbudsjettet. Og helt opp til det kan det komme innspill, som skal være gyldig fra 1. Januar. Det er jo praktisk umulig ikke sant. Men det betyr at de viktige IT-strategiske grepene vi må gjøre er at vi må ha en veldig ryddig og god teknisk arkitektur, som klart skiller funksjonen til de enkelte tjeneste og applikasjonene. Vi har blant annet innført en regel, som gjør at de funksjonelle handlingsreglene vi blir pålagt, de ligger et sted. Kodeverk skal ligge et sted. Og slike ting, det er viktig. Også må vi ha en struktur i dataene som gjør at dette er endringsvillig. Vi må kunne implementere endringer svært fort, uten at det korrumperer det som allerede er. Så det er nok den viktigste strategien. Og også det at teknisk kompetanse skal ut, det står en leverandør for, også har vi bare bestillers ansvar.

Har dere mål for disse kriteriene?

Vi kommer til å lage det. Moderniseringsprosjektet som startet for et år siden, har første leveranse nå rett over sommeren. Og da kommer vi til å måle det. Det er klart at det har vist seg at en sånn strategi er dyrere, og mer komplisert enn det man i utgangspunktet tror. Det har en pris, det må vi være veldig klare på. Så det jeg tror kommer ut av dette er; ”Fint og bra, og veldig riktig, men det koster”.

Skiller dere mellom forskjellige typer vedlikehold? For eksempel ved planlegging og budsjettering.

Vi har egentlig en tre-nivå tankegang. Vi deler inn i ren drift, som har med hardware og infrastruktur og gjøre. Så har vi vedlikehold, som vi kaller det. Det er egentlig en veldig passiv del, primært feilretting av alvorlige feil. Kanskje også litt ytelse og sånne ting, og typisk preventivt vedlikehold. Den siste gruppa har vi kalt for videreutvikling. Og det er ikke prosjekt vel og merke, det er små videreutviklingsoppgaver som ikke blir prosjekt.

Akkurat. Er dette noe dere skiller mellom helt konkret i budsjetter, eller er dette bare en definisjon dere har når dere snakker om vedlikehold?

Dette gjenspeiles i budsjett.

Neste spørsmål; hva er de viktigste faktorene for at et IT-prosjekt skal bli vellykket?

Ja, noen nøkkelbegreper der. Vi kan begynne litt innenifra virksomheten. For en etat som oss, altså vi er jo offentlig, og vi har alltid drevet IT selv. Så det vil si at saksbehandlere kan komme å snakke med utviklerne. Så vi har etablert en kultur, og den må brytes. Nå er det kost/nytte som gjelder. Suksessfaktoren for at dette skal lykkes er at lederne og mellomlederne forstår dette. Nummer to er at toppledelsen har hundre prosent ”attention” på prosjektet, og har tillitt til prosjektet. Det tredje er, organisasjonen må stille nok kompetanse og ressurser til rådighet, for

prosjektet. Da har vi snakket litt om det interne. Også må vi se litt på kontrakten. For kontrakten er avgjørende. Den konkurransen som har gått på forhånd, og den kontrakten vi faktisk får. For den kommer vi tilbake til hele tiden. Og et suksesskriterie er at vi som eier kontrakten, vi må kunne kontrakten. Vi må kjenne våre deler. Også må vi ha en omforent holdning til hvordan vi skal håndtere kravene. Hvis man ikke er fleksibel og ryddig på den biten der, blir prosessen veldig tung. Så det betyr at kjennskap til og omforent holdning til hvordan kontrakten skal behandles. Og så spilles det over til leverandøren. Først må vi kunne det, og så må vi ble enig om leverandøren. Problemet på leverandørsiden er at den består også av skarpe, og mindre skarpe kniver. Og det er klart at jo skarpere kniver du får, jo bedre går det. Det er menneskene som avgjør det.

Disse tingene du nevner, er det ting dere har opplevd som problematiske, og sånn sett fått forståelse for viktigheten av?

Ja, vi ser det i etterkant ikke sant. Når du kjører en konkurranse, det tar gjerne et år. Og da begynner man ofte og bli lei, og sulten på å bli ferdig. Ikke bli det. Ta den tiden det trenger. Og det vil si, få gode svar på alle krav. Avklar uklarheter hele veien. Det får man igjen. Vi brukte en PS2000 kontrakt, og aingsmodellen, den måten prosjektet skal gjennomføres på er vel så viktig som kravene. Er ikke metoden god, så svir du til slutt. For da er det så mye som glipper. Bruk tid på å spikre metoden, det mener jeg også er et suksesskriterie. Dette er det vi merker nå at vi sliter mest med.

Har dere en plan eller et ønske for å minimere andelen ikke produktivt vedlikehold, eller såkalt upkeep?

Klart, det ideelle hadde jo vært om vi ikke hadde hatt det. Men sånn er ikke verden. Men du kan si at jo bedre du har gjennomført prosjektet, jo mindre slikt arbeid for du. Men preventivt vedlikehold for eksempel, det viser seg at det gir i sum god betaling. For da får du mindre trøbbel, og mindre feil. Så vi har ikke hatt noen diskusjon om dette. Sånn må vi bare ha det. Nesten sånn at vi kanskje burde gjort det enda mer, for vi får igjen på et annet budsjett. Vi angrer for eksempel bittert på at vi ikke har vært flinkere til vedlikehold av design dokumenter tidligere, det får vi svi for nå.

Akkurat, da kan vi jo hoppe litt. Det er nemlig et spørsmål vi kommer til; er gjenbruk av spesifikasjon, design og kode viktig for virksomheten?

Ja, det er det. Men vi har ikke vært flinke nok tidligere, ergo må vi finne opp altfor mye krutt pånytt.

Har dere retningslinjer for gjenbruk?

Altså, nå har vi etablert det. Vi etablerte det i 2001. Så derfra og ut har vi. Men 70% av den gamle basen er laget før det. Og der er det både dårlig kode, og dårlig dokumentasjon. Og se situasjonen, de som laget det, de er pensjonister i dag. Og du kan jo tenke deg hvor lang tid det tar når andre skal forstå det de har laget pånytt. Vi sitter og leser COBOL kode for å forstå, og det er ikke effektivt altså. Og sånn som arbeidsmarkedet har blitt nå, folk flytter på seg hele tiden, og så kunnskapsintensivt dette her er, så har vi ikke råd til å la folk gå med den kompetansen.

Og vi har kanskje vært inne på det, men hva er motivasjonen deres for å lage erstatningssystemer?

Det vi gjør nå, er vel erstatningssystemer. Men vi lager jo alt på nytt. Men funksjonaliteten skal beholdes, og så litt til. Det har kommet nye funksjonsområder som vi kan ta i bruk.

Så dere legger til funksjonalitet i samme prosjekt?

Ja, det gjør vi faktisk.

Og hvordan går det?

Det er vondt og vanskelig. For en utvikler er det alltid enklest om han blir fortalt, lag det samme som der. Vi har blant annet lagt til en helt ny behandlingsform av en helt ny sakstype, og det har svidd. For vi må finne opp så mye rart underveis.

Så dere har fått lagt til ny funksjonalitet, men er ikke helt fornøyd med gjennomførelsen?

Vi får det jo igjennom, men det går på bekostning av tid.

Tror du dere heller burde ventet og lagt det til senere?

Ja, det er vel noe av det vi har lært nå. Men så er det også en tredje måte å gjøre det på. Dette prosjektet skal gå frem til 2013-2014. Og vi har allerede nå avdekket et par forenklinger vi ønsker oss, et stykke utover. Og de skal vi begynne med piloter på nå, slik at vi får avklart ting før prosjektet begynner. For det blir rett og slett for dumt å implementere noen av de tungvinne løsningene vi har i dag. Så det er en tredje måte å gjøre det på. Hvis man skal jobbe over en viss tid.

Benyttes, eller er det planer om å benytte en tjeneste orientert arkitektur? Det har jeg jo forstått at dere er i gang med, du kan jo si litt generelt om det.

Ja. Vi har en funksjonell tilnærming, og en teknisk. Vi har gått for en Microsoft løsning, ned BizTalk i midten. Så BizTalk er tjenestebussen, som sikrer en konsistent behandling av alle data. Samtidig har vi, som vi har snakket om, at kode skal bare ligge et sted. For å sikre gjenbruk, og få slutt på redundans. Men så har vi jo også brukt tjeneste orienteringen funksjonelt. For designmessig er det enklere å tenkte tjenester, enn applikasjoner. Så vi har etablert en funksjonell tjenestekatalog, som er uavhengig av hvordan de teknisk skal implementeres. Og det er en fin måte å kommunisere med leverandøren på. Også tar leverandøren den, og designer den sånn som er teknisk optimalt og fornuftig. Og det man sikrer da, er en mye sterkere funksjonell sporbarhet til det tekniske. Det er på en måte en logisk sammenheng i det hele. Og igjen kommer jo sånne ting som vedlikehold inn, og det med tjeneste orientering gjør det enklere å lage grensesnitt mellom et nytt system og eksisterende system. Det gjør overgangen mye mer ryddig. Så ryddighet og forståelse er det som er det viktigste her. Men vi ser at vi kan ikke holde oss til alle prinsippene. Det blir for dyrt, går for sent eller blir ikke god nok ytelse på det. Så vi må innimellom fravike. Men det har vært en veldig ryddig måte å jobbe på.

Ifølge skjemaet du har fylt ut, bruker ikke dere spesielt mye av ressursene til vedlikehold. Fint om du kan si litt om dette.

Nei, og det tror jeg nok er riktig. For vi har hatt en svær gedigen dinosaur applikasjon, og den er på en måte kontrollerbar. Den har ikke noe særlig grensesnitt, den har veldig lite nymotens trafikk eller webservice eller sånne ting. Alt er veldig hardt, SQL rett i basen, og på den måten der. Og det er ryddig, dersom man har dyktige erfarne utviklere, og da blir det ikke så mye

vedlikehold. Og det er klart at problemstillingen nå fremover, er at ufattelig mange flere mennesker skal snakke sammen, og jobbe med de samme tingene. Og dette kan nok føre til at mer tid går med til vedlikehold. Hvis man har 60% vedlikehold, så tipper jeg 2/3 går med til koordinering. Ikke sant, for alle på kjenne til alt. Jo større og mer tjenesteorientert dette her blir, mer komplisert blir det. Også må man kanskje ha sterkere og bedre verktøy for å holde orden på det. Men kompleksiteten øker, og da øker også det generelle vedlikeholdet. Jeg tenker på alle de møtene vi har… men vi må jo ha det! Før så satt jo en databaseansvarlig, og utviklerne over. Og de snakket fort sammen, og det gikk greit. Nå er det så mye forskjellige språk og teknologier, og plutselig er definisjonen forskjellig. Uansett hvor gode verktøy vi har for å generere skjemaer automatisk, så blir det alltid feil. Det er noe med det, det involverer så fryktelig mange flere mennesker. Men så tror jeg samtidig vi må lære oss det. Vi er enda i den spede barndom. Men det som er da, hvis du skal lage en søknad til kunden. Så vet du at et eller annet sted i systemet kan du få tak i kontoen til kunden, og det kan gjøre søknadsprosessen enklere for kunden. Så vil du jo gjøre det! Og alle er ivrig på det, også blir det mer og mer, og det genereres opp veldig kompliserte systemer. Men samtidig har du ikke noe valg. Skal du hive deg på IT, og ha den servicen kunden fortjener, så må du jo legge deg på det nivået.

Og hva opplevdes som de største problemene med det forrige systemet?

Det var faktisk databaseutvidelser. Det var krevende. Det er fordi den databasen vi har nå, det er en nettverksdatabase. Det er ufattelige kjappe databaser, men veldig tungvindt og vedlikeholde. Hvis du utvider med et felt, må det jo genereres i dager. Så utvide maskineriet er veldig tungt. Ellers rensker vi logger, og følger opp avvik som dukker opp. Men det er ikke så veldig mye, det er ikke det. Men et problem er at kunnskapen om de gamle systemene sitter i hodet på folk, og så blir de borte. Og det er vanskelig. Men nå skal alle de gamle systemene bort, og erstattes hele veien.