NTNU
Innovation and Creativity

Component Based System Development in the Norwegian Software Industry

Marius Sommerseth

Master of Science in Computer Science
Submission date: June 2006
Supervisor: Reidar Conradi, IDI

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

Problem Description

This Master thesis will try to identify differences and similarities in the utilization of reuse and product families in Norwegian software companies, as well as what are seen as important success factors for getting this to work. This will be done by executing a survey in companies that create and sell software, companies that create software for internal use, companies that create hardware and software for it, and consulting companies. Differences and similarities in companies of different sizes will be explored. Differences between how this is used internal in departments against whole companies will also be investigated. Success factors that are discovered will be discussed against success factors from older research.

The results from the IKT-Norge survey "Programvarebransjen 2005" and the results from the depth study "Reuse through Product-Families and Frameworks" by Marius Sommerseth will also be used to accomplish this.

Assignment given: 20. January 2006
Supervisor: Reidar Conradi, IDI

Abstract

Today it has become common practice to apply systematic reuse during software development. By reuse, the gain from creating a piece of software can be multiplied, as instead of creating a new component each time, old ones can be reused. This increases productivity (shorter time-to-market, less cost) and also software quality, as the components get well tested through using them in different systems. There are, however, many ways of applying reuse.

There are different types of components that can be applied in systematic reuse. The most common ones are internally developed, OSS, COTS, or outsourced components. There are also many different ways to share and access the components among the developers. Today all companies who apply reuse have some sort of distributed way of sharing.

To use product families is also one way of applying reuse. This can take reuse to another level as the reused parts can be vast, but it can also be used for branding a line of products.

The main part of this thesis is a quantitative survey that was executed with a questionnaire. 32 Norwegian software companies participated in the survey. The questionnaire asked about who applied reuse and product families, how they applied it, and what the respondents thought were important when applying it. The data collected is used to answer 3 research questions and are also discussed against related research. The data is also used to see if there are any differences between how reuse is applied in companies of different sizes and internally in departments as well as for whole companies. Also the impact of different program languages and development processes/methods on reuse is explored. This survey builds upon the pre-study “Reuse through product-families and framework” [MS00]. In the pre-study subjects from 12 Norwegian software development companies were interviewed about how they utilized reuse and product families. This was a qualitative survey with open questions, which was used to discover trends in Norwegian software development companies, and these trends are in this thesis examined.

The data from another survey done by IKT-Norge is also used in this thesis, but only the questions added extra for NTNU. These were about process improvement as well as reuse. There were a total of 142 Norwegian companies that responded, and 60 who answered the extra questions. The IKT-Norge survey is also compared against the thesis survey.

Preface

This report is a Master thesis written in the 10th semester of a Master study at Norwegian University of Science and Technology NTNU. This thesis builds upon the Depth study "Reuse through product families and framework" [MS00] written as part of the Master study.

This study is loosely connected to the Family project[1] which is a project for studying Norwegian companies and how they take advantage of product families in their development process.

I would take this opportunity to thank my supervisor, Professor Reidar Conradi for all of his help through this entire thesis, and also Bård Kroghus at IKT-Norge for his help and data from the IKT-Norge survey. I would also like to thank Ph.D. student Magne Syrstad, Ph.D. student Finn Olav Bjørnson and Ph.D. Carl-Fredrik Sørensen for their help with testing the survey, as well as Ph.D. student Odd Petter Slyngstad for providing me with the list of company contacts.

Trondheim, June 2006

Marius Sommerseth

[1] http://www.ikt-norge.no/templates/Project.aspx?id=583

Contents

List of figures

List of tables

1 Introduction

In this chapter the motivation for and the goals of the thesis are described. Also the outline of the report is presented.

1.1 Motivation

This thesis builds upon the depth study “Reuse through product-families and frameworks” [MS05]. In this depth study some ideas for further research were suggested. Some of these are looked into in this thesis as well as some other interesting points from literature. These ideas were on what success factors were important and if there were any differences between how reuse and product families were used in different types of companies.

Also some articles mentions success and failure factors that would be interesting to see if holds. The articles are somewhat old and also not only for Norwegian companies. It can therefore be interesting to examine if there are differences between these and the information that will be found in this thesis.

1.2 Goal of the Thesis

It would be interesting to investigate how reuse and product families were applied in Norwegian software development companies. Not only the points mentioned in the depth study, but also which success factors were seen as important and not important by the companies, as well as the reasons why they applied reuse and product families. The depth study only investigated trends, it would therefore be interesting to try and acquire some statistical significant data that could be concluded on.

To map differences between how reuse and product families were done in different companies would also be interesting. This means to see if companies of different sizes applied reuse and product families in the same way, and also to see if it was done differently internal in departments and for whole companies.

With this thesis I wish to answer the question: **“How do Norwegian software companies utilize component based development, reuse, and family lines in their development processes?”** I wish to find similarities and differences between the way they apply systematic reuse, and identify success factors and critical factors.

1.3 Scope of the Thesis

As this study only is interested in how Norwegian companies apply reuse and product families only Norwegian companies will be contacted. Companies with different business domains will however be contacted. The only criteria is that the companies are into some kind of software development and isn’t only a service company. The different domains that companies will be contacted from are:

- Companies who develop software and sell it.
- Companies who develop software for internal use in the company.

- Companies who develop hardware, but also software/firmware for the hardware.
- Consulting companies.

1.4 Report outline

The thesis has been organized into chapters as follows:

In chapter 2 State-of-the art a brief explanation of how software engineering is done today is given as well as a state-of-the art for systematic reuse and product families.

Chapter 3 Research Design explains how the research is to be conducted, and explains the different surveys that are to be used.

In chapter 4 Results and Main Findings the results and the main findings for the two studies are presented.

In chapter 5 Discussion the two surveys are discussed. Also the research questions are answered and the thesis survey is discussed against related research. The threats to validity are also discussed in this chapter.

In chapter 6 Conclusion and Further Work the conclusions that can be made are presented. Also further work is outlined.

In the Bibliography chapter the references used in the thesis are presented.

In the Glossary chapter some words that are used in the thesis are explained.

In Appendix A: The thesis questionnaire the thesis questionnaire as well as the invitation letter is presented.

In Appendix B: The IKT-Norge questionnaire a print out of the web-based IKT-Norge questionnaire is presented.

In Appendix C: Data from the Thesis Survey the data collected in the thesis survey is presented.

In Appendix D: Data from the IKT-Norge Survey the data collected in the IKT-Norge survey is presented.

2 State-of-the art

In this chapter a brief explanation of software engineering is given and an explanation of component based software development, reuse and product families. Also older studies with success and failure factors are presented.

2.1 Software Engineering Today

Wikipedia[2] defines Software Engineering as “the profession that creates and maintains software applications by applying technologies and practices from computer science, project management, engineering, application domains, and other fields”. This means it is a joint effort of many different practices. The term Software Engineering was first used at 1968 NATO Software Engineering Conference in Garmisch, Germany by the chairman Friedrich Ludwig Bauer. The idea was to create an engineering discipline for the creation of software [WRH00].

The reason why software engineering is needed is that in the earlier years of computers the creation of software was a messy and unstructured process. There was a lot of extra work being done because there was no higher overall plan to make the whole process draw towards one common goal. The programs would often have a lot of errors and had to be thoroughly tested to create satisfying results, which was a time and money consuming effort since there was no easy way to do it. They also saw the new advantages that software and computers had and wanted to utilize them.

Software engineering has come a long way since the 1960s. Today the creation of software is a highly evolved and technical process. A lot of work is put in defining what the customer, or user, wants the software to do before it is created. Then a sturdy architecture is created so that the developers know what they are supposed to do, and the work can be shared between many developers. If the architecture is good the need for testing will be greatly reduced.

The process itself can be seen as in Figure 1. A product idea and resources are put into the software process and a software product is created. The figure is adopted from [WRH00].

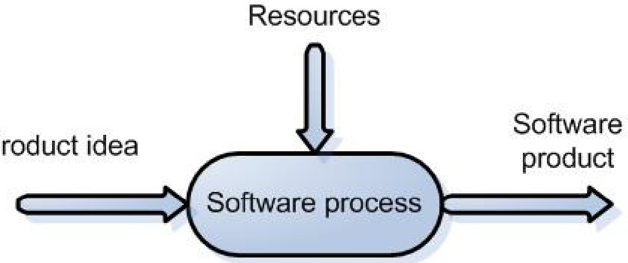


Figure 1: Software process[3]

[2] Wikipedia – The free Encyclopedia, http://www.wikipedia.org

[3] Figure adapted from [WRH00]

Software engineering is always changing and today many are driven more towards agile development. Agile development uses shorter time boxes as it is a well known fact that customers often change their mind during a development process. Therefore it is important that the developers quickly can change what they are doing, because if they cant the customers aren't happy, and the next time they will find someone else to do the job. With agile development they can.

By using frameworks and patterns that have been used before to create software the company know the quality the processes will have. There can be different patterns for different demands, such as high security, reliability etc. By applying reuse time can also be shaved of the development time, this is further discusses in the next section.

2.2 Component Based Software Engineering

Not many years ago software engineering was creating software from scratch to fill the user's needs and demands. This took a lot of time because the company developing the software had to have developers for all of the components of a program, and because nothing was reused every project was started with a clean sheet. An example is when creating an operating system the company would have to have developers to create interfaces to external devices, developers to create graphics, etc. This took a lot of time and a lot of money, and the developers would have to do things they hadn't specialized in, but because there were little or no competitors this wasn't a problem. But then again the software had lots of bugs, and took long time to create.

In today's software engineering market the need for getting as short as possible time-to-market (TTM) is an important factor. Developers are depended upon to create sturdy and flexible software in fractions of the time used before. "Being last-to-market spells sudden death in the software industry, and any gimmick that carves days or weeks from the development schedule decreases this possibility" [JV98]. This and the need for tailoring programs to the specific needs of individual customers have created the need for a new kind of development strategy, creating software from parts acquired from different sources. This is so called Component-Based Software Engineering (CBSE) [MCK04] or Software componentry. The idea is that software is created from smaller software components that are put together to form a bigger program. A software component is some software functionality that has been encapsulated. The idea of software being put together from pre-made components was first published in the report from the 1968 Software engineering conference sponsored by the NATO Science Committee. Here M.D. McIlroy presented his thoughts in his address "Mass produced software components" [NATO68]. McIlroy saw the need for modernizing software development, and thought that software created from components and put in catalogues was the way to go. He wanted to industrialize software development.

The idea of using components to put together products is not a new one. The American system of manufacturing developed by Eli Whitney in 1799 uses templates to make standardized interchangeable parts, or components. This was used to create muskets, and later used by Ford to revolutionize the car market.

The idea was that the parts could be created by any developer and put together to make a whole.

2.2.1 Systematic Reuse

Another advantage of CBSE is that the components can be used, and reused in many projects. Wikipedia[4] defines reuse as “the putting of an item to another use after its original function has been fulfilled”. Instead of having to get new components, or create new ones, it is possible to reuse parts of software that has been created earlier for other systems. This doesn’t just apply to code components, but also to documentation, architectural design, test plans, process models and other parts of a project. With having generalized components that can be used in all the steps of the projects the savings can be tremendous. Only by not having to create the specification documents from scratch, but instead from a template will save hours on a project. And with the use of frameworks, that can describe the whole process, weeks can be saved on a development project.

The components used in CBSE can be either COTS (commercial-off-the-shelf) OSS (open source software), components that are outsourced, or components created by the company itself. What they all have in common is that they all have to be generalized and saved for efficient reuse.

2.2.2 COTS

COTS is computer software that is made by a by a vendor and sold to other companies for use in their programs. The components are made generally and not for a specific company or task, and are therefore off-the-shelves. They aren’t customized for the individual companies needs. Using COTS a company can buy ready made program components that can be incorporated into bigger programs. The benefits with using COTS are many, among them that “the functionality can be accessed immediately, it can be obtained at a significantly lower price, and developed by someone who is an expert in that functionality” [JV98]. Therefore the price of developing a program, and also the TTM can be minimized, and the quality can be improved.

The study [LCS05a] concludes that the main reasons for using COTS in commercial projects is that its users believe that paid software follow market trends and that the vendors will provide good technical support.

Even though the components most of the time come finished off-the-shelf, a recent study [LCS05b] shows that companies that are going to buy the components often influence the vendors in different ways to get them to create what they want. This can be by buying the company or giving extra money.

The primary problem with COTS is that the company purchases a complete component that most of the time comes without source-code. If the requirements for the system is changed (which often happens in software development especially if the customer doesn’t completely know what they wants) often the component has to be changed to. Then the company has to contact the vendor

[4] Wikipedia – The free Encyclopedia, http://www.wikipedia.org

from which it purchased the component and get them to change it if possible, if not they will have to buy a new component that does what is required. This takes a lot of time and therefore a lot of money. Another problem stated in [JV98] is that the functionality provided by a COTS component can have more functions than what is needed by it, and this again requiring the creation of software utilities for restriction of the components functions. Another risk is that the vendor which created the component goes bankrupt or stops creating components. This would be a serious problem if something needed to be changed, and also gets in the way of upgrading the software as there won't be created newer versions of the component. Also unless the company which the component is bought from has been used before there is no way of telling what kind of quality the component will have. Problems can surface after any period of time.

A study [LCS05a] revealed that COTS users also have higher risk on estimating the effort into selecting. They also have problems following requirement changes and controlling the negative affects of security (because they can't see how the components is built, they don't know if there are any threats to security).

There are also other types of OTS (off-the-shelf) components used today. The most used are MOTS (modifiable-off-the-shelf), GOTS (government-off-the-shelf), and NOTS (NATO-off-the-shelf). MOTS is a COTS supplied with source code, so it can be modified by the consumer. This is almost as a bought OSS. GOTS are components developed by, and distributed by the government for use in government application. Both GOTS and NOTS, which is the same as GOTS but developed by NATO, can be used for creating programs for the government and NATO. An example is a security component that has to be incorporated in programs for communicating with government servers. Also by demanding the use of such components governments and NATO have more control of what a program contains, and that it follows given standards.

2.2.3 OSS

OSS is computer software that follows the Open Source Definition[5]. Open Source Software is software that is free to use, and free to develop and use in new software. The idea was proposed by Richard M. Stallman in 1984 when he created the Free Software Foundation (FSF)[6]. His though was that "Free software is a matter of liberty not price". FSFs thoughts on free software are that "the user should have the freedom to run, copy, distribute, study, change and improve the software"[7]. This doesn't mean that the software should be non-commercial, but if you buy it you can do whatever you want with it. Even modify it and sell it to other users. Even though OSS is built upon an idea proposed by Richard M. Stallman it was Bruce Perens and Eric Raymond that founded the open source movement in 1998 [HK03]. Much of the principals are the same, but the differences are as stated in [HK03] "It differs from that movement primarily on philosophical grounds, preferring to emphasize the practical benefits of such licensing practices over issues regarding the moral rightness and importance of granting users the freedoms offered by both free and open source software" (by

[5] Open Source definition http://www.opensource.org/docs/definition.php
[6] The Free Software Foundation - http://www.fsf.org/
[7] FSF – The Free Software definition - http://www.fsf.org/licensing/essays/free-sw.html

the movement the Free Software Foundation is meant). Bruce Perens and Eric Raymond felt that Richard M. Stallman had a good idea, but his ideologies were wrong.

The software is therefore distributed with its entire source code, but there is nothing that prevents the developer from selling it as part of a program. Open Source Software applies licenses to the program for its protection. There are many types of licenses[8] where the most widely used is GPL (GNU General Public License[9]) and Open Software License[10]. Licenses are inherited meaning that if a part of a program licensed under a license is used in another program, that program also has to be under the same license. This is sometimes a problem for a commercial company, because if they use a GPL licensed code in a program, the whole program will have to be GPL licensed, and therefore anyone wanting to use the source code can do so. In this way the source code can be stolen and used in other programs, and weaknesses in programs are easily detected.

OSS also has a lot of strengths. First of all, it's free to acquire. Second being that it is distributed with source code so if there is need for changing the components this isn't a problem. This also makes it possible to upgrade products and create new and better versions. By opening the source code and the programs for many users a lot of persons are involved in developing and testing the software. This makes for better software as the total effort and expertise put into it is vast. This can be seen as peer review [LCS05a] [BL01].

The study [LCS05a] concludes that the main reason why OSS is selected in commercial projects is that the code can be acquired for free, and also to avoid possible vendor support risks.

The advantages are extremely good, but there are also down-sides. Even though OSS components are free to acquire they also need a lot of time and effort to do what they are supposed to do. While COTS comes ready to perform a specified task OSS rarely comes "good to go" and often without documentation and an easy way to configure and manage it. This means that there has to be put down a lot of effort from the developers to first understand the source code and then change it to do what it is supposed to do. Therefore the initial cost can often be greater than for COTS-based software. But in the long run OSS isn't more expensive than COTS solutions [PDG05].

[BL01] argues that OSS has a resemblance of the academic community. Also here publications are given freely to whoever wants to read and use them as references. But used in another report it has to be stated clearly where it has been taken from. This isn't done for money, but for recognition in the community. The same practice can be seen with OSS, where people aren't after the money, but after the recognition and sometimes also the possibility of being contacted by companies with job offers. [BR03] and [HK03] argues that OSS has evolved out of the academic community.

[8] Open Source licenses http://www.opensource.org/licenses/
[9] GNU General Public License http://www.opensource.org/licenses/gpl-license.php
[10] Open software License http://www.opensource.org/licenses/osl-2.1.php

OSS is based on a gift giving culture [BL01]. Therefore it is important for companies that use OSS to give something back. This means helping with projects, giving financial support, and giving back updated components. There are a lot of savings for companies from using OSS in their projects, and therefore it is peril to give back. The companies don't want to be seen as freeloaders.

2.2.4 Outsourced Components

While COTS components are finished components that are sold as they are, the whole component can also be outsourced. This means that another company that have specialized in making a specific type of components can be used, and therefore the quality of the components acquired this way are high. While this is a better solution than COTS because it comes ready to plug into the system, it is also a much more expensive one. But the benefits are good, as the components are good to go, the support is good, and it is created for a specific task. If it needs to be altered the company that created the component can also be contacted for this.

Another reason why outsourcing of components is used is to outsource work. Instead of having to hire extra manpower for a big project outsourcing can be used to get what is wanted without having to hire more people, because if these people aren't needed later they will have to be fired. It can therefore be seen as short term manpower.

The down sides with outsourced components are much the same as those with COTS. If the company which the component is bought from stop creating it or goes bankrupt this can become a problem as there is no way to change it. However it could be delivered with the source code and this will then eliminate this problem.

2.2.5 Internally Developed

While COTS, OSS and outsourced components are so-called "Third party software components" components can also be created internally in the company. Internally developed components are the components that usually need most work to be able to be reused in other projects. This is because COTS often come generalized, OSS has to be generalized before it is used, and outsourced components come as components that can be "plugged into" the software. Therefore it is important to set aside some time in projects to make the components ready for reuse, describe it and save it somewhere for reuse. Even though this can add some extra costs to a project it will benefit the company in the long run.

2.2.6 Technologies for Sharing of Components

A reusable asset (or component) needs to be accessible for the developers, if not no one will use them. They also need to have high quality, and to be well documented so that developers can find them and use them without having to think about them breaking down and creating problems. This is important for parts like documents, but it is crucial for architecture and code components that are to be reused.

There are many different ways of sharing the components between the developers. Today this is mostly done in some electronical way, as having it on paper will mean that for example code components have to be typed in manually and there isn't any good way to search through them. Earlier it was common practice, at least for forms, to have empty templates that could be copied and filled in. Also a distributed system wasn't usual, and the developers mostly had their own libraries on their computers which only they could reuse.

The common practice for sharing the components today is to save them in a shared library/database that can be accessed by developers to find components, and to save new ones. This can be done in many ways; from as easy as sharing a folder, to as complicated as a system for rating and describing components. The problem with having this type of library however is the scalability. If the library is too small, no one will use it as they know the chance that what they are looking for is in the library is small. If the library is too big no one will use it either because the chance of finding what they are looking for (even if they know it is in there) is small. [CHW97] defines acquisition, classification, representation and retrieval as the complexity issues for engineering large-scale software reuse. To get the savings that are possible from software reuse it is therefore crucial to have a system that works and that developers use. If not it will become outdated and die out. There also needs to be a good way to search and find the components needed.

Today it is common practice to have some sort of distributed, computer based system which has different degrees of functions regarding on which type of system it is. The most common ones are:

Configuration system (Concurrent Versions System)
This is a client server system that keeps track of components that are updated in the system, including keeping systems updated with the right versions of components. The project is stored on a server, and then several developers can connect to the server, check out a copy and work on it, and then later check in their modifications. If two developers work concurrently and try to check in changes on the same files the client is made aware of the conflict, if there isn't a conflict the version number of all files involved are incremented. The server also keep a log of all changes done, and it is therefore easy to go back in time and see who and what was changed.

Database dedicated to reuse
Another way to share components between users is to have an own database created especially for sharing of components. A database has built-in functions for searching, and this is one of the reasons why it is a much used way for sharing components. Also it can be created with own fields for context, version number etc. All of the users have access to search the database for components they need, and to upload new components. The problem with using a database is that the version controlling has to be done manually. Which persons should have access to update components also has to be set manually. This is a simpler and more manual system than a configuration system. But it is very easy to use and set up, and is also cheap.

Intranet

Intranet is a “private internet” that uses web protocols to connect together computers, and has a web portal that offers different services to its users. It uses some kind of access control to make it secure and to restrict access only to the users intended to have access to it. An intranet can be used for various purposes, for example for keeping track of the users schedules, sending out messages, keeping track of contacts, or as thought of here to share components and files. Many intranets have functions for uploading files and commenting on them, and this makes it possible to share reusable components. By using this it is easy to access the components for all of the developers and it is also easy to upload new components for sharing as the system is already known. The problem however is that it isn’t made for this purpose, and when a component is rewritten it will have to be uploaded as a new one. There is no version controlling, and there doesn’t have to be any built-in search functions either, so finding what you are looking for could take long time if the library is big and the descriptions are bad. It is therefore very much a manual way of sharing.

Shared folders

The easiest way of sharing components is probably to have some shared folders on a file server that all the users can access. The different types of components can be put into different folders. This is the easiest and cheapest way of sharing as it only requires a network (which all software development companies have today), and no extra programs or configuration. But this is also the most manual way of sharing. The components will only be saved as files, and they will have to have extra files with context. The operating systems built-in search functions can be used to search these, but this is a difficult way to do it. There is also no way of having version controlling and this can lead to many different versions. If the library grows to some size it will be impossible to find what one is looking for.

2.2.7 Problems with Component-based Software Development

There are a lot of problems that have to be addressed when using component based software development, many of them have already been addressed in the previous sub-chapter.

Another problem is that code components have to be generalized so that they can be used in different systems. It takes some extra resources to generalize a code component before it is put into the library. There are many ways to do this today, either the components can be developed with this in mind, or they can be generalized at the end of each project.

As mentioned COTS components often come without source code, and even if it comes with source code it is rarely changed. Also OSS is often treated as closed source, because the resources needed to understand and change it can be tremendous. A study [LCS05a] discovered that they both are treated much alike. Therefore glueware, or proxies, is often created. This is software that stands between components and translates so that they can interact without changing the component itself. The same study [LCS05a] however discovered that new development tools like .net automatically creates this glueware in many projects today. Therefore addware (software to add missing functionality) is created more often than glueware.

2.3 Product Families

SEI (The Software Engineering Institute)[11] defines a product family as; "a set of software-intensive systems that share a common, managed set of feature that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way"[12]. A core asset is a substantial part of software that is used in more than one of the products in a product family, and therefore reused. Core assets can't just be software components, but also be the architecture of a system, or the documentation. Product-lines are also a commonly used name on product families. In this report the name product family will be used.

The book Software Architecture in Practice [LCK03] defines the following as possible reusable assets:

- Requirements analysis, because many requirements can be the same in different products.
- The architectural design of systems.
- Software elements or components.
- The models and the analysis of the project.
- Test plans, test cases, test data and how to fix errors.
- The planning of the project (budgets and schedules).
- Processes, methods and tools.
- The people working on one product in a family can also work on other products.
- Prototypes of systems.
- Defect elimination meaning that defects in the first product will not be in the second.

The parts of a project that are reusable are therefore many. This doesn't just save time and money, but also creates better products because the company knows what they are doing. This is a practice used by many companies today to produce families of similar systems that have one common part, and one part that is specific for each system.

There is also another reason for using product families which is called branding (or family branding). Wikipedia[13] defines family brandings as "a marketing strategy that involves selling several related products under one brand name". This doesn't require sharing any components, but instead marketing many products under the same family name. The idea with branding is that when a company has purchased one product in a family they will often tend to buy other product from the same family if needed. If they are satisfied with one product they know that they probably will be satisfied with the others. This is something

[11] The Software Engineering Institute - http://www.sei.cmu.edu/

[12] SEI – About software product families, http://www.sei.cmu.edu/productlines/about_pl.html

[13] Wikipedia – The free Encyclopedia, http://www.wikipedia.org

that shouldn't be taken on lightly, as having one product in the family that isn't the same quality as the others can ruin the whole product family name. Therefore this shouldn't be used if the products in the family aren't of high quality.

The methods when dealing with product families are different than those of ordinary software engineering. [MC03] is a report of a study done in the company Ericsson[14] on the methods they used when they created two telecommunication systems based on reusing the same software architecture, software process, and many other core assets.

The span of a product family is therefore far greater than that off reuse with component based software development. The core assets are saved in a core assets base [LCK03] which then can be accessed to reuse the assets. This saves time and money because the asset doesn't have to be created more than one time. A lot of companies have shown great increase in productivity and decrease in cost by creating product families. There are many examples of how well this works. Nokia creates 25-30 new models of mobile phones every year, before they only created 4, this increase is because of their product family approach [LCK03].

When having a product family there are different ways that the products varies from each other. These are:

- Parameter adoption. With parameter adoption there is one main product that is configured in different ways by setting parameters in different ways.
- Addware. With addware there is a core that is the same for all the products in the family and extra modules and software are added to get different functionality.
- Common framework/architecture, adapted modules. In this type there is a framework or an architecture that is the same for all of the products in the family. However the modules that are used to put the products together are different.
- Common modules, adapted framework/architecture. Here the products are put together from the same modules, however the framework and architecture that is used, and how the modules are put together is different.

2.3.1 Challenges with Product Families

Even though the advantages are many this isn't a strategy that can be taken on lightly. The biggest challenge is getting the scope of the product family right. The scope is the description of which products will be in a product family and which won't. If the scope is too big the products in the family can be too different and the core part that is shared will be small or non existent, and if it is too small the products that can be placed in it will be too few to utilize the full potential. The scope also influences the core assets base. If the scope is big the core assets base will also be big. This means that a developer will have to search through a lot of assets to find what he is looking for, and it could just not be worth the effort (maybe it is easier to create a new component). But then if the scope is small the

[14] Ericsson - http://www.ericsson.com

core assets base also will be small and the chance for a developer finding what he is looking for decreases.

For instance a navigation system for a boat will work with one type of components (engine, GPS, radar), but will it work with other components? Len Bass, Paul Clements, and Rick Kazman define the scope of a product family as a doughnut [LCK03]. Where the area around the doughnut is outside the scope of the product family, the area on the doughnut can be handled in the product family with some effort, and the area inside the doughnut is what falls inside the product family. The latter is what should be created by a company. The scope of a product family will evolve as the product family is changed[15] as seen in Figure 2.

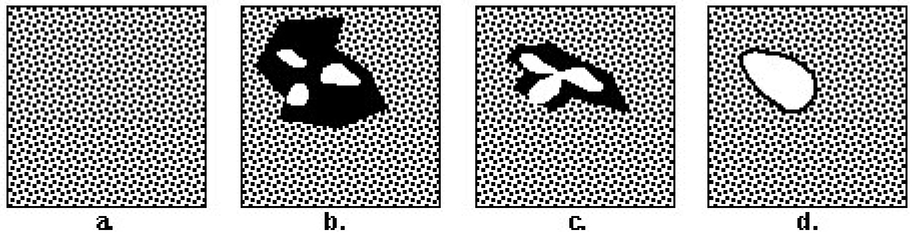


Figure 2: The Evolution of a Product family Scope[15]

In the figure what is inside the scope is colored white, what can be inside with a little effort is colored black, and what is outside is marked with dots.

When there exist no product family (Figure 2a) everything is outside the scope. Then a first product is created in the family. Now what can be inside the family with some efforts is big (black), and what is inside is smaller (white) (Figure 2b). As the product family evolves the part that can be inside the scope gets narrower and the part that is inside the scope gets bigger (Figure 2c and d). This means that the scope gets specialized.

There are many approaches for scoping a product family, but one aspect that always should be kept in mind is that not only the system which is being created should be taken into concern. What kind of customers the company has should also be used when creating the family. An example is that private users often are looking for different things in their products than companies. Therefore if a company has both as customers it could have a different product family for each of them.

[LCK03] defines two strategies for creating a new product family; proactive and reactive. There is also a third one which isn't mentioned, this is incremental. This can be seen as something in between of proactive and reactive.

In a proactive approach the scope of the family and the core assets base are created before the first program in the family. The company does this by "taking advantage of their experience in the application area, their knowledge about the

[15] Taken from What's the difference Between Product family Scope and Product family Requirements?, Paul Clements, http://www.sei.cmu.edu/news-at-sei/columns/software-product families/2003/2q03/software-product families-2q03.htm

market and technology trends, and their good business sense" [LCK03]. They therefore use previous experiences to create the family. By using the proactive approach the company is able to foresee how the family will become and therefore "take charge of its own fate" [LCK03]. This is maybe the most expensive way of starting a family, as it is started from scratch and therefore nothing is reused. However all the products are created the same way, which can make for a better family.

In a reactive approach the family is built up of one or more existing products. This is usually done because the company has problems with forecasting how the market will become, or they may not have the resources of creating a core assets base before creating products. The strategies, the scope and also the core assets are "built up from what has turned out to be common – instead of what was preplanned to be common" [LCK03]. This is a cheaper strategy. Ralph Johnson and Brian Foot emphasize that reusable artifacts are not designed, they are gradually discovered [JF98]. This means that because the reactive approach is not as much "forced upon" as the proactive it is what gives the best results. The disadvantage with this type of starting a family is that the old systems which the family is based upon will maybe have to be redone. Especially if it is built upon many systems, they will then have to be redone to be a family.

The last strategy, which isn't mentioned in [LCK03], is that of incremental product family. This is when the product family is started during the creation of a product. This can be done if the potential for a possible product family is observed while creating a product. This has some of the advantages from both proactive and reactive families, as there already are some components that can be used to create the core library, and since the product isn't finished the work to create it as part of a family will be small. The negatives for this strategy are the same as for proactive and reactive, but not as bad as for them.

2.4 Possible Success factors

The article **"Sixteen Questions About Software Reuse"** by William B. Frakes and Christopher J. Fox [FF95] presents a survey of 113 people from 28 U.S. and one European organizations about what they think is essential for software reuse to work. The survey was performed in 91-92.

Through the survey they identified which factors affected reuse and which didn't.

The factors that the subjects in the survey meant gave better reuse were:

- Type of industry.
- Perceived economic feasibility.
- High quality assets.
- Common software process.
- Reuse education.

The factors that the subjects in the survey looked upon as didn't help to create better reuse were:

- Programming language.
- CASE.

- Experience.
- Recognition/Awards.
- Legal problems.
- Repositories.
- Organization size.
- Quality concerns.
- Reuse measurement.
- NIH (Not-invented-here).

In the article **“Success and Failure Factors in Software Reuse”** by Maurizio Morisio, Michel Ezran, and Colin Tully [MET02] 32 reuse projects funded by The European Commission are studied to find out if there are any relations that can explain why some of the projects fail to get reuse to work while others succeed. The 32 project are selected out of a total of 288 Process Improvement Experiments (PIEs) and are the ones judged to really be dealing with reuse. They tried to identify failure and success factors in the different projects.

The success factors the researchers discovered were:

- Smaller companies had the advantage of easier communication.
- Products that were in a product family succeeded more than those that was isolated.
- Roles dedicated to reuse were necessary.

The factors for failures were many:

- A repository doesn’t automatically mean successful reuse.
- Didn’t modify non-reuse processes.
- No deep management commitment.
- No production of assets.
- No training/awareness action.
- Reusable assets produced but not used.

The article concludes therefore what is important to do and not do to get reuse to work.

3 Research Design

In this chapter the research conducted in the thesis is outlined. The research questions that are to be answered are presented, as well as what methods are used to acquire and test the data are described.

3.1 Background

As mentioned in the introduction it would be interesting to further investigate some of the more interesting ideas presented in the depth study [MS05].

The points which were interesting were:

- **Rewards:** It has long been believed that rewards don't give better reuse, but it would be interesting to further investigate this.
- **Internal practice:** It would be interesting to see if internal practice for reuse by training and having a reuse system that works gives better reuse.
- **Small companies vs. large companies:** In the depth study it seems like reuse and product families works better in small companies than in larger.
- **Internal in departments vs. for whole companies:** Some of the subjects interviewed in the depth study felt that reuse worked better internal in their departments than for the whole company.

Also in the articles [FF95] and [MET02] some interesting success and non-success/failure factors are presented. These were presented in chapter 2.4 Possible Success factors, and some of these will be tested in the thesis.

3.2 Research Questions

As mentioned earlier the question; "How do Norwegian software companies utilize component based development, reuse, and family lines in their development processes?" will be tried answered in this thesis.

I will try to identify similarities between how the different companies utilize reuse and product families, what they look upon as crucial factors and why they apply reuse and product families. In which companies reuse is successful and which it isn't will also be looked at. Also the differences between companies of different sizes, and internally for departments in companies with departments and for the whole company will be investigated.

To answer this, the following research questions have been defined:

RQ1: What is the difference between how reuse is done in small, medium and large companies?

RQ2: What is the difference between how reuse is done internally in a department and for the whole company in bigger companies?

RQ3: What is the difference between how product families are used in small, medium and large companies?

3.3 Surveys

In [WRH00] it is written that a survey "could be seen as a snapshot of the situation to capture the current status". A survey is conducted when "the use of a technique or tool already has taken place or before it is introduced". Therefore it can be conducted after something has been done to capture what has been done, for example to study how a new process has influenced something, or in my case to study how the companies see the utilization of reuse and product families in their companies.

There are mainly two types of surveys; qualitative surveys and quantitative surveys. Qualitative surveys focuses on how an individual or a group of individuals view and understand the world. It tries to identify how the subjects think their surroundings work. Quantitative surveys on the other hand try to get statistical valid samples. Therefore they are created from already known views of how something works and doesn't try to identify new ones. It is easier to do analysis and statistical tests on this kind of data. [WRH00] states this about quantitative research: "The aim is to identify cause effect relationship."

In a survey a sample is selected from the entire population and this sample is studied to collect the information needed for the research. By doing a survey on a sample of the population it is possible to understand the population from which the sample was drawn. By doing this it is possible to take, for example, answers from 50 out of 500 companies and then understand how it is done for all 500. It is though important that the sample is randomized, and that the companies are selected from all of the different types of companies there is so no types are left out of the sample.

According to [WRH00] there are 3 general objectives for conducting a survey. These are:

- **Descriptive survey.** A descriptive survey is conducted not to find out why an observed distribution exists, but rather what the distribution is.
- **Explanatory survey.** An explanatory survey is conducted to make explanatory claims about the population.
- **Exploratory survey.** An exploratory survey is conducted as a pre-study to a more thorough investigation.

When conducting a survey first the information is collected and then it is arranged into a form that can be handled to find tendencies. The information can be collected either through interviews or questionnaires. Questionnaires can be provided as a form either on paper or electronically. A questionnaire is usually sent to the subjects with instructions on how to fill it out. The subject then answers it and returns it to the researcher. With interviews the researcher (or some other person) does face-to-face or phone interviews with the subject. With interviews the response ratio is typically higher, and the error rate in filling in is smaller. It is also possible to get additional information. Even though interviews have a lot of advantages it takes a lot of extra time and resources to extract the information.

3.4 The Thesis Survey

To answer the research questions a descriptive survey would be conducted. A descriptive survey is a survey "conducted to enable assertions about some population" [WRH00]. It was decided to use a questionnaire to gather the information for the survey. The reason this was selected is that the survey would be a quantitative one and therefore the questionnaire wouldn't have open questions. There is therefore less chance for misunderstanding. Another reason is that it takes less time to fill in a questionnaire than to submit to an interview. Therefore a questionnaire would be used and the subjects would be phoned to get them to submit their answer.

3.5 The Thesis Questionnaire

One important part of a survey is the construction of the questionnaire. If the questionnaire has flaws the information extracted could be rendered useless and the survey would fail. Therefore the questionnaire has to be well constructed and tested before it is used, because if not it could be misinterpreted. To create the questionnaire the information collected in my depth study [MS05] and a survey conducted by IKT-Norge[16] were used as a basis for formulating the questions. It was a long and time consuming process to make a good questionnaire.

3.5.1 Creating the Questionnaire

When creating the questionnaire, first what information was needed had to be determined, then the questions had to be created with this as a basis. The questionnaires main goal was to answer the research questions. Also since it would be compared with the articles [FF95] and [MET02] they were used to define what information was needed. The information that was thought needed was:

- **Background information about the company.** Here information like numbers of employees, what the company did, and the names of the company and the subject were needed.
- **Technical information about the company.** This was needed to find out what the companies used to develop their software (development platforms, programming languages as well as methods/processes).
- **Reuse.** It would be needed to find out which of the companies applied reuse and for how long they had. Information about how well reuse worked, what the companies reused, and what systems they used to share it was also needed. Also, for comparisons, some way of rating what was important for the company regarding reuse was necessary to identify success and failure factors.
- **Product families.** Here it would be needed to find out which of the companies used product families and for how long they had used it. What was shared among the different families, what the differences between the

[16] IKT-Norge - http://www.ikt-norge.no/

products in the families were, and how product families were started was also interesting. The size of the families and why product families were used in the different companies would also be asked about.

- **Other.** To have an open question at the end that could catch other aspects that the subjects felt had been left out, but could be important for the study, was also needed.

Then the questionnaire was created using this as a basis. It was divided into four parts:

- General information about the company.
- Technical information about the company.
- Systematic reuse in the company.
- Product families in the company.

These would be used to answer the research questions as well as to make general conclusions about the companies. The thesis questionnaire can be seen in Appendix A.

What purposes the different questions were meant for are shown in Table 1. Here RQ1-RQ3 means the research questions. GI means general information, this means that the questions were meant to get general information about the companies and subjects like for example names. GC stands for general conclusions and these questions were meant to get data that could be analyzed to make general conclusions. Also the two articles [FF95] and [MET02] are shown in the table, and which questions were meant to compare with these can also be seen.

Table 1: Questions relating to information needed

Questions	RQ1	RQ2	RQ3	GI	GC	[FF95]	[MET02]
Part 1 - General information							
QT1				x			
QT2				x			
QT3				x			
QT4	x	x					
QT5				x			
QT6				x			
QT7				x			
Part 2 - Technical information							
QT8						x	
QT9						x	
QT10					x		
QT11					x		
Part 3 - Reuse							
QT12	x	x			x		
QT13	x	x			x		
QT14	x	x			x		
QT15		x			x		
QT16	x	x			x		
QT17	x	x			x		
QT18	x	x			x		
QT19	x	x			x		
QT20	x	x			x	x	x
Part 4 - Product families							
QT21			x		x		
QT22			x		x		
QT23			x		x		
QT24			x		x		
QT25			x		x		
QT26			x		x		
QT27			x		x		
QT28			x		x		
QT29			x		x		
QT30				x			

The questionnaire was created as a word file that could either be answered by marking in the file or printed and filled out on paper. To have the questionnaire as an electronic based system was also considered as this would make the answering and the registration easier, but because the large amount of time and effort needed to create a safe and good system the idea was rejected. Since only 32 answers were needed, registering these wouldn't take too much time.

Because it was my first time creating a questionnaire a lot of time was used understanding how to create the questions and a lot of people were asked for help. The first version had almost only open questions, with help from my teaching supervisor I soon realized that it would be hard to group together and make statistical analysis on the information if it was too scattered. I therefore tried to use closed, check off based questions as much as possible. This would make the statistical analysis easier, and also make it easier to make conclusions.

3.5.2 Testing the Questionnaire

When I was satisfied with the questionnaire it was first tested on Ph.D. student Magne Syrstad. As he had created questionnaires before he came with a lot of valuable inputs towards the questions I had, and also towards questions that I should have to get the information needed. The input was used to create a new questionnaire which was then tested on Ph.D. student Finn Olav Bjørnson and Ph.D. Carl-Fredrik Sørensen. They both had some inputs on the questions, and it was especially important to me to get input on questions that could be misunderstood, as this would lead to unusable data.

After the rounds of internal testing the questionnaire was tested on a few companies. It was sent to 5 companies where 3 of them replied by filling in the questionnaires. They also commented on questions and aspects they thought were unclear. Their responses were then used to create the final questionnaire.

The finished questionnaire can be seen in Appendix A.

3.6 The IKT-Norge Questionnaire

In this thesis some answers from a survey conducted by IKT-Norge in cooperation with NTNU is used. This was a survey conducted to map the software industry in Norway, and therefore only Norwegian companies participated. Some extra questions on quality systems, process enhancements and component based development were added for NTNU as an optional volunteer part.

The IKT-Norge survey was performed during spring 2006, but the questions asked about answers from 2004. The companies were contacted by e-mail, fax or letter and asked if they would submit to the survey. The survey was called the business report “Programvaresektoren 2005” (which means program sector 2005) and was answered electronically on a web page. About 1000 companies were initially contacted. Of these about 250 registered for the survey and were sent the survey and information, and then about 190 of these started the survey. 142 finished it, and 60 of these also answered the extra questions from NTNU.

The IKT-Norge questionnaire used can be seen in Appendix B. This is a copy of the information on the web page, and is therefore not an exact copy of what the subjects used. It does however have all the questions and information given. The data from the survey was than saved as an SPSS file, and also as an excel spreadsheet.

The data from the questionnaire can be seen in Appendix D.

The difference between the IKT-Norge survey and the thesis survey is the sampling of the subjects. In the IKT-Norge survey they were randomly selected. They were selected in three ways:

1. Picked from companies with NACE business code 7200 from the Brønnøysundsregistrene[17] (not all of them)

[17] Brønnøysundsregistrene - http://www.brreg.no

2. Hand picked over the last 1-5 years from profiled software activities in the press, the yellow pages etc.
3. Selected from the previous business reports.

3.7 Gathering Data

The companies used in the Thesis-survey were selected in the following way: First a list of companies compiled by InfoSector[18], which is a sub company of IKT-Norge, was acquired. The list was from a survey on COTS/OSS conducted spring 2004 by NTNU, and was acquired from Ph.D. student Odd Petter Slyngstad. This was a convenience sampling since there was no influence on what types of companies they were, what was available was taken. [WRH00] states that convenience sampling is when "the nearest and most convenient persons are selected as subjects".

The list consisted of companies in 3 different categories grouped by company sizes. These were 0-19 employees, 20-99 employees, and ≥100 employees. The companies used in the pre-stud [MS05] were also added as these could be used again. The data collection process consisted of:

1. Around 30 companies from each of the categories were selected and an enquiring e-mail was sent to them together with an invitation letter and the interview guide which can be seen in Appendix A. Because the list was old some of the e-mail addresses didn't work, and some of the people had switched companies. Therefore around 15 more subjects were sent e-mails (5 from each category). The process to get the answers was a long and time consuming one. While some of the subjects answered rapidly others had to forward the e-mails to other persons (in the larger companies), this took time and was sometimes frustrating. The total number of subjects the e-mail were sent to, how many got through and how many answered can be seen in Table 2.

2. After sending the e-mails I waited for one week before I started phoning the subjects to hear if they had received the e-mail, and if they would be interested in participating. A lot of time was used phoning, as many of the persons on the list were hard to get in contact with. Also many hadn't received the e-mail and were therefore sent new ones.

3. Most of the answers were filled into the word file and returned attached to an e-mail. Two of the answers were faxed, and one was sent by ordinary mail.

In Table 2 how many subjects were contacted, and how many answered can be seen. The table shows how many were initially sent out for all the groups, how many of these didn't get through (Not working), how many extra were sent out and how many of the extras that didn't get through (Not working (ii)). It also shows the total number that was contacted (how many got through), how many answered, and the percentage of contacted that answered.

[18] Infosector.net - http://www.infosector.net/

Table 2: Number of subjects

	1-19	20-99	≥100	Total
Initially sent out	32	34	29	95
Not working	3	4	2	9
Extra sent out	6	5	5	16
Not working (ii)	1	0	1	2
Total contacted	34	35	31	100
Answered	8	16	8	32
% Answered	24 %	46 %	26 %	32 %

3.8 Presentation of the Data

The answers from the subjects were plotted into an excel spreadsheet. This was done to make it easier to understand and interpret the data and run statistics on it. Descriptive statistics was then used on the data to describe and graphically present interesting aspects. [WRH00] states that “The goal of descriptive statistics is to get a feeling for how the data set is distributed”.

For measuring the data, tables are used, and for the graphical presentation histograms and pie charts are used. Pie charts are used where the answers are exclusive, this means that the subjects only could answer one alternative. Histograms are used where answers were non-exclusive. This means that the subjects could answer more than one alternative. Some of the exclusive answers are however presented in histograms. This is where the alternatives were so many that the pie chart became difficult to interpret.

The data is presented in different ways, but where it is possible the mean value and standard deviation (SD) is calculated. This is on the questions which had a nominal rating scale as answer. Also median, which is the answer in the middle, is presented where this is possible. In the tables and figures the number of subjects who answered is also given. This is either presented as N (number) where the subjects could only answer one alternative and NeN (non-exclusive number) where it was possible to answer more than one alternative. In some tables a percentage value is also given. Where this isn’t explained otherwise the percentage gives the percentage of the number of subjects who answered yes on the alternative.

To present a lot of the answers from the surveys both tables and histograms are used. These present much of the same data, but the reason why both are used is because the information is easier to interpret in a histogram, but because also the percentage value is needed to see the relations, and it sometimes isn’t possible to have the percentage value in the histogram, tables are also used.

3.9 Statistical Tests being used

To answer some of the research questions tests would have to be run on the results to find out if the data is enough to conclude from. The subjects were divided into three different groups by size. The companies who are divided into departments can be seen as two groups, one for the whole company, and one for the departments. When doing this kind of test a null hypothesis is first created.

Then a test is run to see if it can be rejected, if it can conclusions can be drawn, and an alternative hypothesis will be true. The tests that are going to be used are:

ANOVA (Analysis of variance) [WRH00].

When three or more different samples are going to be tested against each others ANOVA will be used. [DL00] says that "ANOVA separates the total variation in a set of measurements into a component due to random fluctuations and a component due to actual differences among the alternatives". In its simplest form (which is going to be used in this thesis) the test is used to test if a number of samples have the same mean value. The test is performed as showed in Table 3.

Table 3: Performing of a ANOVA test[19]

ANOVA, one factor, more than two treatments	
Input	A samples, $x_{11}, x_{12}, \ldots x_{1n1}$; $x_{21}, x_{22}, \ldots x_{2n2}$;…; $x_{a1}, x_{a2}, \ldots x_{ana}$
H_0	$\mu_x = \mu_y = \ldots = \mu_{xa}$, i.e. all expected means are equal.
Calculations	Calculate $SS_T = \sum_{i=1}^{a}\sum_{j=1}^{n_i} x_{ij}^2 - \frac{x^2..}{N}$ $SS_{Treatment} = \sum_{i=1}^{a}\frac{x_{i.}^2}{n_i} - \frac{x_{..}^2}{N}$ $SS_{Error} = SS_T - SS_{Treatment}$ $MS_{Treatment} = SS_{Treatment}/(a-1)$ $MS_{Error} = SS_{Error}/(N-a)$ $F_0=MS_{Treatment}/MS_{Error}$ Where N is the total number of measurements and a dot index denotes a summation over the dotted index, e.g. $x_{i.} = \sum_j x_{ij}$
Criterion	Reject H_0 if $F>F_{\alpha,a-1,N-a}$. Here, $F_{\alpha,f1,f2}$ is the upper α percentage point of the F distribution with *f*1 and *f*2 degrees of freedom, which is tabulated.

T-test.

When there are two different samples that are going to be tested against each other a t-test can be used. [WRH00] says that t-test is "a parametric test used to compare two independent samples". The test is performed as showed in Table 4.

Table 4: Performing of a t-test[20]

t-test	
Input	Two independent samples: $x_1, x_2, \ldots x_n$ and $y_1, y_2, \ldots y_m$.
H_0	$\mu_x = \mu_y$, i.e. the expected mean values are the same.
Calculations	Calculate $t_0 = \frac{\bar{x}-\bar{y}}{S_p\sqrt{\frac{1}{n}+\frac{1}{m}}}$, where $S_p = \sqrt{\frac{(n-1)S_x^2+(m-1)S_y^2}{n+m-2}}$
Criterion	Two sided (H_1: $\mu_x \neq \mu_y$): reject H0 if $\lvert t0\rvert > t_{\alpha/2,n+m-2}$. Here $t_{\alpha,f}$ is the upper α percentage point of the t distribution with *f* degrees of freedom, which is equal to n+m-2. One sided (H_1: $\mu_x > \mu_y$): reject H0 if $\lvert t0\rvert > t_{\alpha/2,n+m-2}$

[19] Adapted from [WRH00]

[20] Adapted from [WRH00]

Both of the tests have two or more samples and a null hypothesis as input. Then calculations are done on the samples to see if the null hypothesis has to be rejected or accepted. In this thesis the tests where however run in excel. To do this the data was plotted into an excel worksheet, and then the test is run on the data by selecting them and selecting the right test. The data from the test is then presented, and the conclusions can be made. The test will be run with a significance level of 95 %.

4 Results and Main Findings

In this chapter the results gathered from the thesis and the IKT-Norge surveys are presented. As explained before only the extra questions added for NTNU are explained from the IKT-Norge survey. The main findings of the surveys are also presented. In this chapter the question numbers are also used, question QT1 - QT29 are from the thesis survey and can be seen in Appendix A. Questions QN7.2 – QN7.16 are from the IKT-Norge survey and can be seen in Appendix B. **Through this chapter N and NeN is used in the tables and figures to state the number of subjects who answered the questions. N (number) is used where the subjects could only answer one alternative and NeN (non-exclusive number) is used where it was possible to answer more than one alternative.** These are also used when calculating the percentage values in the tables. The percentage is the number of subjects who answered the alternatives. The tables and the diagrams from the IKT-Norge survey are marked with IKT-Norge, and the ones from the thesis survey are marked with Thesis.

As mentioned before both tables and histograms are used to present some answers. These present much of the same data, but the reason why both are used is because the information is easier to interpret in a histogram, but because also the percentage value is needed to see the relations, tables are also used.

4.1 IKT-Norge Survey

Even though the answers from the IKT-Norge survey isn't used to answer the research questions, because they aren't grouped in the same way as the thesis survey ones (by business type and size), the information collected is however valuable. It also had more answers than the thesis survey.

A print out of the questionnaire used can be seen in Appendix B, and all the data collected can be seen in Appendix D.

The subjects in the IKT-Norge survey were asked about how many man-labour years they had in the companies. This was however only answered by a few, and could therefore not be used to group the companies. Therefore total income was used to group the companies that participated. The income numbers that were used are taken from Brønnøysunnsregistrene[21] and are the total income the company reported in 2004. In Table 5 the numbers for all the companies who participated in the survey are shown, and in Table 6 the numbers for the companies who also answered the extra part is shown. Total shows the total income for the companies that are in this group in million NOK. As mentioned before there were 142 subjects that answered this survey, and 60 who answered the extra part. Some of these companies however hadn't reported their total income to Brønnøysunnsregisteret[21], and could therefore not be considered in the estimate. The total income for the companies over ≥100 is very large compared to the other groups. This is because some of the companies were very large

[21] Brønnøysunnsregistrene - http://www.brreg.no/

companies that are into many businesses (three of the companies in this group had a collected income of over 7000).

Table 5: Companies grouped by income for all companies –IKT-Norge

	Companies	% Companies	Total	%Total	Mean	Median
0-19	83	65 %	546,84	4 %	6,67	5,30
20-99	27	21 %	1002,44	8 %	37,13	30,50
≥100	17	13 %	11162,89	87 %	656,64	156,66
N=	127					

Table 6: Companies grouped by income for companies who answered the extra part –IKT-Norge

	Companies	% Companies	Total	%Total	Mean	Median
0-19	32	59 %	205,92	3 %	5,42	4,06
20-99	14	26 %	523,8	7 %	40,29	28,55
≥100	8	15 %	6431,46	90 %	803,93	145,88
N=	54					

The survey had a total of 142 answers, 60 of these (42 %) agreed to answer the extra questions which are interesting for this thesis. The first 6 questions are however irrelevant as they are on quality systems and process improvement. They are however presented here for later use.

4.1.1 Quality Systems and Process improvement

The first question, after the question about whether or not the company wanted to answer the questions from NTNU, was a question about background info (question QN7.2). Here the subjects had to answer either yes or no on 7 questions. The questions and the answers to them are shown in Table 7.

Table 7: Background info –IKT-Norge

	Yes	No	%Yes
Has the company cooperated with a Norwegian R&D institution to improve the development processes/better program quality in the last 5 years?	18	42	30 %
Has the company hired a consulting firm to better the development processes or program quality in the last 5 years?	22	38	37 %
Is the company ISO-9000 certified	5	55	8 %
Is the company planning to be ISO-9000 certified within the next 2 years?	14	46	23 %
Has the company created an internal quality system?	42	18	70 %
If No, has the company plans on creating this kind of system in the next 2 years?	25	35	42 %
Is the company part of a bigger enterprise which has made an internal quality system?	11	49	18 %
NeN=	60		

If the subjects answered that the company was part of a bigger enterprise which had made an internal quality system they were asked to rate how much it had been altered locally (question QN7.3). The answers are summed up in Table 8. From the numbers it is apparent that most of the companies have altered the system to some degree. Only 2 of the 11 subjects (18 %) answered that their companies hadn't altered the system locally.

Table 8: Company has altered internal quality system locally –IKT-Norge

Not altered (1)	2	18 %
Altered some (2)	4	36 %
Altered	1	9 %
Altered a lot	4	36 %
Don't know	0	0 %
N=	11	

On the next question (question QN7.4) the subjects were asked if their companies had an ongoing initiative in process improvement. As shown in Figure 3, 39 of the 60 subjects (65 %) answered yes on this question.

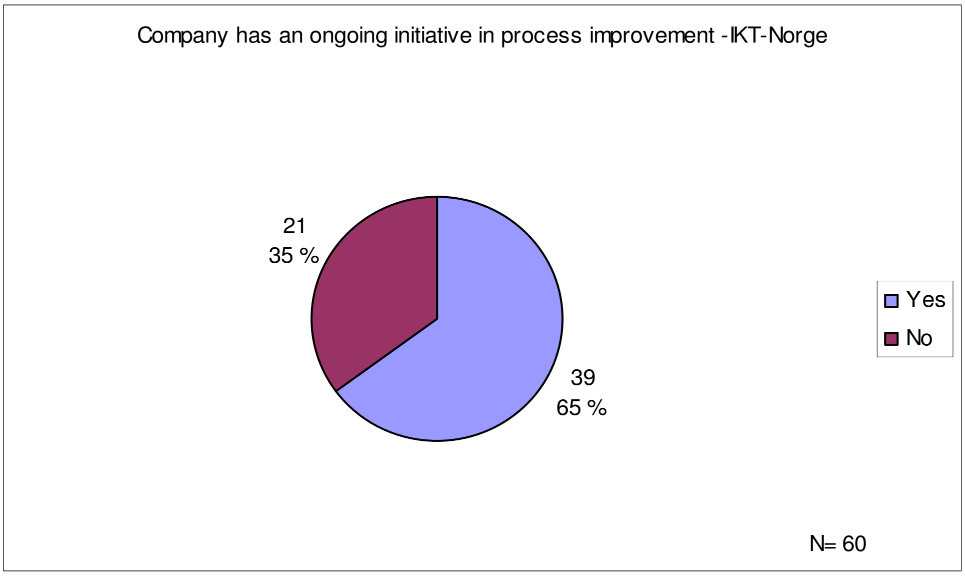


Figure 3: Company has an ongoing initiative in process improvement –IKT-Norge

The subjects that answered yes were than asked what the motives were for this. The answers are shown in Table 9 and Figure 4. From the numbers it is apparent that better program quality is definitively the most used motive in the companies (92 %), but also better reputation among customers (74 %) and lower cost (62 %) scores high.

Table 9: Motives for process improvement –IKT-Norge

Lower costs	24	62 %
Shorter development time	22	56 %
Better program quality	36	92 %
Better reputation among customers	29	74 %
Other	2	5 %
NeN=	39	

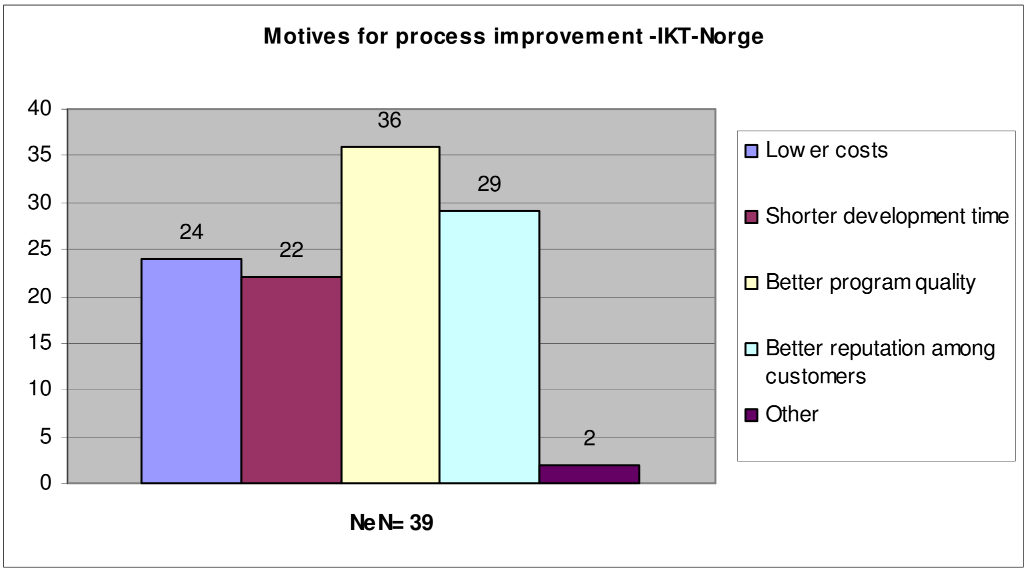


Figure 4: Motives for process improvement –IKT-Norge

Then the subjects were asked whether or not their companies used agile development processes in their development (question QN7.6). 29 of the 60 subjects (48 %) answered that this was used in their companies. This is shown graphically in Figure 5.

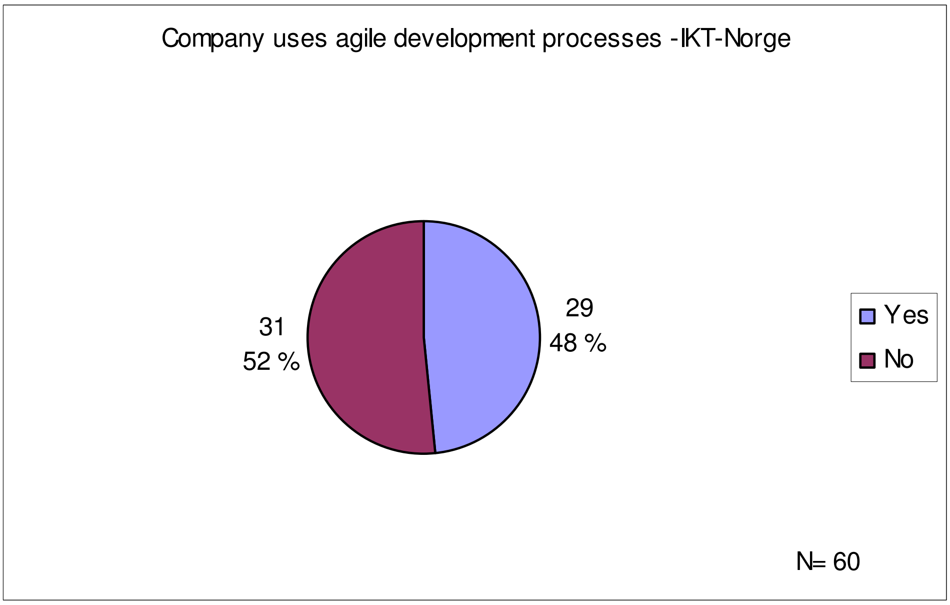


Figure 5: Company uses agile development processes –IKT-Norge

The subjects who answered that their companies used agile development processes were asked about the motives for this. The answers are shown in Table 10 and Figure 6. Shorter development time is the most common motive, 22 of the 29 subjects (76 %) answered this as a motive. Better program quality was also a highly used motive which 19 of the 29 subjects (66 %) answered. Only 13 of the 29 subjects (45 %) saw lower cost as a motive however.

Table 10: Motives for agile development processes –IKT-Norge

Lower costs	13	45 %
Shorter development time	22	76 %
Better program quality	19	66 %
Other	5	17 %
NeN=	29	

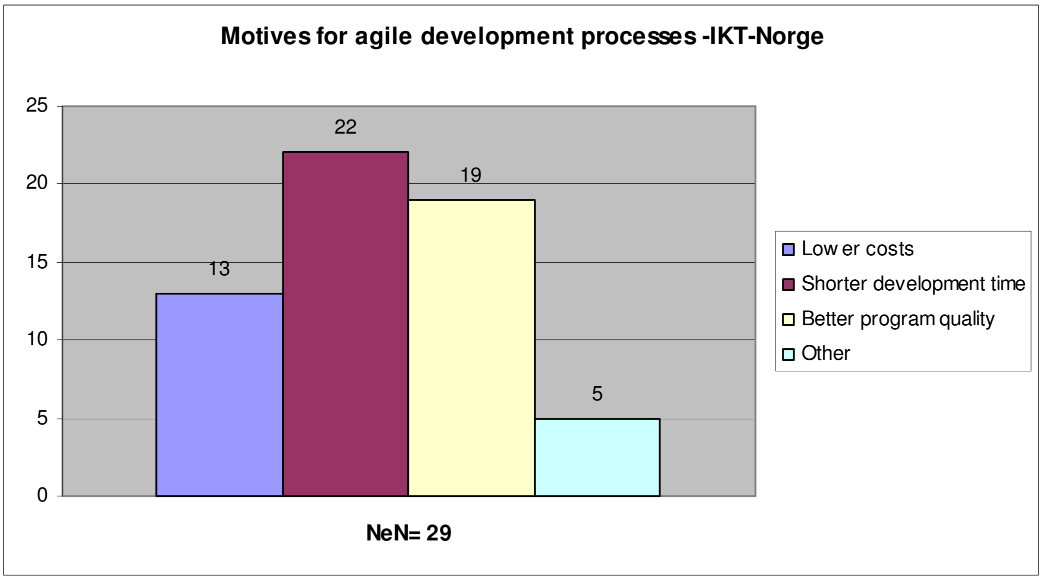


Figure 6: Motives for agile development processes –IKT-Norge

4.1.2 The use of Component-based Development with COTS/OSS.

The first question that was relevant to the thesis (question QN7.8) was whether or not the company used component based development with COTS and/or OSS. 24 of the subjects (40 %) answered that their company used COTS while 29 (48 %) answered that their companies used OSS. A total of 41 subjects (68 %) answered that they used at least one of COTS and OSS.

This data is than interpreted as follows for later comparison: 12 of the companies used only component based development with COTS, while 17 only used with OSS. 12 companies used both OSS and COTS and 19 used neither.

The answers are summed up in Table 11 and shown graphically in Figure 7.

Table 11: Companies use of COTS/OSS –IKT-Norge

COTS only	12	20 %
OSS only	17	28 %
Both COTS and OSS	12	20 %
None	19	32 %
N=	60	

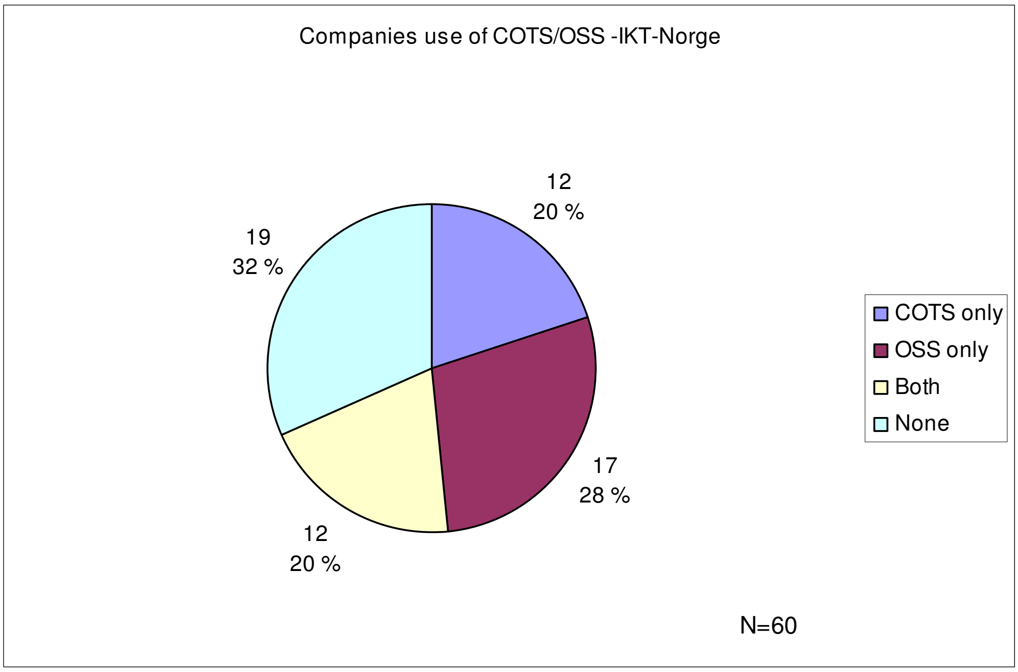


Figure 7: Companies use of COTS/OSS –IKT-Norge

On question on motives for using COTS or OSS (question QN7.10) it is apparent that the most common reason (79 % of the companies who answered they used either COTS or OSS) for utilizing it is lower costs. All of the alternatives were however commonly used as shown in Table 12 and Figure 8. The company that answered that they used COTS or OSS for other reasons answered that they did it because it was "less work".

Table 12: Motives for using of COTS or OSS –IKT-Norge

Lower costs	33	79 %
Shorter development time	31	74 %
Better program quality	26	62 %
More standardized products	30	71 %
Other	1	2 %
NeN=	42	

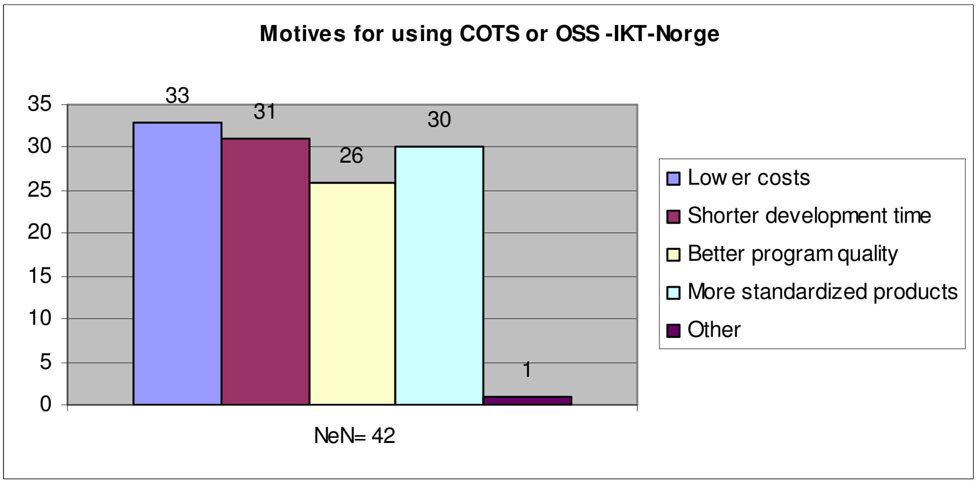


Figure 8: Motives for using COTS or OSS –IKT-Norge

4.1.3 Systematic Reuse

The subjects were asked if their companies applied systematic reuse in their development process (question QN7.11). As shown in Figure 9, 55 (92 %) of the companies used systematic reuse, while only 5 (8 %) didn't.

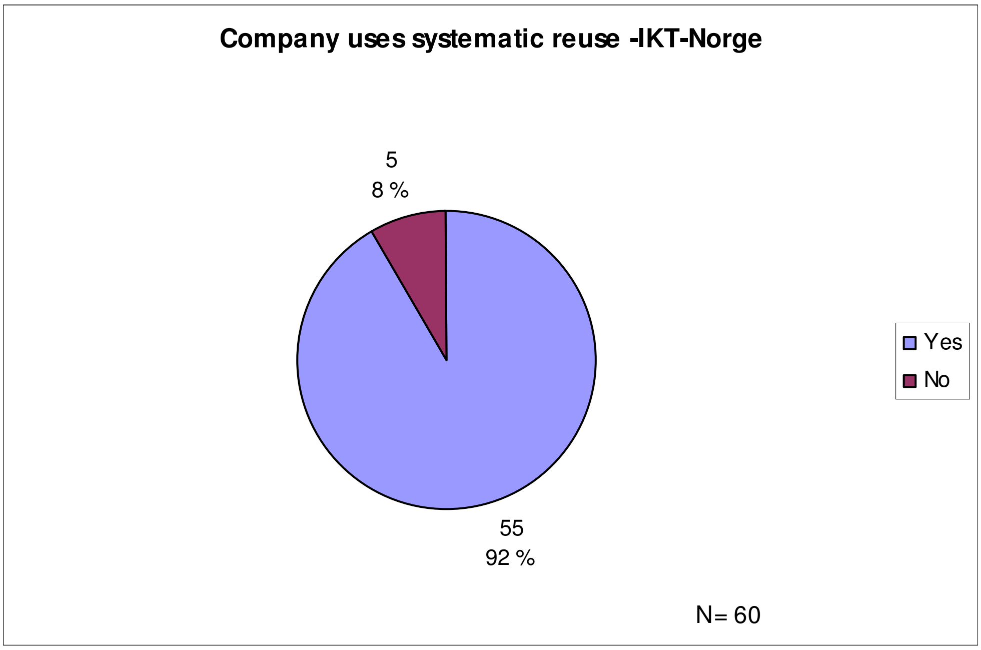


Figure 9: Company uses systematic reuse –IKT-Norge

Further the subjects who answered that their company applied reuse were asked to specify what types of components were reused in their companies (question QN7.12). The answers are show in Table 13 and Figure 10. From the answers it is easy to see that code modules/components are the most used reuse components by good margin, 49 of the 55 subjects (89 %) answered that it was reused. The design/architecture is also used by a lot of the companies (39 of 55) while documentation, test plans and requirement specifications are used by some companies, but not all.

Table 13: Types of components that are reused –IKT-Norge

Requirement specification	25	45 %
Design/architecture	39	71 %
Code modules/components	49	89 %
Documentation	32	58 %
Test plans	32	58 %
Other	1	2 %
NeN=	55	

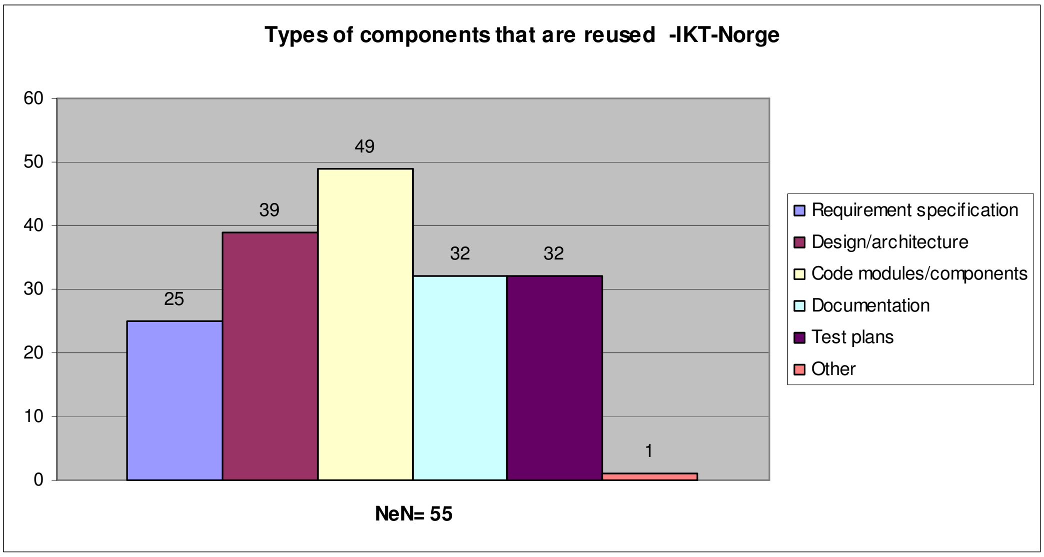


Figure 10: Types of components that are reused –IKT-Norge

The next question asked about the motives for applying systematic reuse (question QN7.13). The alternatives were the same as in the question on motives for using COTS and OSS. On this question lower costs, shorter development time and better program quality got almost as many answers (49, 48 and 49), while more standardized products got 41. There was one subject that answered other on this question, and specified that his company was also motivated by "easier maintenance and upgrading of customers solution". The answers are shown in Table 14 and Figure 11.

Table 14: Motives for systematic reuse –IKT-Norge

Lower costs	49	89 %
Shorter development time	48	87 %
Better program quality	49	89 %
More standardized products	41	75 %
Other	1	2 %
NeN=	55	

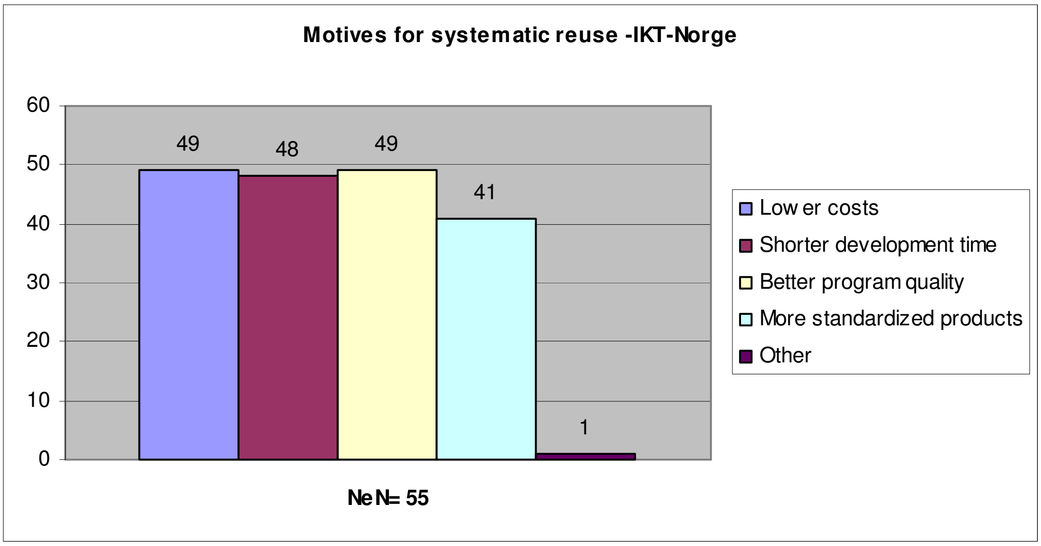


Figure 11: Motives for systematic reuse –IKT-Norge

4.1.4 Outsourcing to other Companies

The final part of the IKT-Norge questionnaire was about outsourcing to other companies. The first question of this part asked whether or not the companies used outsourcing (question QN7.14). On this question 24 of the 60 subjects (40%) answered that their companies used outsourcing.

Furthermore these 24 subjects were asked to which regions they outsourced to (question QN7.15), meaning that they had at least once outsourced to this region. As can be seen graphically in Figure 12, and is summarized in Table 15, the companies (which are all Norwegian) mostly used outsourcing to other Norwegian companies (17 of 24). India is the second largest region (7 of 24), with Western-Europe (5 of 24) on third place.

Table 15: Regions outsourced to –IKT-Norge

Norway	17	71 %
Scandinavia wo/Norway	1	4 %
Western-Europe	5	21 %
Eastern-Europe	2	8 %
India	7	29 %
Asia wo/India	3	13 %
USA	2	8 %
America wo/USA	0	0 %
Other	2	8 %
NeN=	24	

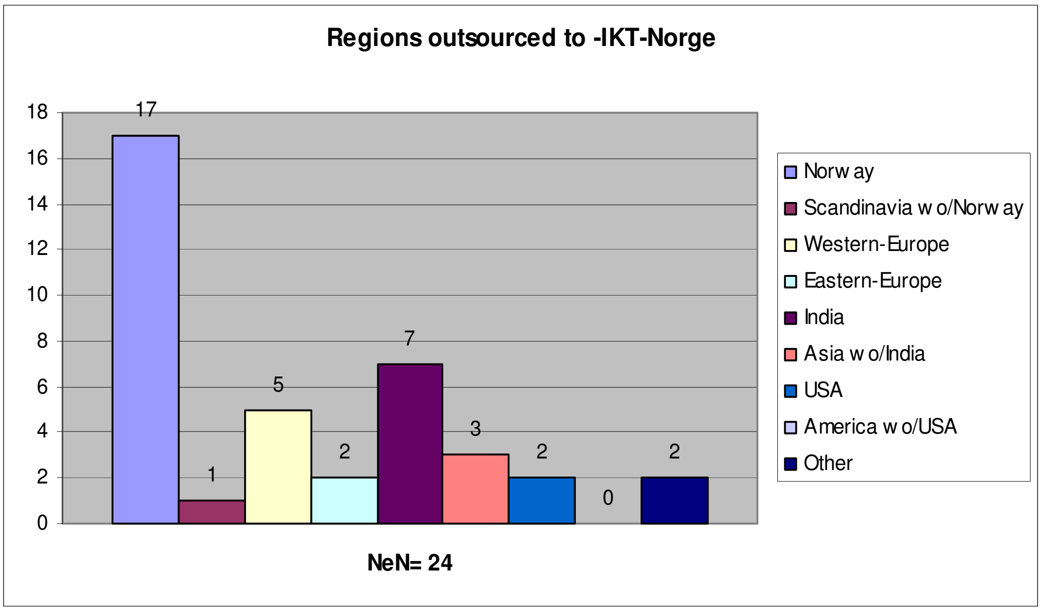


Figure 12: Regions outsourced to –IKT-Norge

The final question of the thesis questionnaire asked the subjects that answered that outsourcing was used what the motives for outsourcing was (question QN7.16). The alternatives were the same as for using COTS and OSS. As seen in Table 16 and Figure 13 lower costs is the definite biggest motivator for outsourcing, 17 of the 24 subjects (71 %) answered that this was a motivator for their companies.

Shorter development time was also a frequent answer (11 of 24), while the subjects didn't think that the programs got better program quality (5 of 24) or that they got more standardized (4 of 24) by using outsourcing. 6 of the subjects answered other on this question. Answers that were given were among others "access to resources", "question of capacity" and "more flexible workforce without long-term commitment".

Table 16: Motives for outsourcing –IKT-Norge

Lower costs	17	71 %
Shorter development time	11	46 %
Better program quality	5	21 %
More standardized products	4	17 %
Other	6	25 %
NeN=	24	

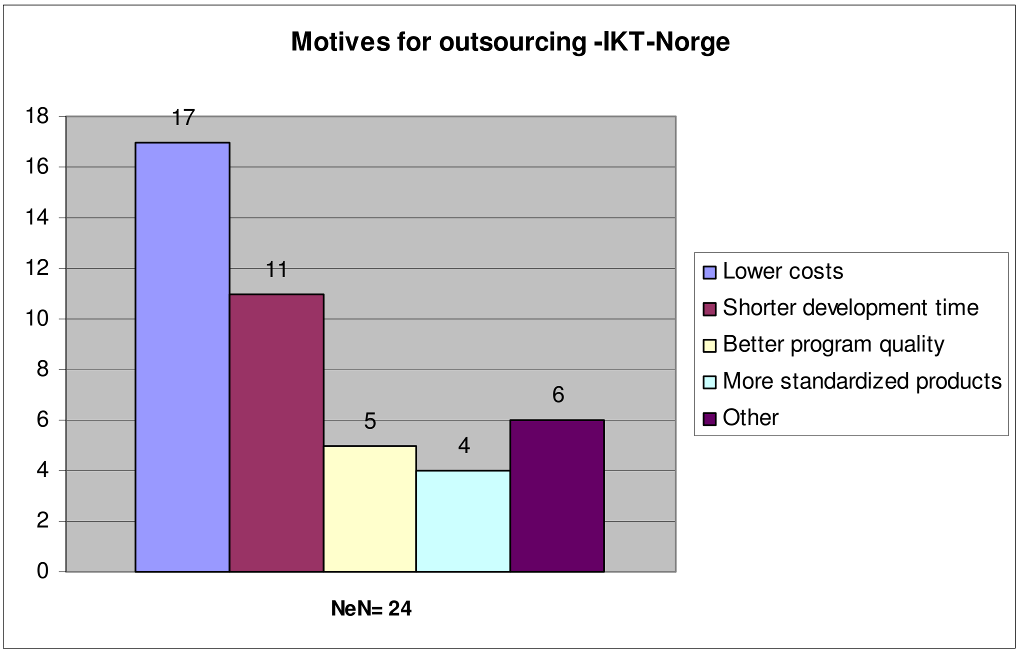


Figure 13: Motives for outsourcing –IKT-Norge

4.2 *Thesis Survey*

In this sub chapter the data collected through the thesis survey is presented. A total of 32 subjects from different companies answered the questionnaire.

The data from the thesis survey can be seen in Appendix C.

4.2.1 General information about The Company

Among the general questions about the company were the company name and name, work title and e-mail address of the subject that answered. None of these are used in the thesis as the questionnaire was an anonymous one.

The subjects were also asked about the number of employees in IT, system development and total (question QT4). These numbers are used to answer some of the research questions by classifying the companies by size. The companies have been put into groups of 0-19 employees, 20-99 employees, and ≥100 employees after the number of total employees. The sizes of the companies involved in the thesis survey are presented in Table 17 and Figure 14. Table 17 also shows the percentage of the total number of companies involved. It is apparent that even though it was strived to get an equal number from the three groups the middle group (20-99) is the biggest group by far. This is because a lot of Norwegian companies are this size, and also because these are the easiest to get to answer. It was hard getting the smallest companies (0-19) to answer because a lot of these had been bought up by larger companies, or they didn't have the resources to answer the questionnaire. With the biggest companies (≥100) on the other hand the problem was getting in contact with the right person for answering the questionnaire, and the e-mail often ended up "drifting" between departments. But as there were enough answers from all of the groups this wasn't considered a problem. When reviewing the number of employees in the companies it is apparent that the largest group is the one with most employees total by far. There was however one of the companies that pulled this number up considerably as they had over 20,000 employees. Median is therefore probably a better measurement.

Table 17: Total number of companies involved –Thesis

	Companies	% Companies	Total	%Total	Mean	Median
0-19	8	25 %	96	0 %	12	12,5
20-99	16	50 %	802	3 %	50,13	40,5
≥100	8	25 %	27110	97 %	3388,75	450
N=	32					

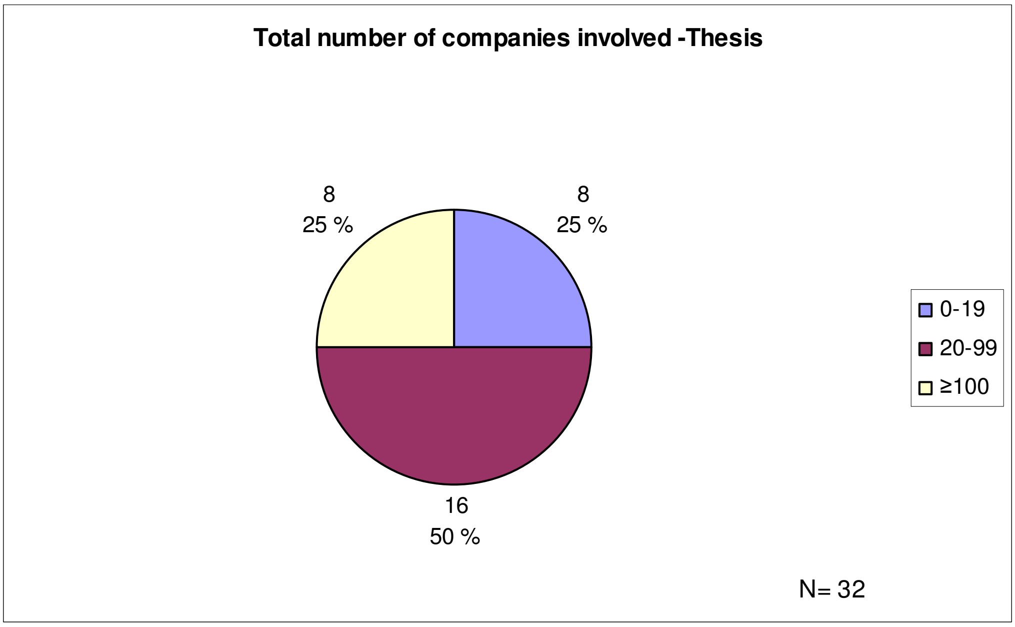


Figure 14: Total number of companies involved -Thesis

The next question asked what the main target group/business domain of the company was (question QT5). Some of the companies however didn't have one, but many primary business domains and the subjects therefore answered more than one of the alternatives. On the alternative other the answer that was most common was “running of servers”. The answers are presented in Table 18 and Figure 15.

Table 18: Business domain -Thesis

Prog. dev and sale	21	66 %
Prog. dev for internal use	1	3 %
Hardware development	6	19 %
Consulting	10	31 %
Other	5	16 %
NeN=	32	

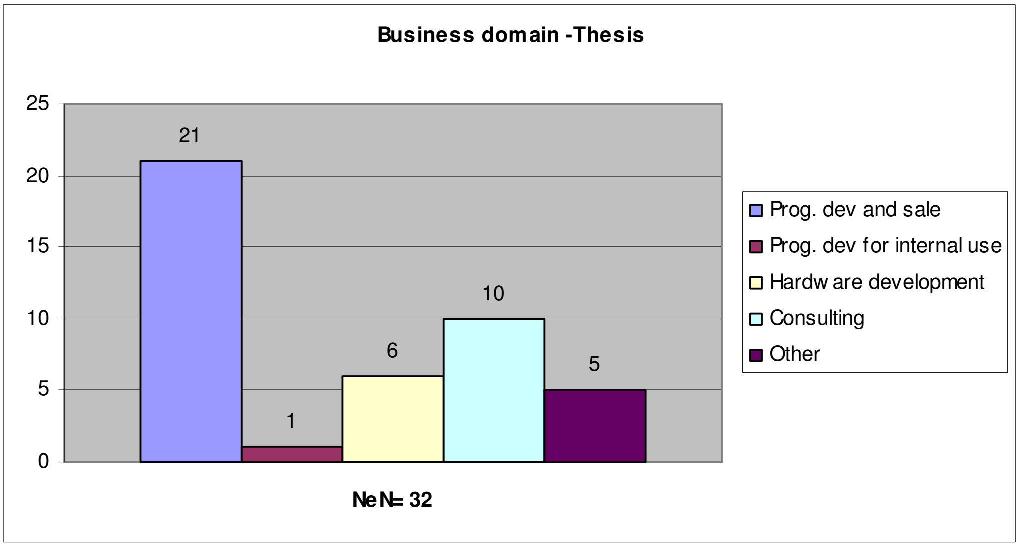


Figure 15: Business domain -Thesis

The subjects were then asked in an open questions to fill in how many larger program development projects they had a year (larger than 2 man labour years) (question QT6), and the number of products/systems that were maintained today (question QT7). The answers to these questions are difficult to sum up (as they were open), but they were used to understand more about the companies. The answers can be seen in Appendix C.

4.2.2 Technical aspects about The Company

The next part of the questionnaire was about technical issues in the companies. Which program languages they used, which processes and so forth. The first question asked what the main programming languages were (question QT8). The subjects were asked to fill in a maximum of 3 marks, but some of the companies had to use 4 because they worked for large companies that used a lot of different languages. The number of answer the different programming language got and the percentage of the companies who used it is shown in Table 19. The numbers are also presented graphically in Figure 16. The languages mentioned under other, but not presented in the table and figure were; Flash, Gypta, PHP two times, Progres, Autocad, Actionscript/Javascript, Delphi, Chill, Lisp, Delphi, SDL, Mathlab, Comsol and Perl. It is apparent that the 2 biggest are C++ and Java, but C#, PL/SQL and C are also used as main programming languages by a lot of companies.

Table 19: Main programming languages -Thesis

C++	13	41 %
Java	14	44 %
C#	11	34 %
Cobol	4	13 %
C	9	28 %
VB	6	19 %
Fortran	1	3 %
PL/SQL	10	31 %
Other	13	41 %
NeN=	32	

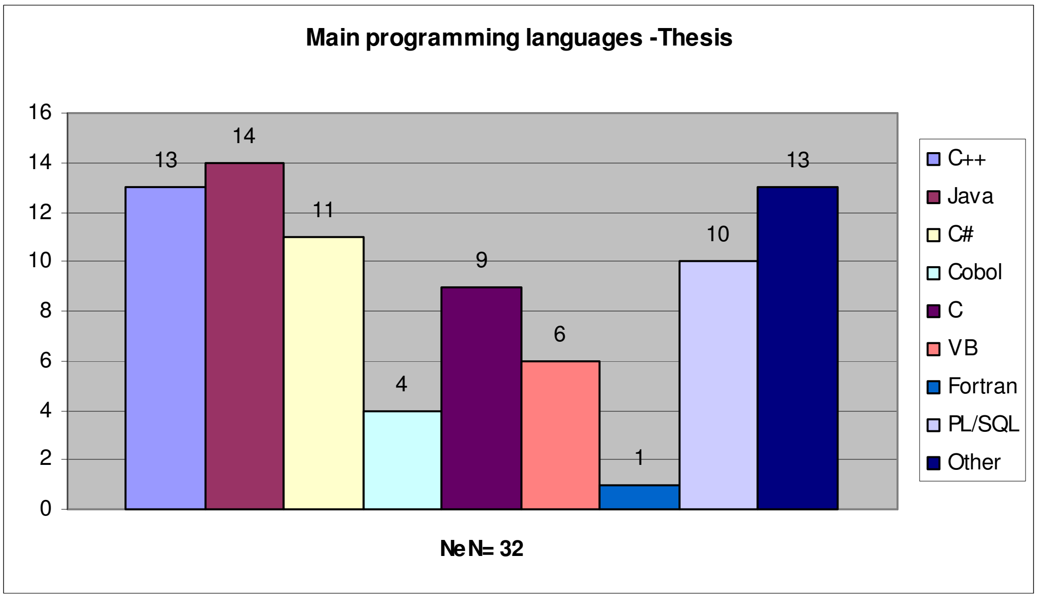


Figure 16: Main programming languages -Thesis

The subjects were also asked if any other programming languages were used (secondary) beside the main ones (question QT9). The same alternatives as on main programming languages were given. The answers are shown in Table 20 and Figure 17. Visual Basic (VB) was the most used secondary programming language among the companies participating (31 %). This is probably because a lot of companies use it for maintaining older programs, but it isn't one of their main languages. The languages that were listed under other were: ASP/VBScript, Javascript, Chill, Python and PHP.

Table 20: Secondary programming languages -Thesis

C++	5	16 %
Java	5	16 %
C#	2	6 %
Cobol	1	3 %
C	5	16 %
VB	10	31 %
Fortran	0	0 %
PL/SQL	6	19 %
Other	5	16 %
NeN=	32	

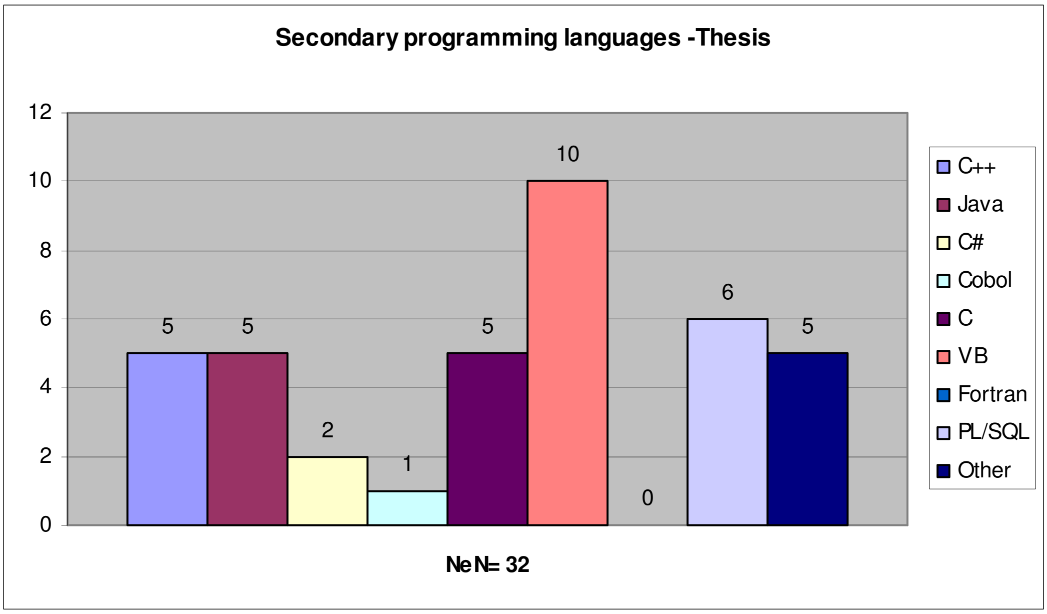


Figure 17: Secondary programming languages -Thesis

The next question asked what tools the companies used in their development processes (question QT10). The question had 7 alternatives including other, but it was apparent that there were many tools not mentioned that were used by the companies. Table 21 sums up the answers, and Figure 18 shows them graphically. There were a lot of different answers on other, one of the subject answered that his company used over 30 more.

Table 21: Development tools -Thesis

.net	19	59 %
Powerbuilder	2	6 %
Oracle	11	34 %
Delphi	3	9 %
UML	14	44 %
J2EE	10	31 %
Other	10	31 %
NeN=	32	

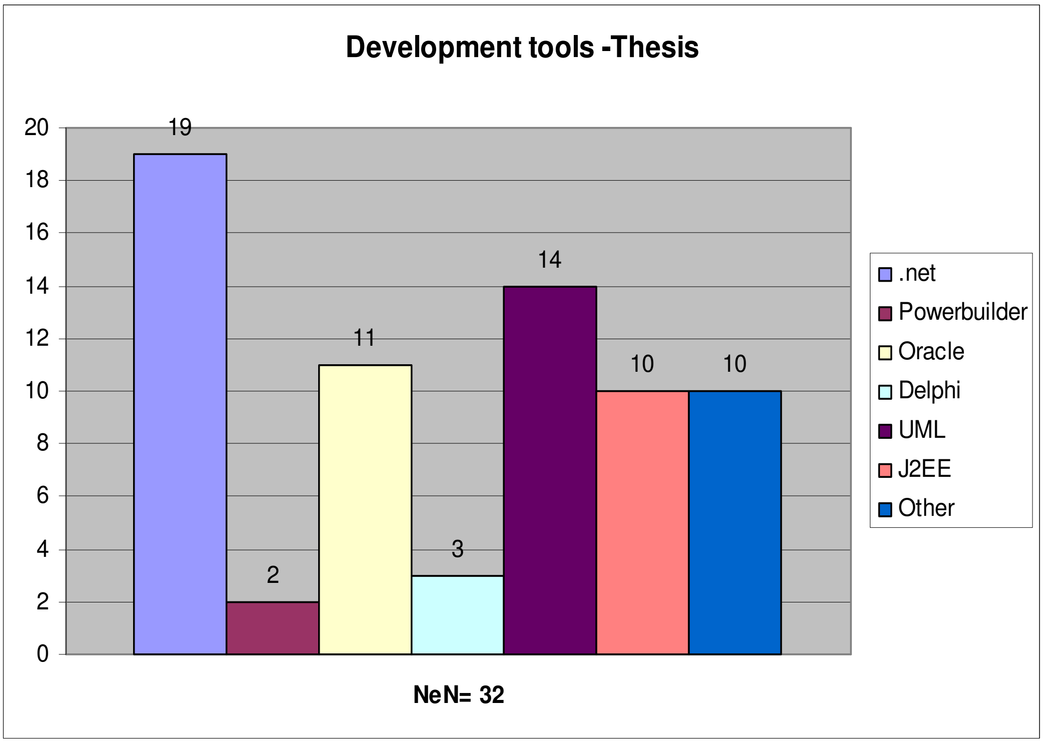


Figure 18: Development tools -Thesis

The last question in this part of the questionnaire asked which processes and methods the company used in their development process (question QT11). The alternatives to the question and the answers from this question are shown in Table 22 and Figure 19. On other 4 of the 7 subjects answered that they used their own methods which they had made. Today it is becoming more usual to do this as the companies create processes that are adapted to how they develop.

Table 22: Development processes/methods -Thesis

RUP	10	31 %
Waterfall	14	44 %
Xp	9	28 %
Incremental	14	44 %
Prototyping	18	56 %
Other	7	22 %
NeN=	32	

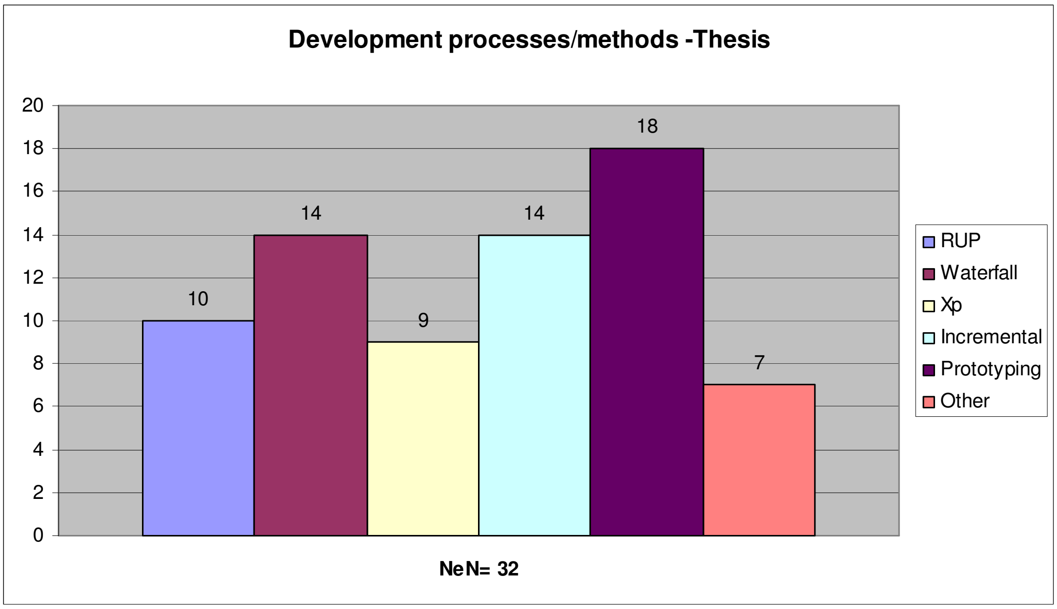


Figure 19: Development processes/methods -Thesis

4.2.3 Systematic Reuse

The next part of the questionnaire was about the utilization of reuse in the companies. The first question was whether or not the company that the subjects worked for applied reuse based on component based development (question QT12). If the subjects answered no they didn't have to answer any of the other questions in this part of the questionnaire because they were all about reuse. Of the 32 subjects that answered the questionnaire 28 (87%) answered that their company applied reuse, while 4 (13%) answered that they didn't. This is shown graphically in Figure 20.

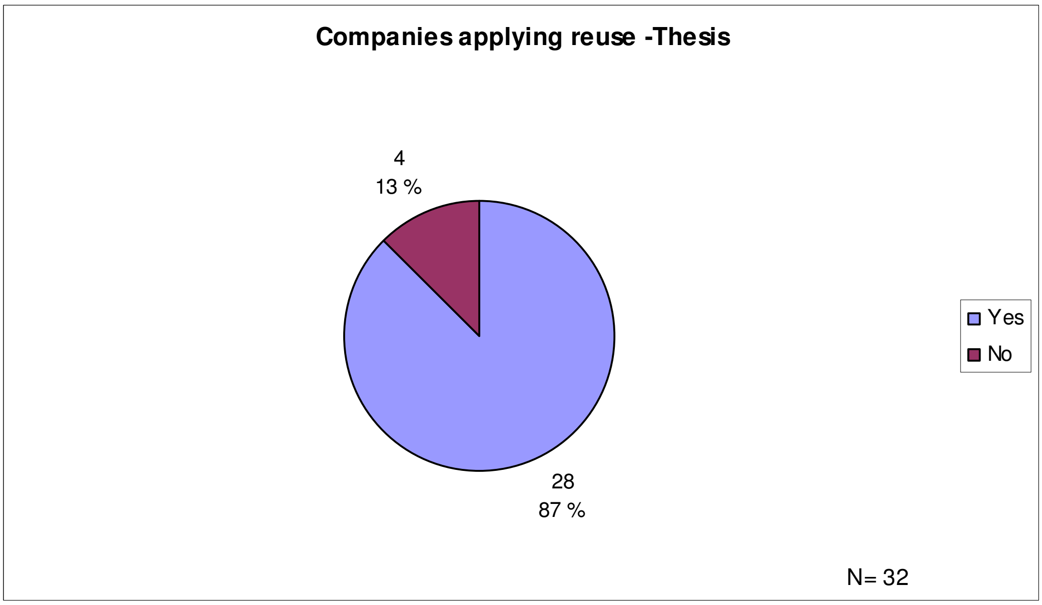


Figure 20: Companies applying reuse -Thesis

As mentioned the rest of this part of the questionnaire was only filled out by the subjects that answered yes on whether or not they used reuse. This was 28 subjects.

The subjects were asked how long systematic reuse had been used in their company (question QT13). The alternatives were below 1 year, 1-2 years, 3-4 years, 5-6 years and above 6 years. The answers are shown in Table 23.

Table 23: How long reuse had been used -Thesis

<1 years	0	0 %
1-2 years	4	14 %
3-4 years	7	25 %
5-6 years	2	7 %
>6 years	15	54 %
N=	28	

On the next questions a scale of 1 to 5 was used to represent the data when estimating the mean value. On this scale very bad is 1, bad is 2, ok is 3, good is 4 and very good is 5. The question was how well the subjects thought reuse worked in their companies (question QT14), and the answer was a nominal scale with these five alternatives. The answers are shown in Table 24 and Figure 21. Most of the subjects felt that their reuse worked ok. Only one of the subjects answered that it worked worse than okay (bad), while 8 answered that it worked good, and 3 that it worked very good. It is apparent that almost all of the subjects are at least satisfied with how reuse works in their company. The mean value is 3,46. This means that it is somewhere between ok and good.

Table 24: How well reuse works in the companies -Thesis

Very bad (1)	0	0 %
Bad (2)	1	4 %
Ok (3)	16	57 %
Good (4)	8	29 %
Very good (5)	3	11 %
N=	28	
Mean	3,46	
SD	0,744	

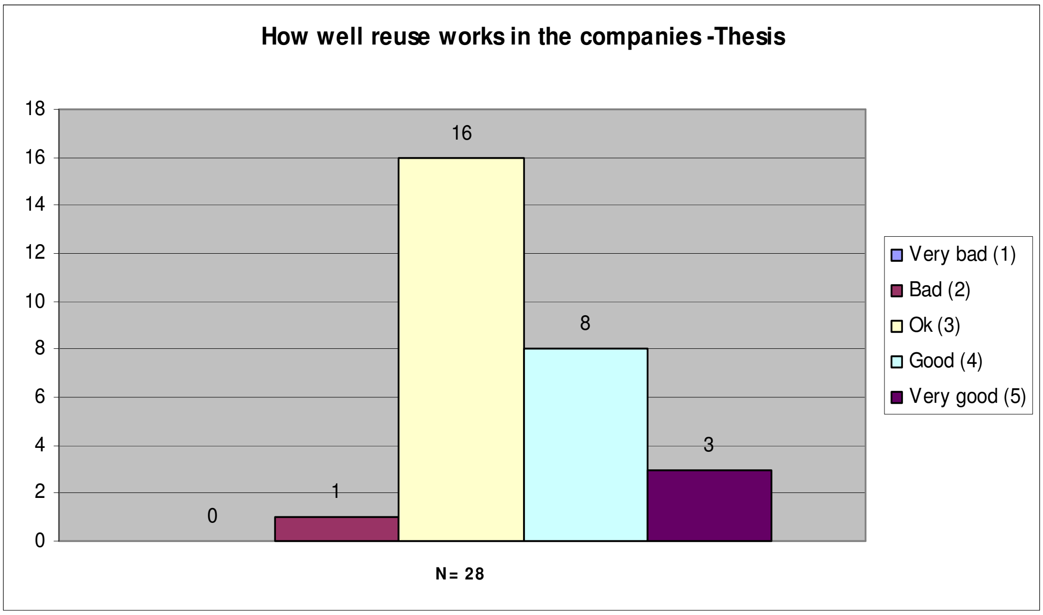


Figure 21: How well reuse works in the companies -Thesis

If this is seen from the sizes of the companies we get a little different numbers. Here only the mean values for each of the groups are shown. The values are shown in Table 25 and are discussed further later in the thesis.

Table 25: How well reuse works in the companies grouped by company size -Thesis

	Mean	SD	N
1-19	3,5	1,048	6
20-99	3,538	0,66	14
≥100	3,375	0,744	8
All	3,46	0,744	28

For answering research questions later in the thesis, how well reuse works in the companies, only for the companies with departments, are shown in Table 26.

Table 26: How well reuse works in the companies with internal development departments - Thesis

Very bad (1)	0	0 %
Bad (2)	1	5 %
Ok (3)	12	55 %
Good (4)	6	27 %
Very good (5)	3	14 %
N=	22	
Mean	3,5	
SD	0,802	

For later comparisons the answers have also been grouped after the different programming languages the subjects answered were their main ones. This is a little different information as many of the companies used a lot of different programming languages (as shown in Table 19 and Figure 16), and therefore answered many of them. The numbers are here shown in Table 27 and Figure 22. It can be seen that the differences between the numbers are small. This will be discussed further later in the thesis.

Table 27: How well reuse works grouped by programming language -Thesis

	Very bad (1)	Bad (2)	Ok (3)	Good (4)	Very good (5)	Mean	SD	NeN=
C++	0	0	6	3	2	3,64	0,809	11
Java	0	0	7	4	1	3,50	0,674	12
C#	0	0	6	4	0	3,40	0,516	10
Cobol	0	0	3	1	0	3,25	0,5	4
C	0	0	7	1	0	3,13	0,353	8
VB	0	0	2	2	0	3,50	0,577	4
Fortran	0	0	1	0	0	3,00	0	1
Pl/SQL	0	0	5	3	1	3,56	0,726	9

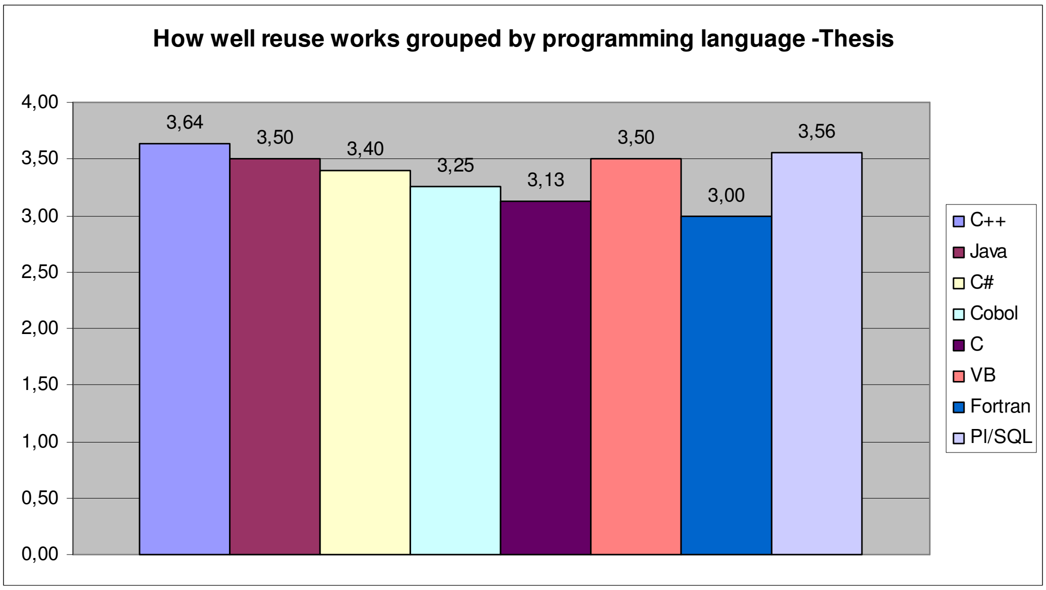


Figure 22: How well reuse works grouped by programming language -Thesis

In Table 28 how well reuse works in the companies are grouped by what processes/methods were used. Also here many of the subjects have answered many alternatives, as they used different ones. It is apparent from the numbers that there isn't any big difference between which process/method the company uses, and how well reuse works.

Table 28: How well reuse works, grouped by development process/methods -Thesis

	Very bad (1)	Bad (2)	Ok (3)	Good (4)	Very good (5)	Mean	SD	NeN=
RUP	0	0	5	5	0	3,50	0,527	10
Waterfall	0	1	7	4	2	3,50	0,855	14
Xp	0	0	5	2	1	3,50	0,756	8
Incremental	0	1	8	4	1	3,36	0,745	14
Prototyping	0	1	9	4	2	3,44	0,81	16

The next question was almost the same as the last one, except it asked how well reuse worked in the department which the subject worked (question QT15). This is a question which only the subjects from the bigger companies answered as they are the ones who usually have more departments than one. 22 of the 28 subjects (79 %) that used reuse also answered this question. The results can be seen in Table 29 and Figure 23. These numbers are discussed later in the thesis.

Table 29: How well reuse works in the company departments -Thesis

Very bad (1)	0	0 %
Bad (2)	2	9 %
Ok (3)	9	41 %
Good (4)	9	41 %
Very good (5)	2	9 %
N=	22	
Mean	3,5	
SD	0,801	

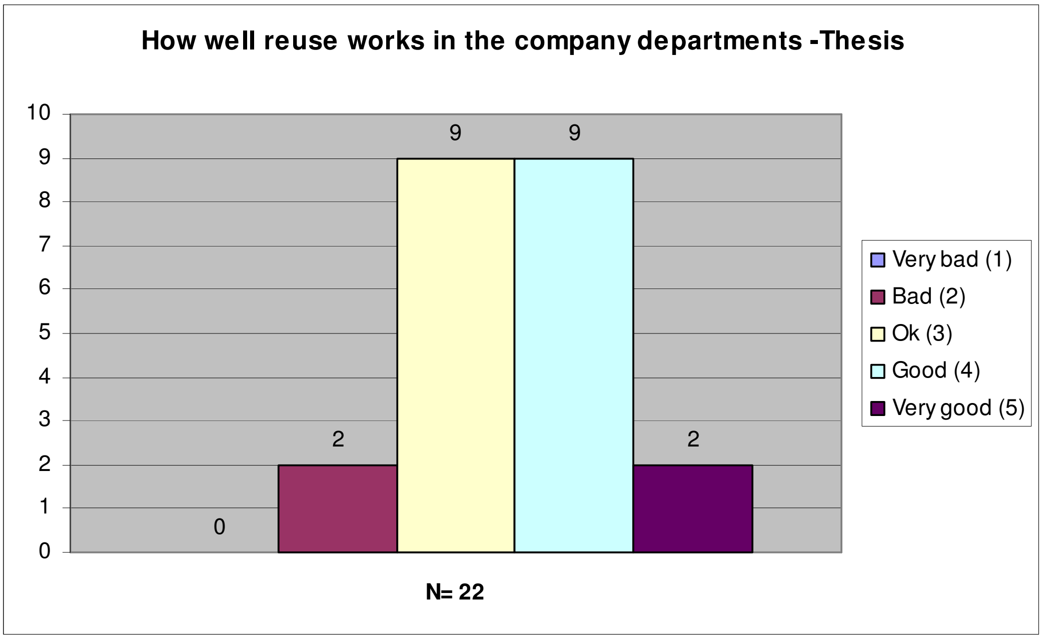


Figure 23: How well reuse works in the company departments -Thesis

Then the subjects were asked which components were reused by their company (question QT16). The components they could select among were open-source (OSS), commercial-of-the-shelves (COTS), internally developed and outsourced components. The answers from this question are shown in Table 30 and Figure 24. It is apparent that most of the companies (93 % of the companies asked) use internally developed components. It has today become common practice to create components that can be reused later. Also a lot of the subjects answered that their companies used OSS (57 %) and COTS (54 %) components. On the other hand only two of the subjects answered that outsourced components were used.

Table 30: Components reused by the companies -Thesis

OSS	16	57 %
COTS	15	54 %
Intern. Dev	26	93 %
Outsourced	2	7 %
NeN=	28	

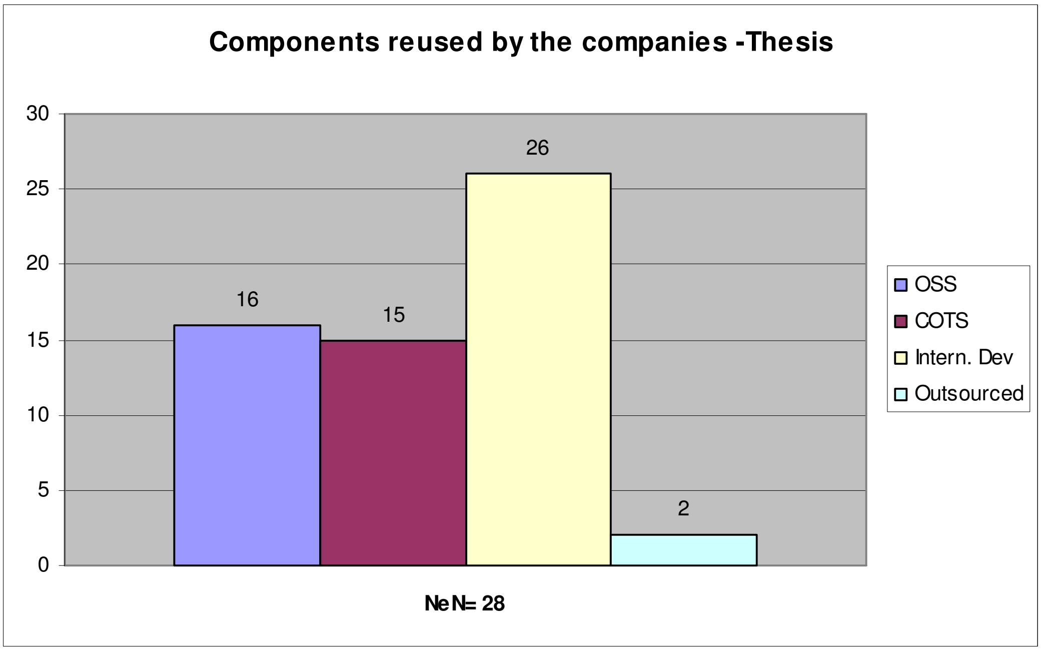


Figure 24: Components reused by the companies -Thesis

For later comparisons with the IKT-Norge survey the numbers are here presented in the same manner as they were in it. Table 31 and Figure 25 shows how many of the subjects answered that their companies used COTS, OSS, both COTS and OSS, or none of them.

Table 31: Companies use of COTS/OSS -Thesis

COTS only	5	18 %
OSS only	6	21 %
Both COTS and OSS	10	36 %
None	11	39 %
N=	28	

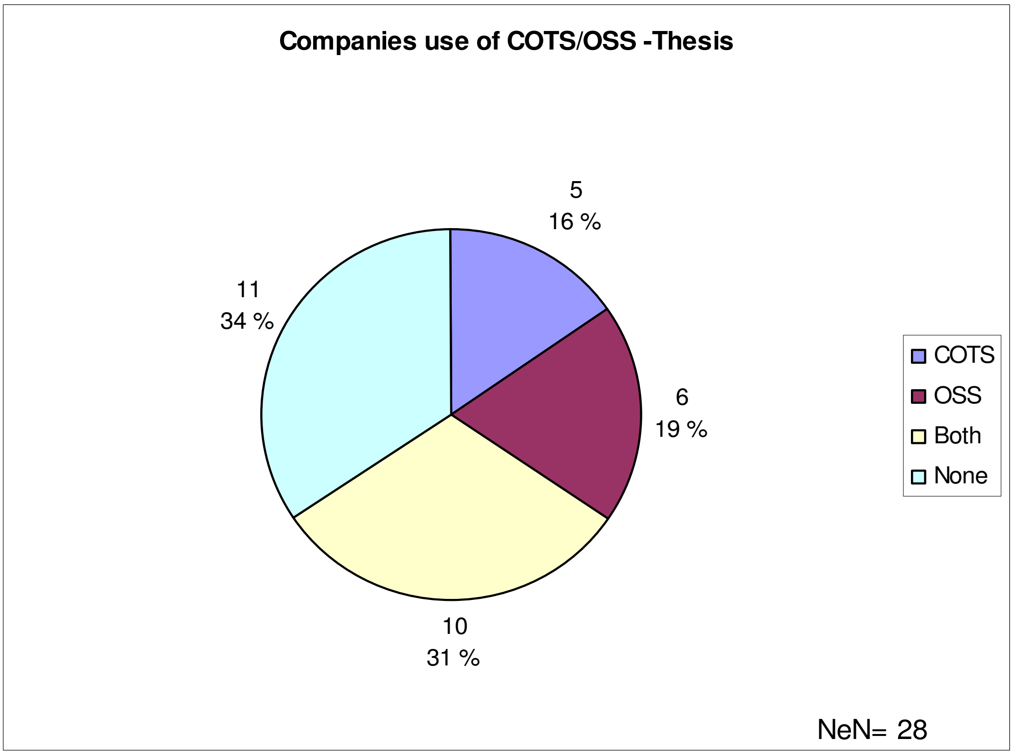


Figure 25: Companies use of COTS/OSS -Thesis

Then the subjects were asked what types of components were reused in their companies (question QT17). There were 7 alternatives including other. The alternatives and distributions of answers are shown in Table 32 and Figure 26. As assumed code modules were the components that were reused most frequently by the companies (96 % of the companies reused them). Architecture and design of systems are the second most reused components with 18 companies using it (64 %). The rest of the components are almost equally used.

Table 32: Types of components that are reused -Thesis

Requirement specification	10	36 %
Architecture/design	18	64 %
Code modules	27	96 %
Test plans	13	46 %
Documentation	10	36 %
Process models/project plans	15	54 %
Other	3	11 %
NeN=	28	

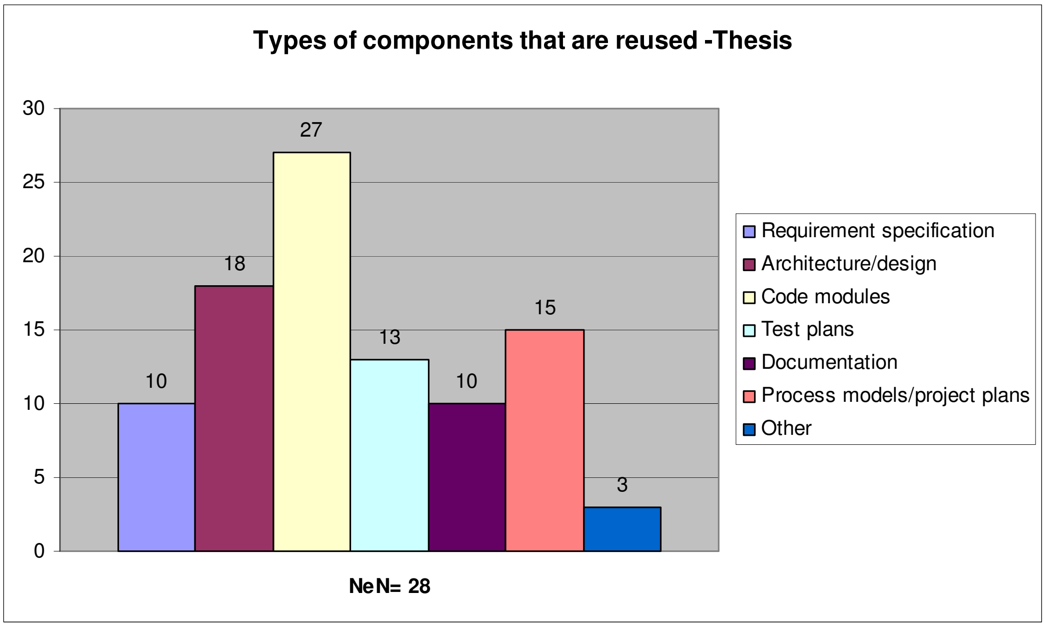


Figure 26: Types of components that are reused -Thesis

The next question was a two parted one. It asked how the companies shared the components between the products. The alternatives were:

- Configuration system/Version control system. A system that is created to handle reuse and version control between products.
- An own database built specifically for sharing of components
- Intranet. A local intranet that isn't created specifically for the sharing of components.
- Shared folders on a local server.
- Other.

This was a two parted question because these could either be department based or company based (question QT18). The answers for the department based systems are shown in Table 33 and Figure 27, while the answers for the company based systems are shown in Table 34 and Figure 28. There were a total of 22 companies who answered on department based sharing, and 28 who answered on company based sharing. It is apparent that the most used way of sharing components is a configuration system, 27 % of the subjects answered this on department based and 79 % on company based. This is a good way to share components which save the developers a lot of time and effort. The second most used, with 23 % of the subjects on department based and 39 % of the subjects on company based, are shared folders on a server, which is a really easy way to share components. The only problem is that it can take a long time to find what you are looking for, if it's even there. An own database and intranet is fairly equal, but on company based the intranet gets a little more answers. This is because a lot of companies have big intranet systems that are used to spread information regardless what it is to the whole company, but to have such a system for use only in a department is not that common.

Table 33: Department based sharing -Thesis

Configuration system	6	27 %
Own database	2	9 %
Intranet	2	9 %
Shared folders	5	23 %
Other	0	0 %
NeN=	22	

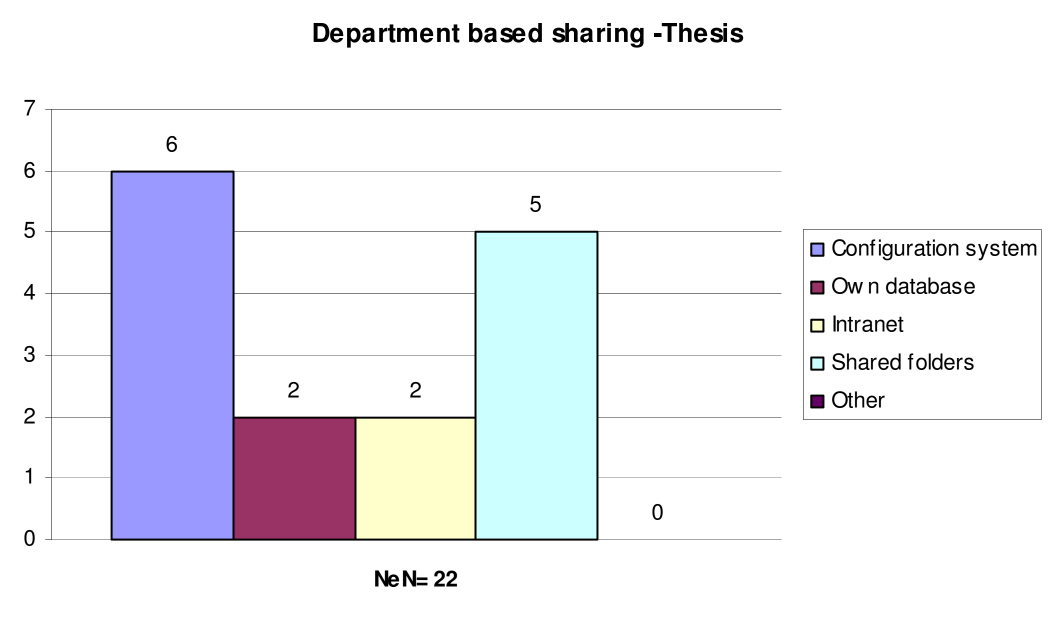


Figure 27: Department based sharing -Thesis

Table 34: Company based sharing -Thesis

Configuration system	22	79 %
Own database	3	11 %
Intranet	7	25 %
Shared folders	11	39 %
Other	3	11 %
NeN=	28	

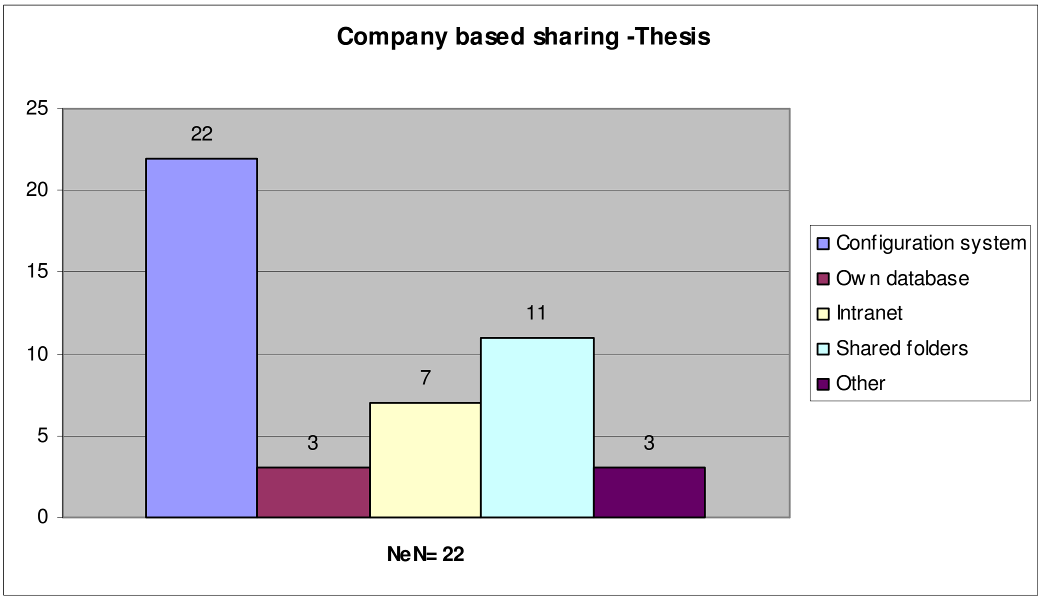


Figure 28: Company based sharing -Thesis

For later comparison Table 35 shows how the companies who had departments did company based sharing.

Table 35: Company based sharing, for companies with departments -Thesis

Configuration system	16	73 %
Own database	2	9 %
Intranet	6	27 %
Shared folders	9	41 %
Other	3	14 %
NeN=	22	

For later comparisons the answers are also here grouped by company size. Only the percentage is shown as this is what is interesting. The department based sharing is shown in Table 36 and Figure 29, while the company based sharing is shown in Table 37 and Figure 30.

Table 36: Department based sharing, grouped by size -Thesis

	1-19	20-99	≥100	All
Configuration system	25 %	30 %	38 %	27 %
Own database	25 %	10 %	0 %	9 %
Intranet	0 %	20 %	0 %	9 %
Shared folders	25 %	20 %	25 %	23 %
Other	0 %	0 %	0 %	0 %
N=	4	10	8	22

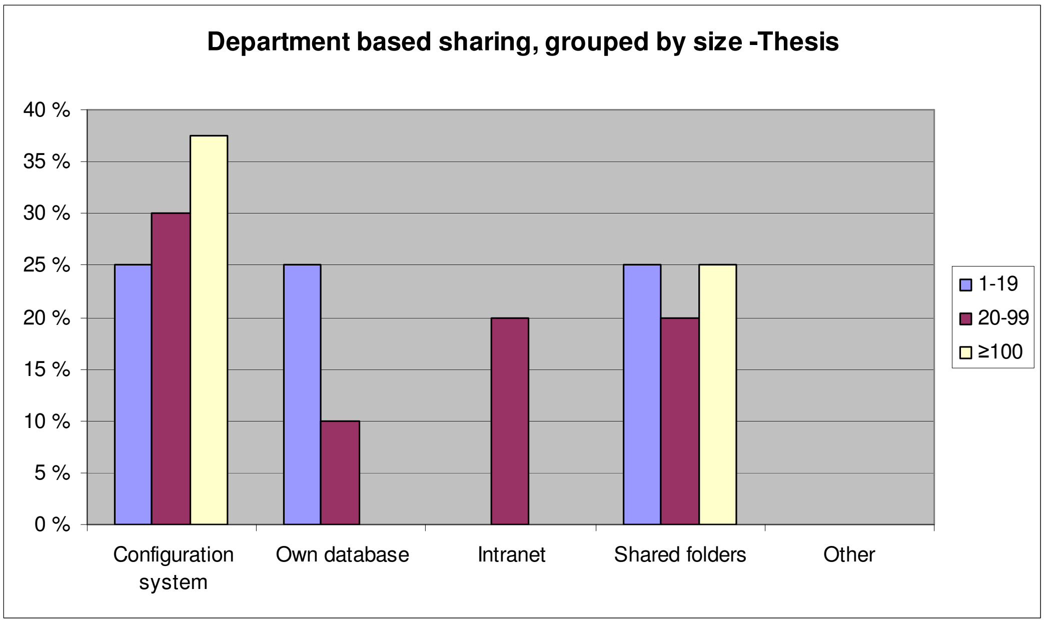


Figure 29: Department based sharing, grouped by size -Thesis

Table 37: Company based sharing, grouped by size -Thesis

	1-19	20-99	≥100	All
Configuration system	100 %	79 %	50 %	79 %
Own database	17 %	7 %	13 %	11 %
Intranet	17 %	14 %	50 %	25 %
Shared folders	50 %	50 %	13 %	39 %
Other	0 %	14 %	13 %	11 %
NeN=	6	14	8	28

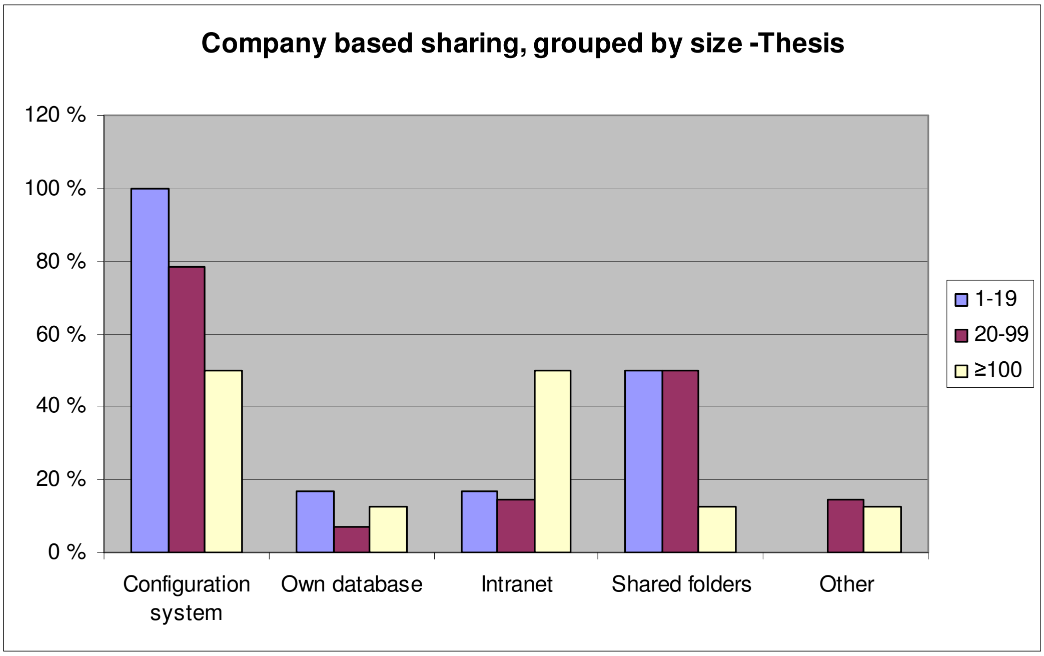


Figure 30: Company based sharing, grouped by size -Thesis

Subjects were furthermore asked to rate the importance of some reasons for reuse with component based development (question QT19), and how important some success factors were for the company (question QT20). The possible answers were:

- Not important (1 point).
- A little important (2 points).
- Important (3 points).
- Very important (4 points).
- Extremely important (5 points).
- Don't know.

These points were used when calculating the mean values.

The mean values for the first of the questions, where the subjects had to rate the importance of different reasons for reuse are shown in Table 38 and Figure 31. Lower cost, shorter development time and better program quality got almost the same mean values (4,19, 4,21 and 4,18). These are looked upon by most of the subjects as very important reasons for applying reuse. That there is less maintenance is also a very important reason (4,00). More standardized programs on the other hand don't score as well among the subjects (3,63). The reason could be that this is more important to the users of the system than the developers. Other is the reason that gets the highest mean value. This is because the subjects only consider what they think are important when they list other reasons, there were also only 7 answers where one was don't know and therefore wasn't use to calculate the mean value. It is therefore hard to get any specifics from its value. The standard deviation values are quite small, with the biggest at 1,122. This means that there is small spread among the answers.

Table 38: Mean values of importance of reasons for reuse -Thesis

	Mean	SD
Lower cost	4,19	0,878
Shorter development time	4,21	0,876
Better program quality	4,18	0,945
More standardized programs	3,63	0,926
Less maintenance	4,00	1,122
Other	4,83	0,408
NeN=	28	

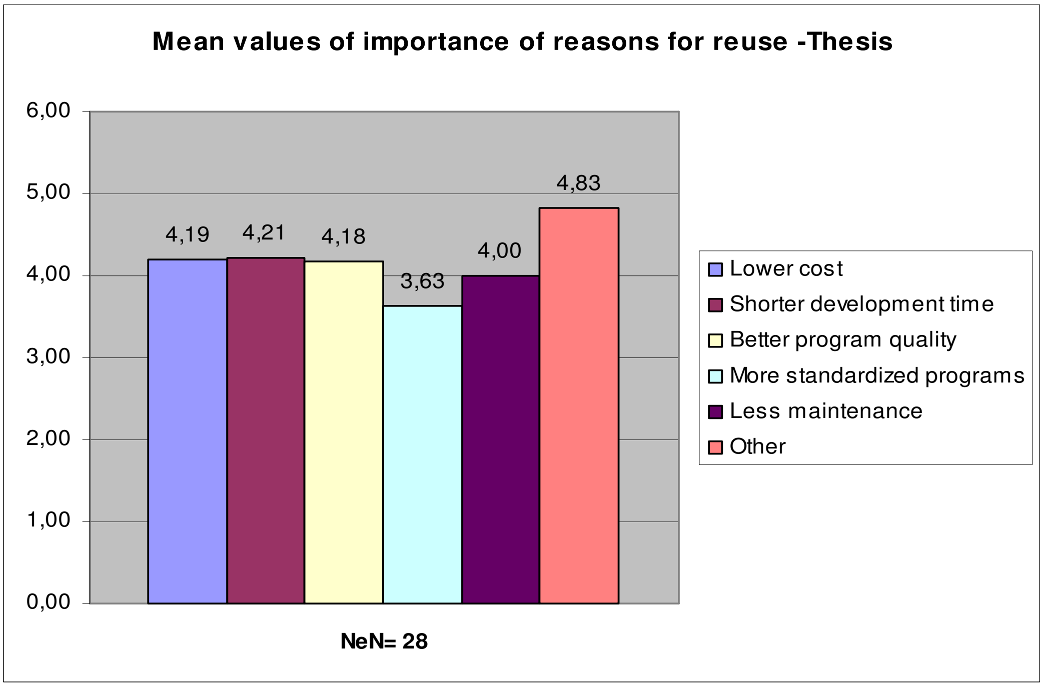


Figure 31: Mean values of importance of reasons for reuse -Thesis

For later comparison the mean values of importance of reasons for the companies, grouped by size, are shown in Table 39 and Figure 32.

Table 39: Mean values of importance of reasons, grouped by company size -Thesis

	Mean			
	0-19	20-99	≥100	All
Lower cost	4,17	4,23	4,13	4,19
Shorter development time	4,33	4,00	4,50	4,21
Better program quality	4,33	4,00	4,38	4,18
More standardized programs	3,50	3,92	3,25	3,63
Less maintenance	4,50	3,71	4,13	4,00
Other	0,00	4,00	5,00	4,83
N=	6	14	8	28

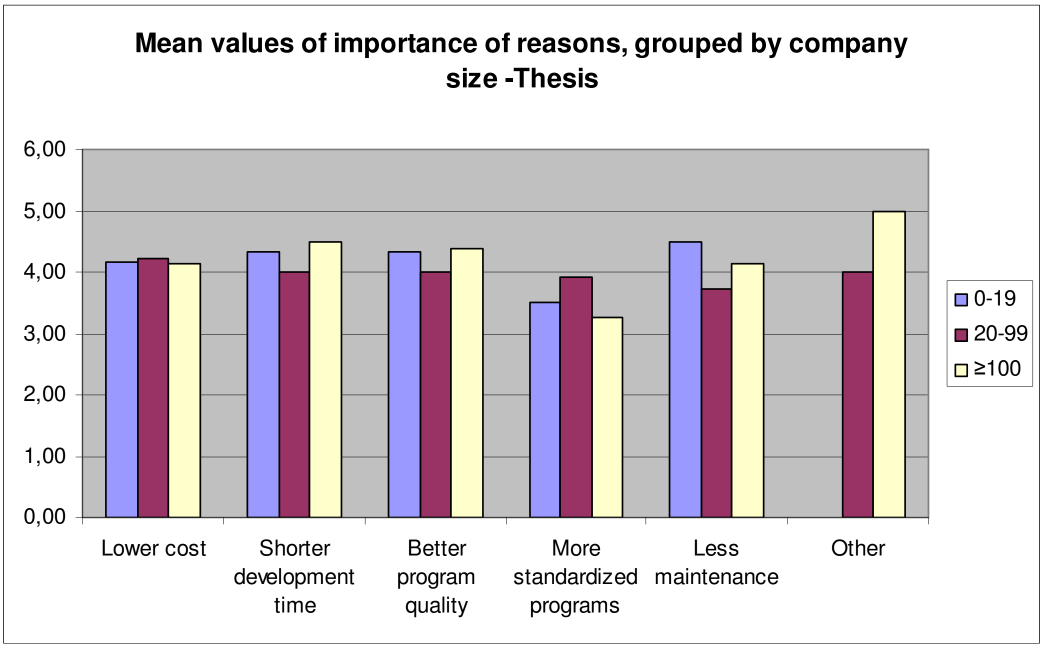


Figure 32: Mean values of importance of reasons, grouped by company size -Thesis

Next the subjects were asked how important they considered some success factors were to their companies (question QT20). The scores are as mentioned the same as in the previous question. Only the mean values are presented. The values are shown in Table 40, and presented graphically in Figure 33 and Figure 34. Because there was so much information on this question the graphical presentation has been divided into 2 parts. The differences between the factors are greater than in the previous question.

The success factor that the subjects see as the most important one is that the architecture is created for reuse (4,37). If this isn't the case the reuse will not be thought well through and minimal at best. Version-control and that the developers see the benefits that reuse gives are almost as important (4,27 and 4,26). These are fundamental parts that have to be addressed, because if the version control doesn't work finding the right components will be a problem, and if the developers doesn't see the benefits they won't use it and then there won't be any reuse. That the components are generalized for reuse also has a mean value over 4 (very important). It is very important because if they aren't there has to be put a lot of work into making them usable in a reuse situation, and the cost can be quite large.

The next reasons that are seen as pretty important are that the company has an internal development process where reuse is a point (3,85), that there is good documentation and availability of components in the reuse library (3,81) and that the leaders see the benefits of reuse (3,78). This means that the subjects think that there has to be a universal development process that describes how reuse should be used, not only to create reusable components, but also how to use them. Having a good reuse library that works and is well updated is important. Also if the leaders don't see the benefits they won't encourage to apply reuse and it won't be used as much as it could be. That the employees are trained in reuse is also important (3,48), because if they don't know how to use it, no one will, and the ones who use it will use it differently.

To have rewards for good reuse is however not seen as an important success factor among the subjects. This had a mean value of 2,69. This is interesting as there are a lot of different opinions about this. It is in many circles believed that this is an important factor, in the survey however it isn't.

To have own personnel dedicated to reuse is viewed as an even less important success factor with a mean value of 2,19. This has also been believed to be an important success factor, the reason why this has changed is maybe because reuse libraries has become so good that own personnel isn't needed. Also the reuse education can have rendered it useless.

The standard deviations are also here quite small, which means that the spread of the answers isn't that big.

Table 40: Importance of reuse success factors -Thesis

	Mean	SD
The company has an internal development process where reuse is a point	3,85	1,01
The developers must see the benefits of reuse	4,26	0,66
The employees are trained in reuse	3,48	0,89
Own personnel dedicated to reuse	2,19	1,27
Good reuse is rewarded	2,69	1,23
The leaders must see the benefits of reuse	3,78	0,93
The architecture must be created for reuse	4,37	0,63
The components are generalized for reuse	4,08	0,84
Good documentation and availability of components in reuse library	3,81	1,1
Version-control	4,27	0,67
Other	0,00	0
NeN=	28	

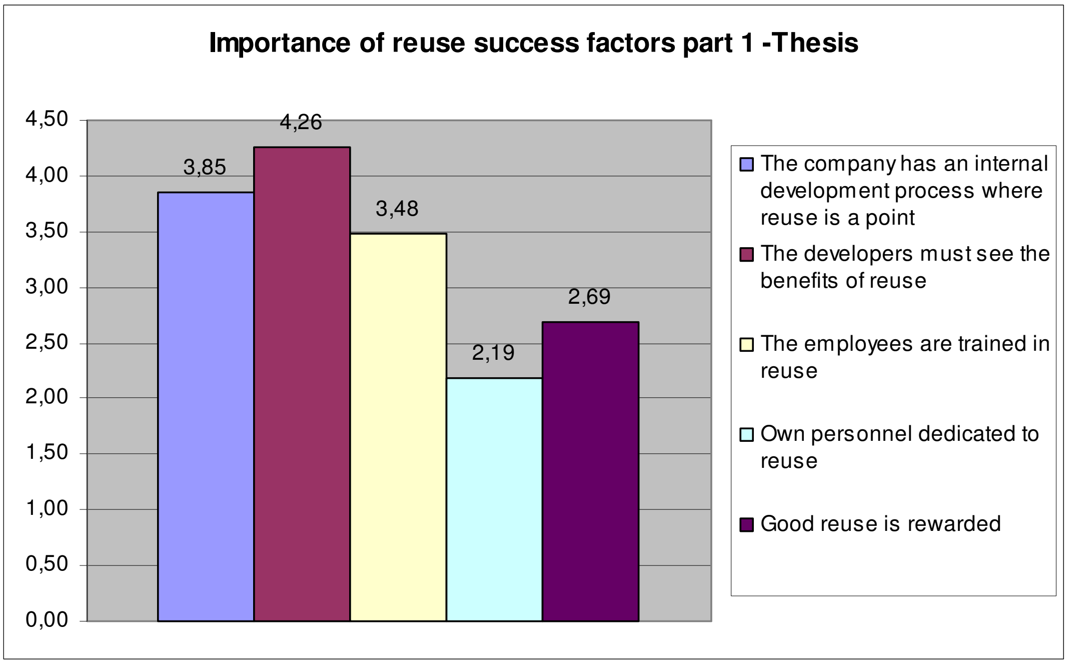


Figure 33: Importance of reuse success factors, part 1 -Thesis

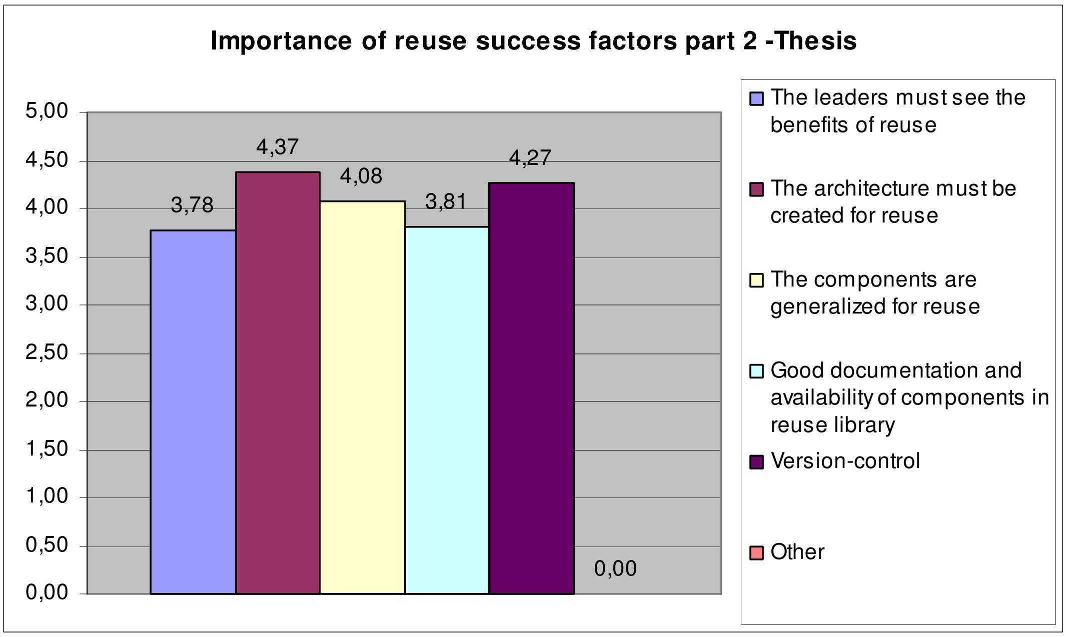


Figure 34: Importance of reuse success factors, part 2 -Thesis

For later comparison the importance of success factors, grouped by company size, is shown in Table 41, Figure 35 and Figure 36.

Table 41: Importance of reuse success factors, grouped by company size -Thesis

	Mean			
	0-19	20-99	≥100	All
The company has an int. dev. proc. where reuse is a point	3,17	3,93	4,33	3,85
The developers must see the benefits of reuse	4,00	4,36	4,29	4,26
The employees are trained in reuse	3,17	3,64	3,43	3,48
Own personnel dedicated to reuse	1,83	2,15	2,57	2,19
Good reuse is rewarded	2,17	2,85	2,86	2,69
The leaders must see the benefits of reuse	3,83	3,64	4,00	3,78
The architecture must be created for reuse	4,33	4,29	4,57	4,37
The components are generalized for reuse	4,00	4,08	4,14	4,08
Good doc. and availability of components in reuse library	3,83	3,62	4,14	3,81
Version-control	4,67	4,08	4,29	4,27
Other	0,00	0,00	0,00	0,00
NeN=	6	14	8	28

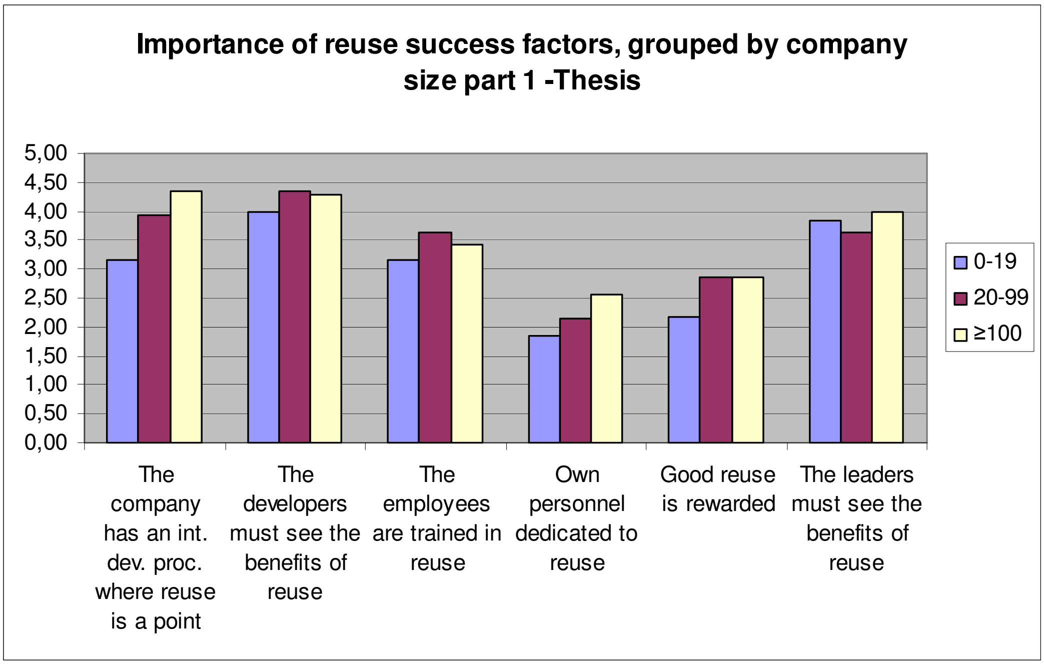


Figure 35: Mean values of importance of reuse success factors, grouped by company size part 1 -Thesis

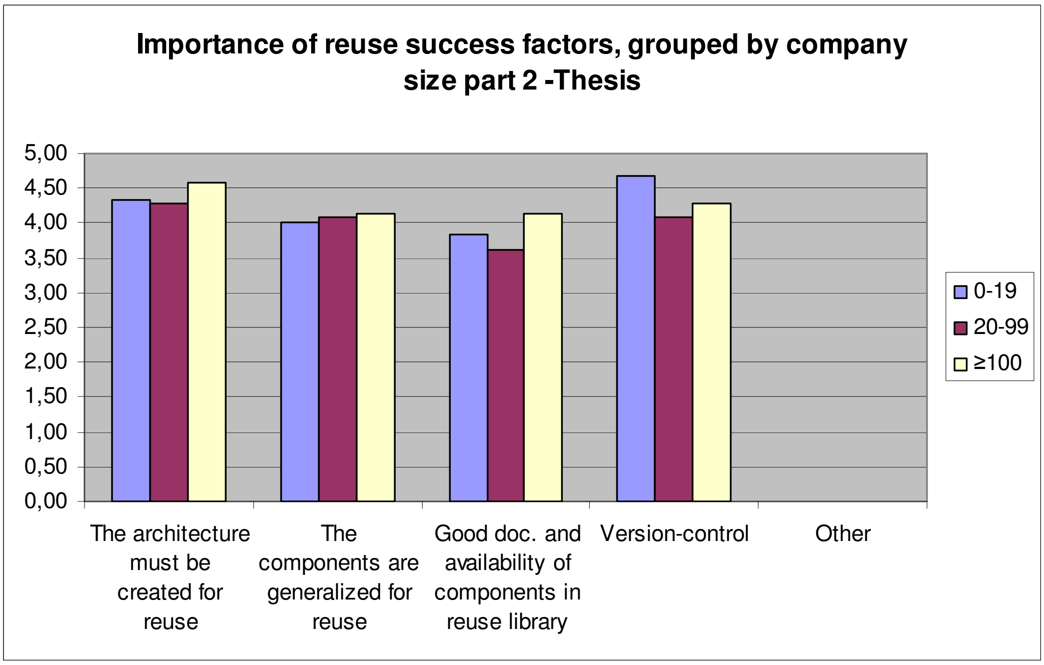


Figure 36: Mean values of importance of reuse success factors, grouped by company size part 2 -Thesis

4.2.4 The use of Product Families

The third part of the thesis questionnaire was about product families. Like the reuse part there was first a question on whether or not the companies used product families (question QT21), if they answered no on this question the rest of the questions in this part wouldn't apply to them. Of the 32 subjects asked 17 (53 %) answered that their companies used product families in their development, while 15 (47 %) answered that they didn't. This is shown graphically in Figure 37.

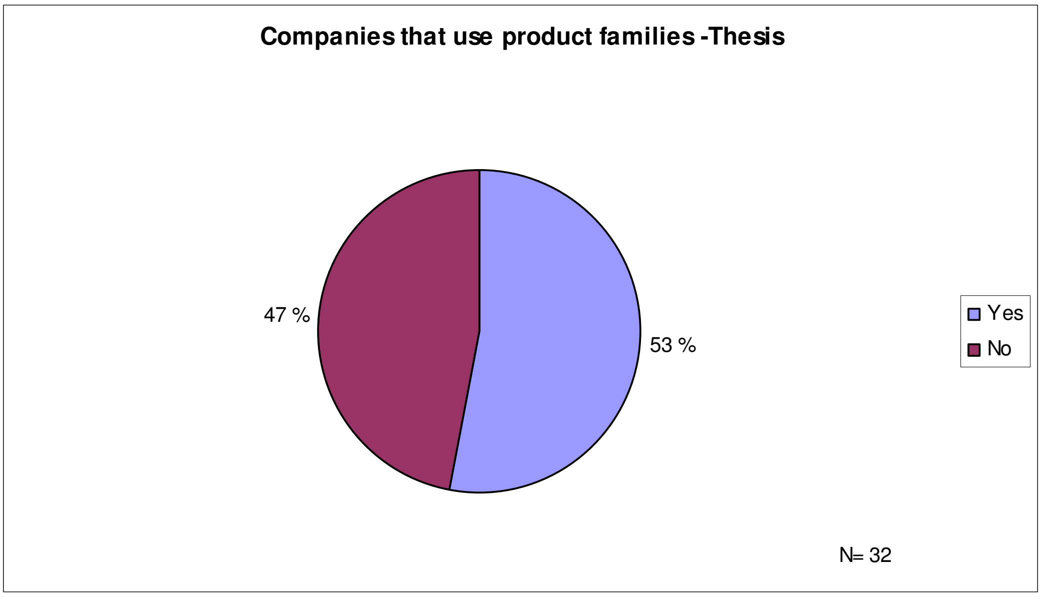


Figure 37: Companies that use product families -Thesis

The rest of this part of the survey was filled out by the 17 subjects that answered yes on the use of product families.

The subjects were then asked how long product families had been used in their companies (question QT22). The alternatives they could answer were below 1 year, 1-2 years, 3-4 years, 5-6 years and above 6 years. The answers from this question are summed up in Table 42. A vast majority of the companies that used product families in their development process had used it for more than 6 years (65 %). This shows that it's the old companies that use it.

Table 42: How long product families has been used -Thesis

<1 years	0	0 %
1-2 years	0	0 %
3-4 years	4	24 %
5-6 years	2	12 %
>6 years	11	65 %
N=	17	

The time the product family approach has been used can also be seen, grouped by the different sizes of the companies, in Table 43. These data are for later comparisons.

Table 43: How long product families has been used, grouped by size -Thesis

0-19

<1 years	0	0 %
1-2 years	0	0 %
3-4 years	3	75 %
5-6 years	1	25 %
>6 years	0	0 %
N=	4	

20-99

<1 years	0	0 %
1-2 years	0	0 %
3-4 years	1	13 %
5-6 years	1	13 %
>6 years	6	75 %
N=	8	

≥100

<1 years	0	0 %
1-2 years	0	0 %
3-4 years	0	0 %
5-6 years	0	0 %
>6 years	5	100 %
N=	5	

Then the subjects were asked what were common between the products in their product families (question QT24). The alternatives given were:

- Branding (to use a common name on a family of products to get market recognition).
- Shared demands (the demands given to the systems in the family are equal).
- Shared architecture (the architecture of the systems in the family are equal).
- Shared code (the code of the systems in the family are equal).
- Shared infrastructure (the framework around the systems and the underlying platform for the systems in the family are equal).
- Shared test-system (the test system used to test the systems in the family are equal).
- Other.

Of the 17 subjects 13 (76%) answered branding and shared architecture were common between the products in their families. These are therefore the most commonly used in product families. But shared infrastructure (12 out of 17) and shared code (11 out of 17) are also used a lot. That the products shared requirements or shared test-system on the other hand was not used by many, only 7 (41%) out of the 17 subjects answered that this was used in their companies. The answers are shown in Table 44 and Figure 38.

Table 44: Shared artifacts in product families -Thesis

Branding	13	76 %
Shared requirements	7	41 %
Shared architecture	13	76 %
Shared code	11	65 %
Shared infrastructure	12	71 %
Shared test-system	7	41 %
Other	1	6 %
NeN=	17	

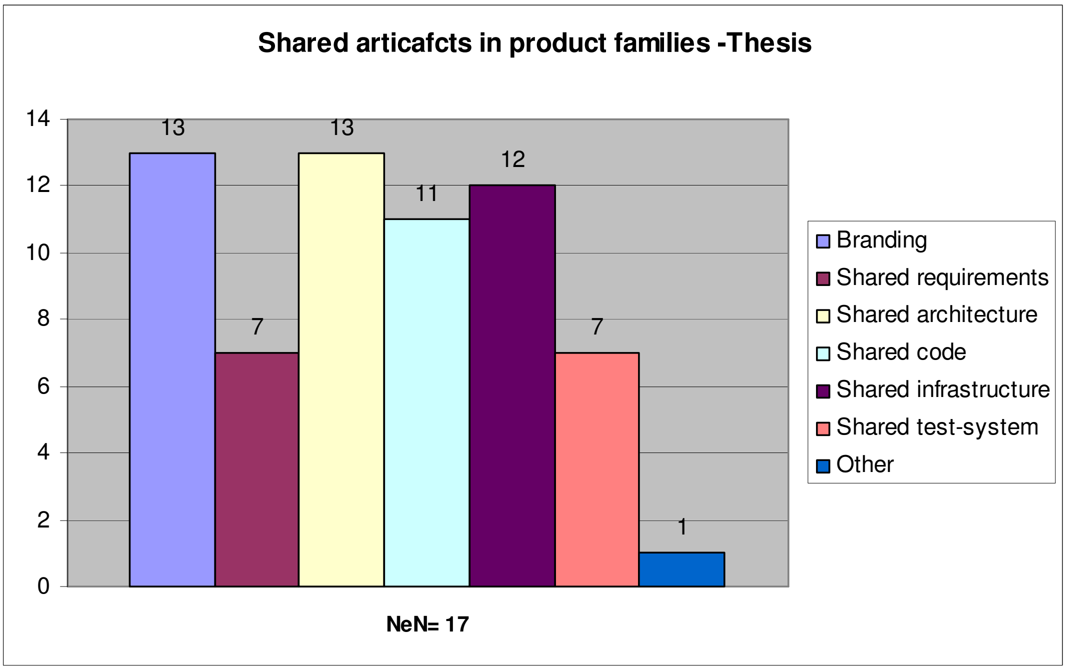


Figure 38: Shared artifacts in product families -Thesis

For later comparisons which artifacts are shared is in Table 45 and Figure 39 shown grouped by size.

Table 45: Shared artifacts in product families, grouped by size -Thesis

	0-19	20-99	≥100
Branding	75 %	88 %	60 %
Shared requirements	75 %	13 %	60 %
Shared architecture	75 %	75 %	80 %
Shared code	75 %	38 %	100 %
Shared infrastructure	100 %	63 %	60 %
Shared test-system	75 %	13 %	60 %
Other	0 %	13 %	0 %
NeN=	4	8	5

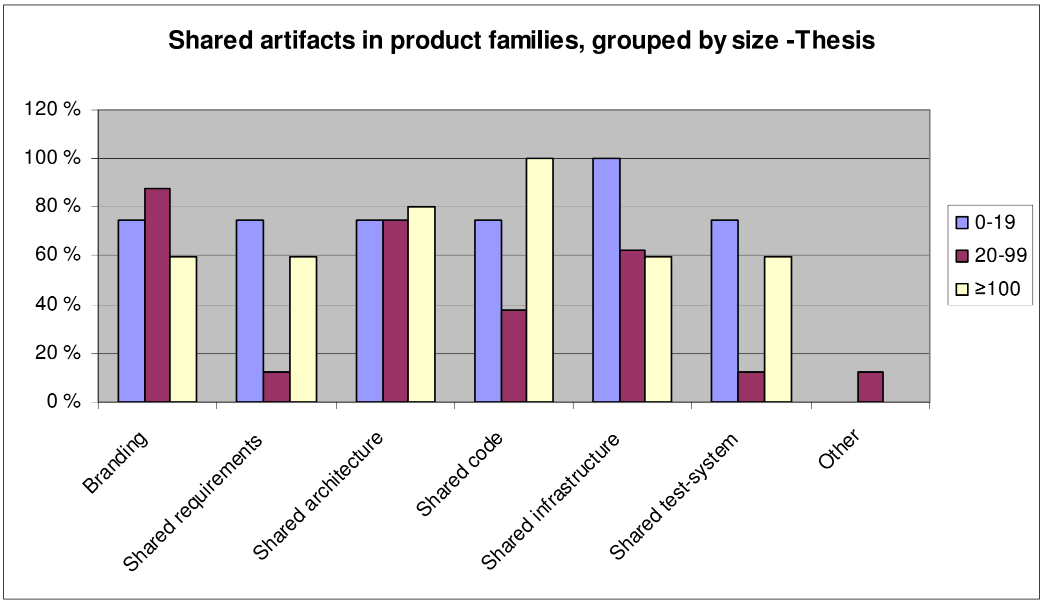


Figure 39: Shared artifacts in product families, grouped by size -Thesis

The next question was on how the companies started their product lines (question QT25). There are different ways of doing this, and the alternatives given were:

- Before (proactive product family, the family is created first and then the systems in the family are created).
- After (reactive product family, old systems are reused to create the family).
- During/Incremental (during the development of a system the family is created).
- Don't know.

This is an interesting questions, as it shows if the companies plan making product families before they start working, or if it something that comes later in the process. 8 of the subjects (47 %) answered that they started a family before they created the first product. This usually means a lot of expenses in the early stages, but often makes for better families later on. 5 subjects (29 %) answered that they created the family on basis from old systems. This is initially a cheaper way to do it, but can sometimes become more expensive because the existing systems often have to have a lot of changes done to them. 8 subjects (47 %) answered that families were created during system development. This often means a good system, because the systems can then be created so that they will fit in the family, and that its components can be reused later in it. The answers are shown in Table 46 and Figure 40.

Table 46: When product families are started -Thesis

Before	8	47 %
After	5	29 %
During/Incremental	8	47 %
Don't know	1	6 %
NeN=	17	

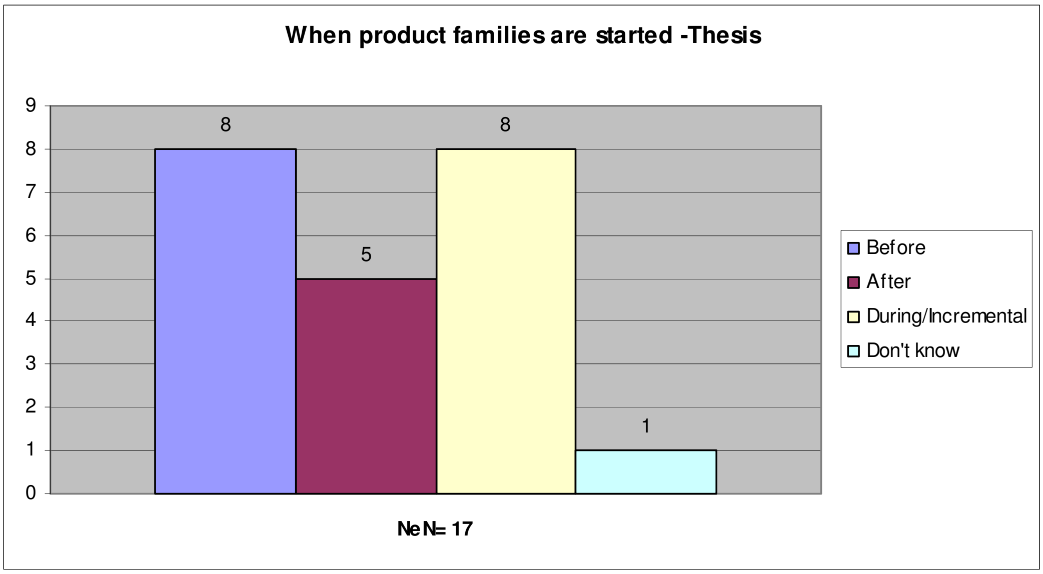


Figure 40: When product families are started -Thesis

For comparison later in the thesis how product families, grouped by size, are started is shown in Table 47 and Figure 41.

Table 47: When product families are started, grouped by size -Thesis

	0-19	20-99	≥100
Before	50 %	50 %	40 %
After	0 %	38 %	40 %
During/Incremental	50 %	50 %	40 %
Don't know	0 %	13 %	0 %
NeN=	4	8	5

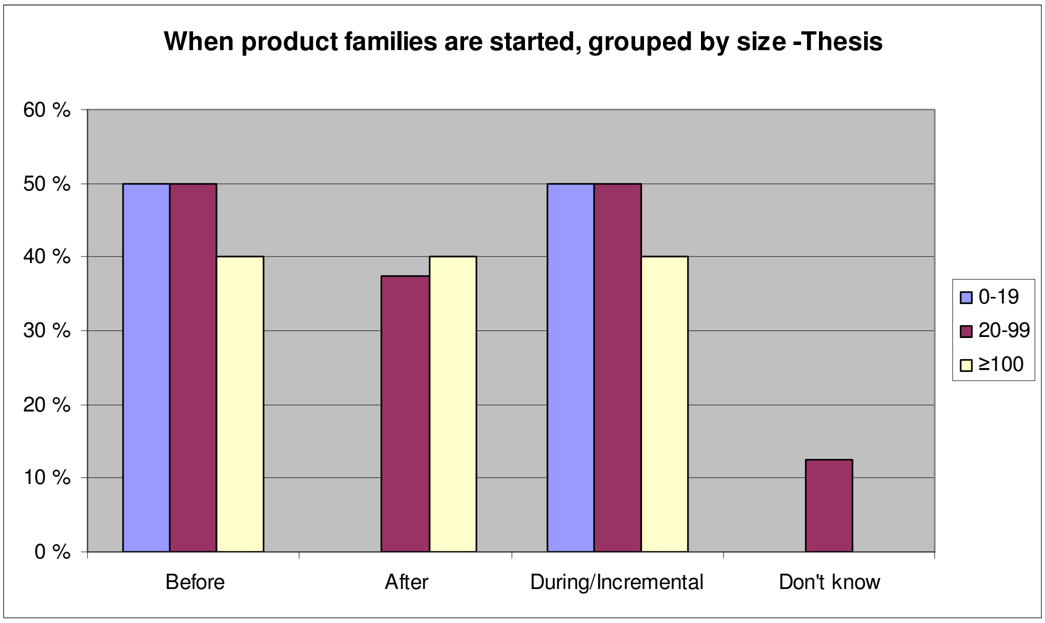


Figure 41: When product families are started, grouped by size -Thesis

Question QT26 asked how the different products in a family were distinguished from each other. This means how the products that are placed together in a family are different. The alternatives for this question were:

- Adapted with parameter altering (the products in the family are the same, but how they are configured is different).
- Adapted with addware (there is a core that is equal for all of the products in a family, but extra software is added to create different/extra functionality).
- Common framework/architecture, adapted modules (the framework/architecture are the same for all the systems in a family, but the modules that the system is put together from has to be altered/configured).
- Common modules, adapted framework/architecture (the modules that are used in the systems are the same, but the framework/architecture for the systems has to be altered/configured).
- Don't know.

Adapted with parameter-altering was the most used way to distinguish/put together a product family. 10 of the subjects (59 %) answered that this was used in their companies. To have a common framework/architecture, but adapted modules was the second most used with 8 (47 %) answers, and adapted with addware got almost as many with 7 (41 %). To have common modules, but to adapt the framework/architecture of how to put the system together wasn't used much in the companies, only 5 subjects (29 %) answered this. This is borderline reuse, and it isn't a common way to create a product family. 2 of the subjects (12 %) didn't know how this was done in their companies. The answers are shown in Table 48 and Figure 42.

Table 48: How products in a family are distinguished from each other -Thesis

Adapted with parameter-altering	10	59 %
Adapted with addware	7	41 %
Common framework/architecture, adapted modules	8	47 %
Common modules, adapted framework/architecture	5	29 %
Don't know	2	12 %
NeN=	17	

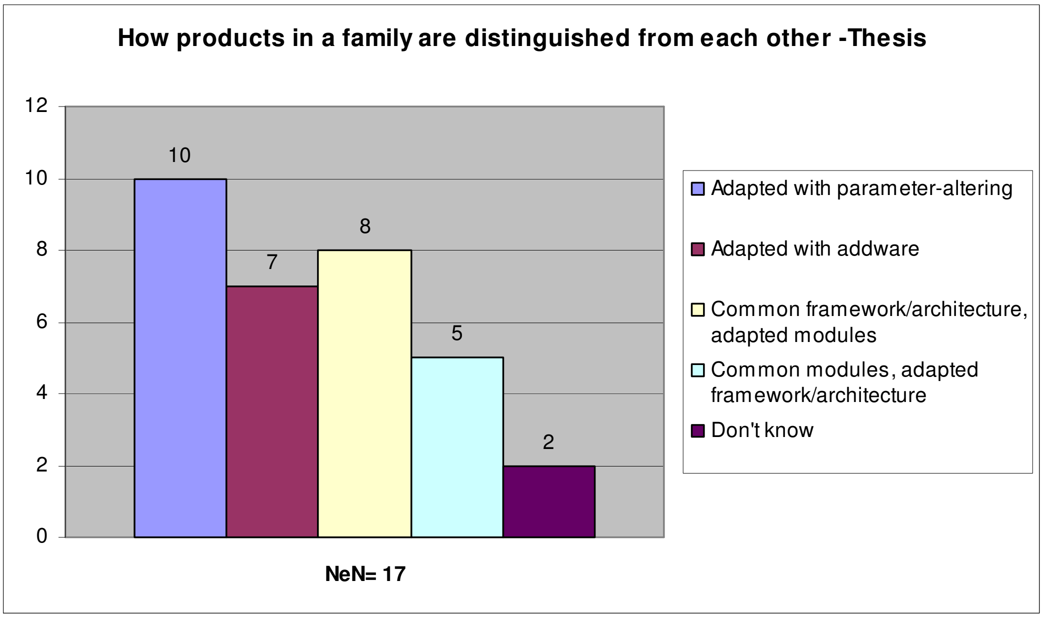


Figure 42: How products in a family are distinguished from each other -Thesis

These numbers are also shown, grouped by size, in Table 49 and Figure 43 for later analysis.

Table 49: How products are distinguished from each other, grouped by size -Thesis

	0-19	20-99	≥100
Adapted with parameter-adjusting	50 %	63 %	60 %
Adapted with addware	50 %	38 %	40 %
Common framework/architecture, adapted modules	25 %	63 %	40 %
Common modules, adapted framework/architecture	25 %	38 %	20 %
Don't know	0 %	13 %	20 %
NeN=	4	8	5

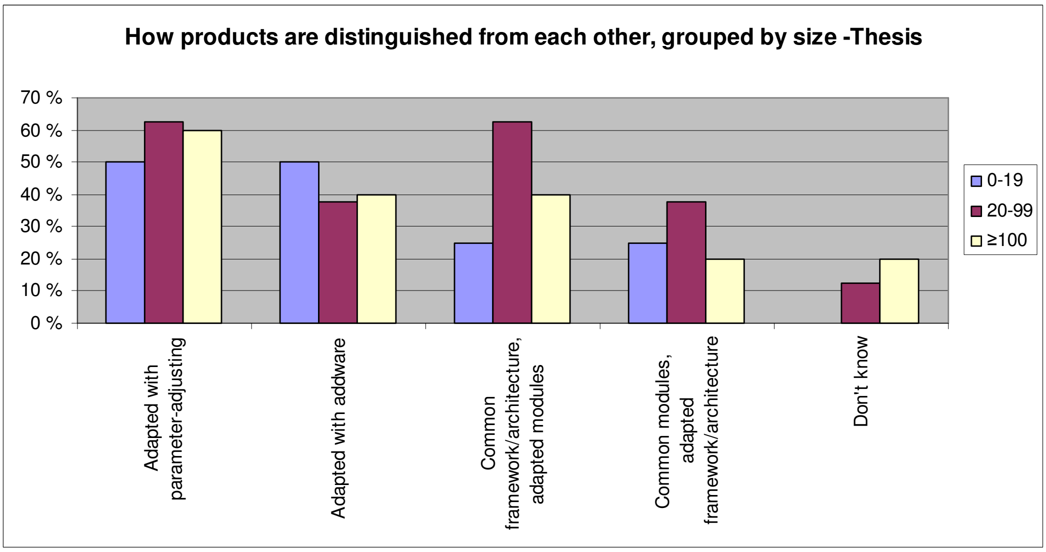


Figure 43: How products are distinguished from each other, grouped by size -Thesis

The next two questions (questions QT27 and QT28) were meant more for getting a good overview of how the companies used product lines. The information extracted isn't possible to do any analysis on as it is very widespread. On the first of these questions the subjects were asked how many different product lines their companies had. This got a lot of different answers ranging from 1 to the most that had 12-15 product lines. The answers can be seen in Appendix C.

The second of these question asked how many products the companies typically had per family. Also this got a lot of different question ranging from 2 to 15. The answers can be seen in Appendix C.

The last question in the product family part asked for the reasons why product families were used in the companies (question QT29). The following alternatives were given:

- Branding (to use a common name on a family of products to get market recognition).
- Defined from different customer groups (different customer groups has their own family).
- Defined from different standards (the family is created so that the products in the family follow a given standard).
- Follows a family of hardware products (the company produces a family of hardware products and the software systems follows this family).
- Other.

The most frequently used reason for product families is branding with 9 answers (53 %). This doesn't require much similarity between the products, but can offer a substantial benefit if the name of the product family gets a good reputation in the market. Defined from different customers and other reasons both got 7 answers (41 %). To have different families for the different customers was something a lot of the companies in the survey did. There were also a lot of other reasons for using product families, among them that companies used it to get more and better reuse. 2 of the subjects (12 %) answered that the product families followed a hardware family. Because these companies were mainly into hardware this was a good approach for them. None of the subject on the other hand answered that they're companies did it because the families followed different standards. The answers are shown in Table 50 and Figure 44.

Table 50: Reasons for using product families -Thesis

Branding	9	53 %
Defined from different customers	7	41 %
Defined from different standards	0	0 %
Follows a family of hardware products	2	12 %
Other	7	41 %
N=	17	

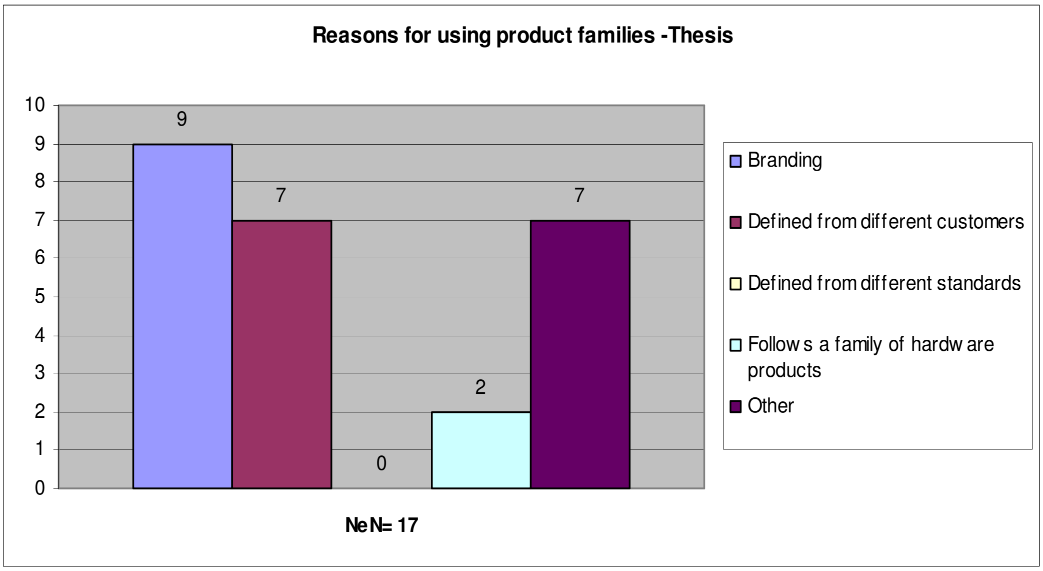


Figure 44: Reasons for using product families -Thesis

In Table 51 and Figure 45 the answers can also be seen grouped by size. This is for later comparisons.

Table 51: Reasons for using product families, grouped by size -Thesis

	0-19	20-99	≥100
Branding	25 %	75 %	40 %
Defined from different customers	25 %	50 %	40 %
Defined from different standards	0 %	0 %	0 %
Follows a family of hardware products	0 %	25 %	0 %
Other	25 %	38 %	60 %
N=	4	8	5

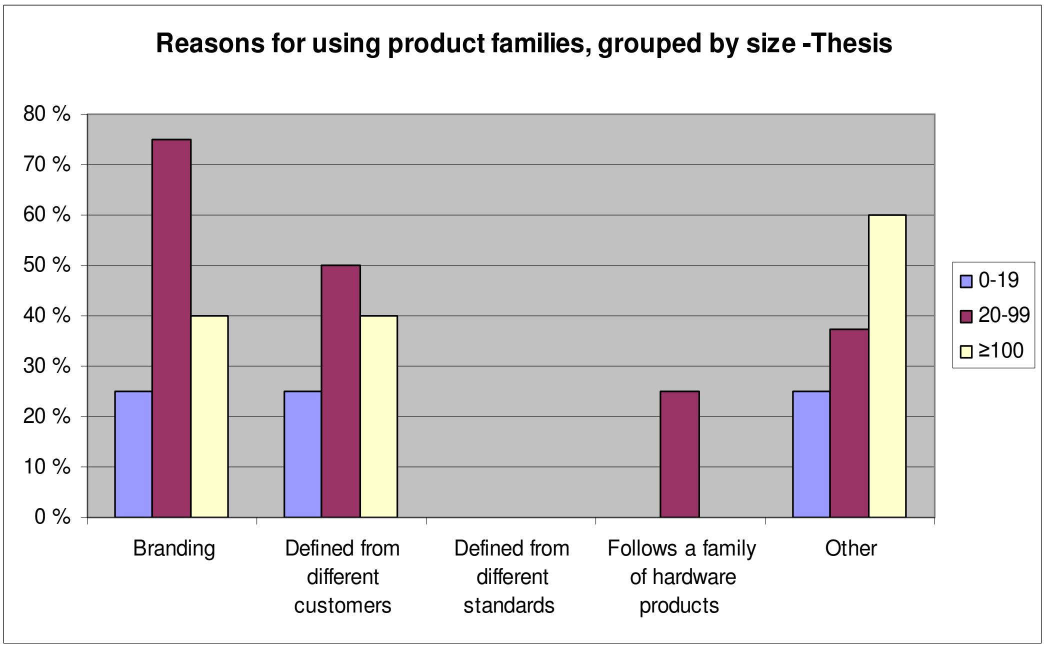


Figure 45: Reasons for using product families, grouped by size -Thesis

5 Discussion

In this chapter the collected data is discussed. The two surveys are discussed against each other, the research questions are answered and the answers are compared with related research. The scale from chapter 4.2.3 Systematic Reuse is used for scores and mean values in this chapter. The scale is as follows; 1- Not important, 2- A little important, 3- Important, 4- Very important and 5- Extremely important. Also in this chapter the question numbers from the questionnaires are used, question QT1 – QT29 are from the thesis survey and can be seen in Appendix A. Questions QN7.2 - QN7.16 are from the IKT-Norge survey and can be seen in Appendix B.

5.1 Comparison between the Two Surveys

The two surveys had a lot of differences since they were run by two different groups/persons, and not for exactly the same purpose. But they also have some similarities that are discussed here.

5.1.1 Applying Reuse

The reuse part constitutes the main part of the extra part of the IKT-Norge survey and one of the main parts of the thesis survey. The first question looked at is if the companies applied systematic reuse or not. This was question QN7.11 in the IKT-Norge questionnaire and question QT12 in the thesis questionnaire. The questions were identical, and just asked if it was used or not. In the IKT-Norge survey 92 % of the subjects answered that it was used, while only 8 % answered no. In the thesis survey 87 % answered yes, while 13 % answered no. These numbers are almost the same, if there were more subjects in the thesis survey these numbers probably would have been equal.

Reuse of COTS/OSS components.
Question QN7.8 in the IKT-Norge questionnaire and question QT16 in the thesis questionnaire asked which types of components were reused in the companies. The answers for these two questions are summed up in Table 11 and Table 31. The part of companies who used only COTS was very much alike (20 % in IKT-Norge and 18 % in thesis) also how many companies didn't use any of the mentioned components was almost equal (32 % in IKT-Norge and 39 % in thesis). The differences between the questionnaires however were between companies who used only OSS, and companies who use both OSS and COTS. In the IKT-Norge questionnaire 28 % of the subjects answered that their companies only used OSS components for reuse, while on the thesis questionnaire only 21 % answered the same. On the other hand in the IKT-Norge questionnaire only 20 % of the subjects answered yes on the use of both OSS and COTS, while on the thesis questionnaire 36 % answered the same. These numbers are shown graphically in Figure 46.

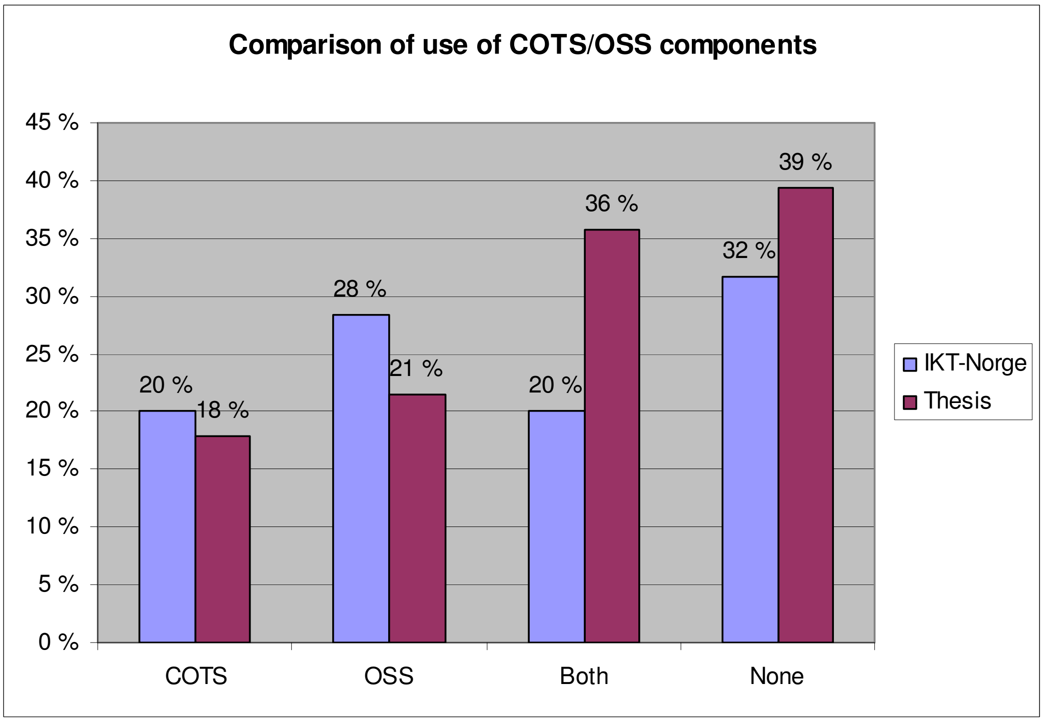


Figure 46: Comparison of use of COTS/OSS components

Even though the numbers are slightly different the differences aren't big enough to question them. The differences could come from something as simple as the subjects in the different surveys didn't completely understand the differences between OSS and COTS, or that the companies involved have very different sizes or have different business domains. The way the questions are asked are also pretty much the same, there was a slight difference because the IKT-Norge questions had yes or no questions on the use of OSS and COTS answer, while the thesis questionnaire only asked which were used.

What is reused.

In both questionnaires the subjects were asked what types of components were reused in their companies. There were two small differences between the questions. First the thesis questionnaire had one more point than the IKT-Norge questionnaire as it asked if process models/project plans were reused. Also in the IKT-Norge questionnaire the subjects were asked to answer yes or no on all of the alternatives, while in the thesis questionnaire the subjects only answered which alternatives they used. This could perhaps make the subjects think more through the alternatives before they answered in the IKT-Norge questionnaire.

In Figure 47 the answers from the two questions (question QN7.12 and question QT17) are summed up. There are no big differences between the two surveys, except maybe test plans (which got 58 % on the IKT-Norge and 36 % on the thesis) and documentation (which got 58 % on the IKT-Norge and 46 % on the thesis).

Again these differences are small, and are probably because the differences in selecting the companies and the number of companies in the studies.

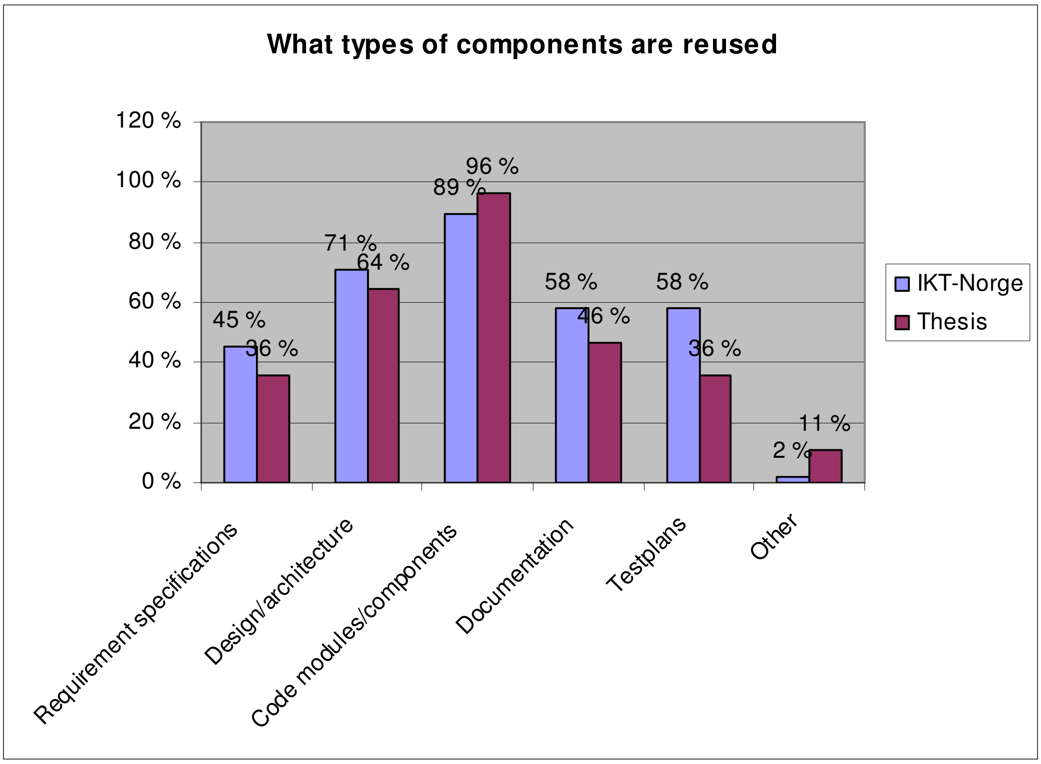


Figure 47: What types of components are reused

As discussed there are some differences between the answers in the two surveys. It is however difficult to understand why these numbers are so different, but they are probably from differences between the companies that submitted to the two surveys.

5.2 Research Questions

In this subchapter the research questions are answered. As the research questions are very wide they are divided into smaller parts and questions that are addressed individually. Some of these parts are answered by testing and by creating hypothesis that are tested, while others are only discussed as there are no numbers to perform tests on. Only the data from the thesis survey is used to answer the research questions.

5.2.1 RQ1: What is the Difference between how Reuse is done in Small, Medium and Large Companies?

To answer this research question it is divided into four parts which are answered individually and then summed up. The parts are:

- **RQ1-1: How well reuse works.** This is answered by defining a hypothesis that is then tested.
- **RQ1-2: Importance of reasons for reuse.** This is answered by running statistical tests on the data which is then discussed.
- **RQ1-3: Success factors.** Also this is answered by running statistical tests on the data which is then discussed.

- **RQ1-4: Sharing of components.** This is only discussed as there were no numbers to run tests on

Of the 32 subjects 28 answered that they applied reuse in their development process. Of these 6 were small companies (1-19 employees), 14 were medium companies (20-99 employees) and 8 were large companies (≥100 employees).

RQ1-1: How well reuse works.

The first part is to find out if there is any difference between how well reuse works in companies of different sizes. The numbers used here can be seen in Table 25. To investigate this a zero hypothesis and an alternative hypothesis have been created:

H_0: There is no difference between how well reuse works in the companies of different sizes ($p \geq 0.05$).

H_1: There is a difference between how well reuse work in the companies of different sizes ($p < 0.05$).

To test the hypothesis an ANOVA test is run on the results from the question of how well the reuse worked in the companies, grouped by company size (question QT14). The results from the test are shown in Table 52.

Table 52: ANOVA results on reuse in different sizes of companies

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
0-19	6	21	3,5	1,1
20-99	13	46	3,54	0,45
≥100	8	27	3,38	0,55

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F critical*
Between Groups	0,134972	2	0,067486	0,110892	0,895492	3,402826
Within Groups	14,60577	24	0,608574			
Total	14,74074	26				

Based on the values from the test H_0 can't be rejected. The P-value, which should be less than 0.05, is 0.895. Also the calculated F value (0,1109) is smaller than F critical (3,402), and therefore at the 0.05 level of significance, the differences between the 3 groups are statistically insignificant.

RQ1-2: Importance of reasons for reuse.

How the companies of different sizes rated the importance of reasons for reuse is then addressed. This was asked about in question QT19. The alternative other isn't taken into consideration when comparing, as the answers here are different from group to group.

The answers to this question can be seen, grouped by company sized, in Table 39 and Figure 32. As seen from the graphical presentation the differences between the groups are very small, except perhaps for more standardized programs, and less maintenance. To find out if the differences are significant ANOVA tests were run on each of the alternatives. The key numbers from this test are shown in Table 53.

Table 53: Key numbers from ANOVA test on importance of reasons for reuse

	F	*P-value*	*F critical*
Lower cost	0,03	0,97	3,40
Shorter development time	0,89	0,42	3,39
Better program quality	0,48	0,62	3,39
More standardized programs	1,43	0,26	3,40
Less maintenance	1,11	0,35	3,39

As mentioned before the value of F has to be bigger than the value of F critical to have a significant difference, also the P-value should be under 0,05 to make a decision that there is a significant difference. None of the numbers are even close, so therefore there isn't any difference between the importances of reasons for reuse between the companies of different sizes.

RQ1-3: Reuse success factors.

Next is the comparison of the importance of success factors grouped by company size. The numbers discussed here are shown in Table 41, Figure 35 and Figure 36. The large differences here are between the smallest companies (0-19) and the other two groups (20-99 and ≥100) which are similar on most factors. Especially on the company has an internal development process where reuse is a point, and good reuse is rewarded. To better see the differences ANOVA tests were run on all of the alternatives except other which no one answered. The key numbers from the ANOVA tests are shown in Table 54.

Table 54: Key numbers from ANOVA test on importance of reuse success factors

	F	*P-value*	*F crit.*
The company has an int. dev. proc. where reuse is a point	3,26	0,06	3,40
The developers must see the benefits of reuse	0,61	0,55	3,40
The employees are trained in reuse	0,59	0,56	3,40
Own personnel dedicated to reuse	0,47	0,63	3,40
Good reuse is rewarded	0,93	0,41	3,40
The leaders must see the benefits of reuse	0,34	0,72	3,40
The architecture must be created for reuse	0,47	0,63	3,40
The components are generalized for reuse	0,11	0,90	3,40
Good doc. and availability of components in reuse library	0,22	0,81	3,40
Version-control	0,81	0,46	3,40

Here the differences are large on some of the alternatives. That the company has an internal development process where reuse is a point almost have an F value bigger F critical, and has a P-value of 0,06. This is good enough to make a conclusion that there is a statistical significant difference. When observing the diagram it can be seen that this is more important for the larger companies than for the smaller ones. This is maybe because in small companies reuse doesn't have to be forced on the developers as they often all know each other and therefore share what they do among each other more than in the larger companies. The communication internal is better and therefore everyone knows what the other developers are doing. This is an advantage that larger companies don't have, as they are often spread out over different departments.

It is also apparent that there are a lot of other differences even though the numbers aren't big enough to make conclusions that there is a statistical significant difference.

The next alternative which had a large difference is to have own personnel dedicated to reuse. Also this is less important for smaller companies, and get more important as the companies grow in size. This isn't a very important alternative though, but it seems that the bigger companies think it's more important than the small ones. This could be because a bigger company will have a bigger reuse library, which then again leads to the need for someone to keep track of it. If not it will become difficult to use.

RQ1-4: Sharing of components.

The final part to explore with companies of different sizes is if there are any differences between how they share their components. The department based sharing data discussed here can be seen in Table 36 and Figure 29, while the company based sharing data can be seen in Table 37 and Figure 30.

The first point that is noticeable is that company based sharing is more used for small and middle sized companies (0-19 and 20-99), while it is less used for the large companies (≥100). This is especially for configuration systems and shared folders. This is because large companies often are divided into departments, and to keep the reuse library from becoming too vast they are often specific for departments. Also the bigger companies often have departments that are into completely different fields, and there is no point of sharing between them if they don't have anything in common that can be reused in other departments. The only exception is for intranet, which the biggest companies have more of for company based sharing than the small and middle sized companies.

For the department based sharing it is noticeable that to have an own database is used most by the smallest companies. As the databases don't have the built in functions that a configuration system has it is often used for easy sharing in companies who don't need all the fancy functions. Another aspect is that only the middle sized companies use intranet for sharing department based. The reason for this is unknown.

Even though not much in this research question can be proven statistically, there are a lot of factors that suggest that there are some differences between how the companies of different sizes apply reuse.

5.2.2 RQ2: What is the difference between how reuse is done internally in a Department and for the whole Company in bigger Companies?

To answer this research question it is divided into two parts that are answered individually, and then summed up. These parts are:

- **RQ2-1: How well reuse works.** This is answered by defining a hypothesis that is tested and discussed.
- **RQ2-2: Sharing of components.** This is only discussed, as there are no numbers to run tests on.

Of the 28 subjects that answered that their companies used reuse, 22 of these also answered that they also had department based reuse.

RQ2-1: How well reuse works.
The first part to be tested is if the subjects who worked in the companies with internal departments think there was a difference between how well reuse worked in their department and in the entire company. The following hypotheses are created for testing:

H_0: There is no difference between how well reuse works internally in the departments, and for the whole company ($p <= 0.05$).

H_1: There is a difference between how well reuse works internally in the departments, and for the whole company ($p > 0.05$).

When testing this hypothesis the answers from question QT14 (for the companies who answered they had departments) and QT15 are used. The data that is used can be seen in Table 26 and Table 29. Because the mean values of these two groups are the same, there is no point in running a t-test to find out if there is any difference between them. H_0 can obviously not be rejected, and there is therefore no statistically significant distinction between how well reuse works in departments and for the entire company for the companies who have departments.

RQ2-2: Sharing of components.
The next part to compare is if there are any differences between how components are shared in the departments and for the entire company, also here only the answers from the companies who answered they had departments are used. The data used is shown in Table 33 and Table 35. They are also shown graphically in Figure 48.

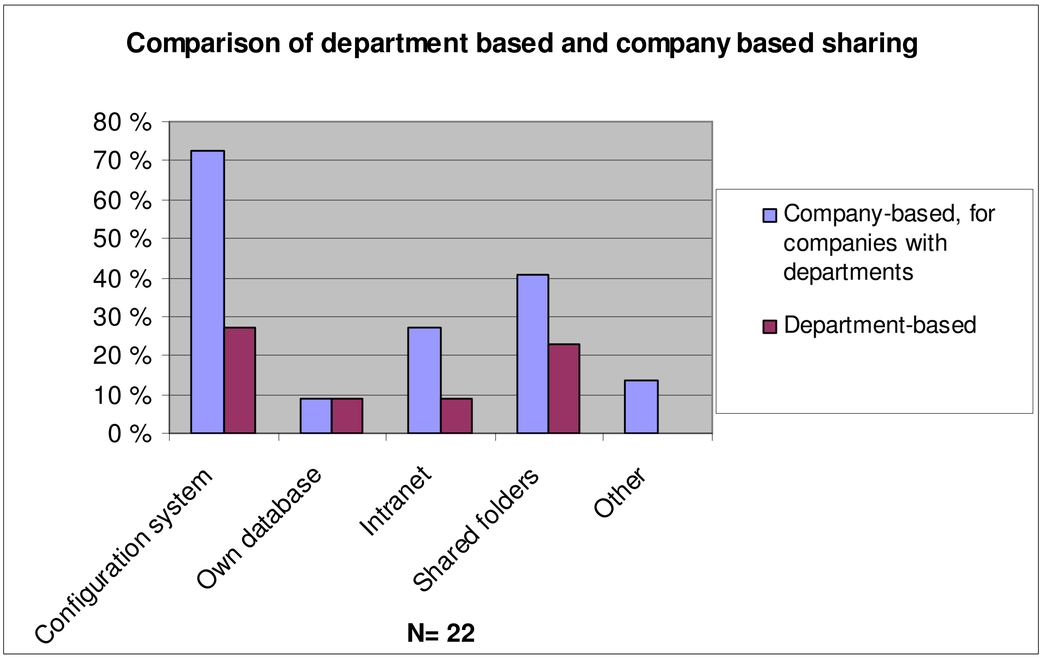


Figure 48: Comparison of department based and company based sharing

It is apparent that company based sharing is more used then department based sharing. This is probably because company based sharing gives bigger reuse libraries, which have a lot more components and therefore a higher chance of finding usable components. Also the costs for managing one big library are usually smaller than for managing many small ones. Especially configuration system, intranet and shared folders are used more for company based than department based. Own database is used equally for company and department based, but it isn't used much at all.

Therefore it is apparent that there is no difference between how well reuse works internally in the departments and for the whole company, but there is however some differences between how the components are shared.

5.2.3 RQ3: What is the difference between how Product Families are used in Small, Medium and Large Companies?

This research question is divided into five parts that are answered individually, and then summed up. All of the parts are however only discussed, none are tested as there are no numbers to run tests on. The parts are:

- **RQ3-1: How long product families have been used?**
- **RQ3-2: What artifacts are shared?**
- **RQ3-3: How product families are started.**
- **RQ3-4: Distinguishing product families from each other.**
- **RQ3-5: Reasons for using product families.**

In the thesis survey 17 of the subjects answered that their companies used product families. This is divided between 4 in the small companies (1-19), 8 in the medium companies (20-99) and 5 in the large companies (≥100). These numbers are small, especially for the small and big groups. Therefore it is hard to make conclusions with any certainty, but trends can be seen.

RQ3-1: How long product families have been used?
The first part to explore is how long product families has been used in the companies of different sizes which the subjects were asked about in question QT22, and is summed up, grouped by size, in Table 43. From the numbers it is apparent that the largest companies have all used product families for a long time. All of them had used it for more than 6 years. In the middle sized group most of the companies had used it for more than 6 years, but also one company for 5-6 years, and one for 3-4 years. In the smallest group however none of the companies had used it for more than 6 years. 3 of the companies had used it for 3-4 years, and one for 5-6 years. It is apparent that the larger the company is, the longer it had used product families. It looks like having product families in the smaller companies is a new trend, while the larger companies have used it for many years.

RQ3-2: What artifacts are shared?
What artifacts the different sizes of companies used is discussed here. The numbers discussed here can be seen in Table 45 and Figure 39. There are large differences between the sharing of many of the artifacts. How many percents share branding and architecture is very much alike. The small and the large companies are almost equal on how many percents share requirements, however almost none of the middle sized companies shared requirements in the families. Shared code and shared test-system is also very similar, some of the large and small companies shared them, but almost none of the middle ones did. The last of the shared artifacts in product families is shared infrastructure. All of the companies from the small group shared these, while 63 % of the middle and 60 % of the large sized companies did. The reasons for these numbers can be many, that the medium companies shared so little is peculiar. As mentioned before it is hard to conclude on these numbers since the number of answers in each group is so small.

RQ3-3: How product families are started.
Then how product families were started can be discussed. In Table 47 and Figure 41 these are shown grouped by size. The percentage of companies who started product families before the creation of a system, and also during the creation of a system is almost equal with 50 % for small and medium companies and 40 % for large companies. The difference is however for the percentage of companies that started product families after the creation of a system. While the medium and large companies did this (38 % and 40 %) none of the small companies started product lines after. This is maybe because small companies have to think more through what they create before they create it, they don't have a lot of resources to spend on creating a system that they don't know is going to become a family. If they are to make a family they plan ahead.

RQ3-4: Distinguishing product families from each other.
The next aspect that was explored in the survey was how products in a product family were distinguished from each other. The numbers discussed can be seen in Table 49 and Figure 43. Here the numbers are fairly equal for most of the alternatives. The one that stands most out is to have a common

framework/architecture and adapted modules. This was used a lot by medium sized companies (63 %) and large companies (40 %). The small companies however didn't use this that much (25 %).

RQ3-5: Reasons for using product families.

In the survey the subjects were also asked what their companies' reasons were for using product families, the answers are shown, grouped by size, in Table 51 and Figure 45. The first of the reasons was branding. Only one of the 4 subjects (25 %) from the small group answered that this was a reason for using product families in his company. This was used a lot more by the companies from the middle group (75 %), but not that much for the biggest companies (40 %). That this isn't a very important reason for the small companies isn't that unexpected. Usually they don't have many different programs and this is therefore not something that is seen the value in. However the reason why the bigger companies don't use this is unknown, but it could be because what the companies in the survey did. There were also only 5 that answered. Defined from different customers is fairly equal. None answered defined from different standards. Only the subjects that represented companies from the middle sized companies answered that follow a family of hardware products was a reason. This is probably because of the business domain that the companies represented were in.

There are a lot of differences between how the companies of different sizes use product families in the development process. As mentioned the number of answers are however small especially for the groups with small and large companies. This makes it hard to make solid conclusions. The trends that however can be seen are:

- The bigger the companies the longer they have used product families.
- The companies from the medium sized group are those that share least artifacts. The only exception is for branding, this is used most by the medium sized companies.
- When product families are started is pretty much the same for all of the groups, the exception is that none of the small companies started product families after the creation of a system which both the medium and large companies did.
- How product families are distinguished from each other is pretty much the same for the companies from the groups of different sizes, the only difference is that to have a common framework/architecture and adapted modules is used more by the medium sized companies, and less by the big and small companies.
- There were a lot of differences between the reasons for using product families. It is apparent that branding is very much used in the medium sized companies, while not so much in the small and large companies. The middle sized group was also the only group to have companies that used product families because it followed a family of hardware products.

5.3 Relating to other Research

Here the answers from the thesis survey are discussed against the other research which is presented in chapter 2.4 Possible Success factors. The two articles are discussed separately and then summed up.

5.3.1 "Sixteen Questions about Software Reuse"

In this article [FF95] some success factors are given, and also some factors that they concluded didn't create better reuse. To discuss this article it has been split into two parts, one where the different success factors are discussed, and one where the factors that don't give better are discussed, and they are summed up in the end. These parts are both discussed against the data from the thesis.

Success factors.
The first of the factors which is mentioned to give better reuse is type of industry. This is difficult to confirm or disconfirm with the thesis survey, as most of the companies were into multiple industries. However it is believed that this is a success factor, as it is easier to reuse for companies that are exclusively into software than for companies who are into consulting. This is because the systems they deliver are often very similar, while the consultants often are hired to do completely different jobs.

The second of the factors who gave better reuse was perceived economic feasibility. To get this to work it is important that the developers see the benefits of reuse, if not they will not use it. This was the rated as the third most important success factor by the subjects in the thesis survey, with a mean value of 4,26 (where 4 was very important, and 5 was extremely important). It is difficult to confirm this, but it can't be disconfirmed. It looks like this gives better reuse.

The third of the factors that the article states give better reuse was high quality assets. If the assets are believed to be of poor quality the developers will not use them, because they think that the work saved will eventually be very little as they will have to use extra time on the assets. In the thesis survey the subjects were asked to rate how important they considered it was that components were generalized for reuse, and to have good documentation and availability of components in the reuse library. These two factors help to ensure that the quality of the components is good, and that they can be reused without having to put too much effort into them. They both scored high on importance, the generalization of components got a mean value of 4,08, and good documentation and availability got a mean value of 4,27. It is therefore apparent that also the subjects in the thesis survey think that high quality assets are important for good reuse.

The next factor mentioned was to have a common software process. The article is however not sure if this factor gives better reuse, as the respondents say no, but it seems from levels of reuse that it does. In the thesis survey the subjects were asked if they thought that the company had an internal development process where reuse is a point was an important success factor. This got a mean value of 3,85, which is somewhere between important and very important. It is therefore safe to say that it is regarded as an important factor for getting reuse to work.

The final factor for better reuse mentioned is reuse education. This is also agreed on to some extent in the thesis survey. On question if whether or not it was important that the employees are trained in reuse the mean value was 3,48. This means that it is between important and very important. It is far from the most important factor, but it is nevertheless important.

Factors that don't give better reuse.

The article also concludes on some factors that the subjects didn't feel gave better reuse.

Not all of these factors can be compared with the thesis survey, as there isn't information about all of them from the survey. However some of them can.

The first of these factors were programming language. In the thesis this wasn't asked about as a factor, but in Table 27 and Figure 22 how well reuse work in the companies have been grouped together by what programming languages the subjects answered as the main ones for the companies. This isn't a very good measure as some companies answered one while others two or three main programming languages. However it is apparent that the differences between the different languages and how well reuse works is very small. Therefore the thesis supports that this isn't a factor that creates better reuse. However this could perhaps be looked further into.

The next of the factors is the use of recognition/awards for good reuse. In the thesis this got the next lowest with a mean of 2,69 (which is between important, and a little important). This supports what they found out, that this doesn't give better reuse.

To have a repository was also one of the factors mentioned that didn't influence good reuse. In the thesis survey it is hard to explore how this is today, since all of the companies who applied reuse had a repository. Today it has become common practice for all companies to have some sort of repository for storing the reusable components. The survey described in the article was however done in 91-92, and not everyone had repositories then. Because today everyone has repositories this factor has become outdated. In the thesis survey however to have good documentation and availability of components in reuse library is a important success factor with a mean value of 3,81. Therefore to have a good repository which is easy to navigate makes it easier to get good reuse than without having one.

The article argues that the size of the organization has no affect on good reuse or not. In the thesis survey it is tested if the subjects from different sized companies think that reuse works better than in the others. This is tested with a hypothesis test, and it is concluded that the null hypothesis can't be rejected and there is no difference between how well reuse works in the companies of different sizes. Therefore this is supported by the thesis.

Even though this is an old survey most of its claims still hold today. It is apparent that reuse haven't changed that much in the last 16 years, at least how the developers see it. The only factor which has become outdated is that repositories don’t automatically give good reuse. This is because all companies that apply reuse have some kind of repository today. Another aspect is that this survey was done on American companies, while the thesis survey was done on Norwegian companies. Also this has no influence on the result.

5.3.2 “Success and Failure Factors in Software Reuse”

In this article [MET02] a study of some reuse projects resulted in the authors finding some success and failure factors which here will be discussed against the thesis survey. The discussion has been divided into two parts, success factors, and failure factors. Under each of these the factors are discussed, and they are summed up in the end.

Success factors.
The first mentioned success factor the article describes is that smaller companies had the advantage of better communication. This isn’t explicitly looked into in the surveys, and it is therefore hard to conclude one way or the other. However some trends can be seen out from how the smaller and medium sized companies shared the assets different from the bigger companies. It is apparent from Table 37 and Figure 30 that configuration system and shared folders are more used company based for these types of companies than the large ones. This is maybe because the communication internal is better and this can therefore be used. However this can’t be concluded.

The final success factor is that roles dedicated to reuse were necessary. In the thesis survey the subjects were asked to rate how important they thought it was to have own personnel dedicated to reuse (question QT20). This was rated as the least important success factors by the subjects, and got a mean value of 2,19. It is hard to conclude one way or another, but it seems like this isn’t a very important success factor. At least the subjects that answered the thesis questionnaire don’t think it was.

Failure factors.
Then the article mentions some factors for failure. Not many of these can be addressed by the data collected in the surveys, but a couple can. Also in the survey the subjects were asked about success factors, and not failure factors. Therefore these are compared with the failure factors from the article.

The first failure factor that can be discussed is no deep management commitment. On question QT20 of the thesis survey the subjects were asked to rate how important they thought it was that the leaders saw the benefits of reuse. This scored a mean value of 3,70. This isn’t the most important success factor for the subjects, but it is however important (between important and very important). Therefore if this isn’t present it could be a failure factor.

The next failure factor is that there is no training/awareness action. In the thesis survey the subjects were also asked to rate how important they thought it was that the employees are trained in reuse. This was thought of as a fairly important factor with a mean value of 3,48. Another success factor that also can be seen as training/awareness is that the company has an internal development process where reuse is a point. This scored even higher with a mean value of 3,85. Therefore it seems that the thesis in a way supports that to have no training/awareness can be seen as failure factor.

In Table 55 which of the success factors given are supported by the thesis, [FF95] or [MET02] are presented. The ones that are supported are marked by S and the ones that aren't supported are marked with N. For the thesis the success factors that got a mean value over 3 are thought of as supported, while the ones that got below 3 are thought of as not supported. For the articles the ones that are mentioned as success factors are marked as supported, the rest aren't marked. There is also a column for the concluding if the success factor is supported or not.

Table 55: Which success factors are supported

	Thesis	[FF95]	[MET02]	Conclusion
The company has an internal development process where reuse is a point	S	S	S	S
The developers must see the benefits of reuse	S	S	-	S
The employees are trained in reuse	S	S	S	S
Own personnel dedicated to reuse	N	-	S	N
Good reuse is rewarded	N	-	-	N
The leaders must see the benefits of reuse	S	-	S	S
The architecture must be created for reuse	S	-	-	S
The components are generalized for reuse	S	S	-	S
Good documentation and availability of components in reuse library	S	S	-	S
Version-control	S	-	-	S

5.4 Possible Threats to Validity

This subsection discusses the different threats to validity for the thesis survey.

When conducting a survey it is important that the results are valid. If they aren't valid they can't be trusted to be used for concluding on, therefore there needs to be adequate validity to trust them. [WRH00] states that "adequate validity refers to that the results should be valid for the population of interest". This means that it can be trusted for the population which is investigated. Also it should be possible to generalize the results for a bigger population.

[WRH00] defines four different validities; conclusion validity, internal validity, construct validity and external validity. The definitions used are taken from [WRH00].

5.4.1 Conclusion Validity

This validity is concerned with generalizing the results to the concept or theory behind it. This means that it is concerned with if the statistical methods used are correct.

The two tests that are used to run tests in this thesis are ANOVA and t-test. These are well known and trusted statistical tests and there is nothing to suspect that they aren't trustworthy. In some of the research questions however there aren't any data that can be used to do these kinds of tests. Here the differences are only discussed. This is harder to conclude on, but it is nevertheless not impossible.

5.4.2 Internal Validity

This validity goes toward if the subjects have answered truthfully, and that it is the intended subject that has answered. Also that their motivation wasn't influenced by something else and that the questions weren't misunderstood. The validity is concerned with if the data is correct or not.

There is no reason to believe that the subjects didn't answer truthfully and also no reason to suspect that somebody else than the persons who said they answered were the one who answered. The subjects worked in different positions, but all of them had knowledge about how reuse and product families were applied, as well as the development of software in their company.

There was no reward for answering the questionnaire other than the promise of the master thesis. It is also unlikely to suspect that any of the subjects were forced to answer by anyone (supervisors etc.), it seems like they answered willingly.

The questions in the questionnaire were also tested thoroughly to make sure that they wouldn't be misunderstood. There are however two questions that had flaws. The first is when asking for number of employees in IT and in software development. It wasn't explained if the employees in software development also should be counted in IT. However since these numbers weren't used for anything this isn't regarded as a threat. The other was that when asking when product families were started the names for starting a product family before and after (proactive and reactive) had been swapped. Given the layout of the question which asked if it was started before or after, and only referred to the name, this is neither regarded as a large threat.

Another fault in the questionnaire is that question number 24 has been left out by error. This is however not regarded as a threat.

5.4.3 Construct Validity

Construct validity is interested in generalizing the result to the concept or the theory. This means finding out if there is any relation between theory and observation. It is concerned with if the research questions and the questions asked in the questionnaire are the right ones.

The research questions that were created are thought to answer what they are supposed to. They were mainly selected from the depth study [MS05], and then divided into sub-questions which were answered individually. The main ideas with the research questions were to check for differences between different groups, this was done without problems.

The questionnaire was created not only to answer the research questions, but also to understand how reuse and product families were used in the different companies. To check if what was regarded as important success factors earlier had changed or if the factors were the same. The questions in the questionnaire provided the information that was wanted from them.

One question that however could have a better way of being rated is if reuse is good or not. This is because there is no good way to define what good reuse is, it is often different what subjects regard as good reuse. However to have a good way of rating this is a problem, as it requires someone to go in and measure it.

One question that maybe is missing in the questionnaire is about failure factors, this would make it easier to compare with the related research as they had some failure factors.

5.4.4 External Validity

External validity is concerned with if the sample of the population can be generalized for the whole population and if the results of the study can be generalized outside the scope of the study. It is concerned with if the companies selected were the right ones.

The sample of companies wasn’t that large with 32 answers. But there is no reason to think that this has any affect on the quality of the answers. Also the answers that could be tested against the IKT-Norge survey were tested, and these were almost identical. Also the distribution of company sizes was good with 8 companies that had 1-19 employees, 16 companies that had 20-99 employees and 8 companies that had ≥100 companies. This made for a good comparison between the companies of different sizes.

For the survey a convenience sample was used. This was because it was hard to come in contact with people if there only was an info-e-mail address. Mails sent to this address have a very high probability of being overlooked and there is also no one that can be called in the company to check on the progress. Therefore a list with contacts internal in the companies was acquired. This list was however vast, and was also grouped after the same sizes as the thesis was interested in, this is the standard way of grouping companies after size today. The companies were also from different business domains. It is therefore believed that there weren’t any threats to validity from how the subjects were selected.

The companies in the survey were only Norwegian, if these apply reuse and product families differently from companies in other countries is unknown.

One possible problem is with the sizes of the groups of different company sizes when asking about product families. Here there only were 17 answers, where only 4 of these had 1-19 employees and 5 had ≥100 employees. These numbers are small when comparing the differences between the groups of different sizes. Therefore it is hard to make conclusions based on these answers.

There is therefore no reason to say that the sample can’t be generalized for the whole population, but the sample for comparing the different sizes of companies that use product families is maybe too small.

6 Conclusion and Further Work

In this chapter conclusions are made based on the surveys and some thoughts for further research are presented.

6.1 Conclusion

In the thesis a lot have been compared against each other; the two surveys, companies of different sizes, internal in a department and for the whole company, and the survey with related research. Even though there are a lot of similarities, there are also some differences.

The two surveys.
First the IKT-Norge survey and the thesis survey were compared. The answers were very similar, and this helps to conclude that the sample in the thesis survey was a good representation of the whole population, even though it was a convenience sample. This is based on the fact that the sample in the IKT-Norge survey was selected randomly from a bigger sample and was also larger (60 answers against 32 in the thesis survey).

6.1.1 Reuse

General conclusion.
From both theses it is apparent that systematic reuse is applied in almost all Norwegian companies, and a lot of different components are reused. It has been used for some time in most of the companies to. There were also a lot of different types of components that were reused. Most of the subjects that were asked felt that reuse worked ok or good in their companies. From the data it can also be concluded that what programming languages and development processes/methods were used had no influence on how well reuse worked.

The reasons why reuse was applied in the companies were different, but except for more standardized software, which the subjects felt was a little less important, the reasons scored almost the same. Other got a higher score than the predefined reasons, but this is because the subjects only think about the most important reasons while trying to define others.

The subjects also felt that many of the given success factors were important. The most important ones were that the architecture was created for reuse, to have good version control, and that the developers saw the benefits in reuse. Not all of the success factors were considered important. The ones that weren't were to have own personnel dedicated for reuse and that good reuse was rewarded.

Companies of different sizes.
In the thesis survey many of the answers were presented grouped by the sizes of the companies. This was to answer research questions. The answers in the IKT-Norge survey was unfortunately not possible to group after company size, and its answers could therefore not be used when discussing this.

From the data collected it can be concluded, with some certainty, that there isn't any difference between how well reuse works in companies of different sizes. There was a small difference between the mean values, as the subjects from the largest companies answered that it didn't work as well as in the small and medium companies. However this number was very small.

There is also no difference between what the companies of different sizes think is important reasons for applying reuse. It can be concluded that the reason they apply reuse are the same for the companies of different sizes.

There is however some differences between what the companies of different sizes regard as important success factors. When running ANOVA tests on the numbers this can however only be concluded on one of the success factors; that the company has an internal development process where reuse is a point. Some of the other factors got very different scores on importance, but can't be concluded on with ANOVA tests. How important it is that good reuse is rewarded is one of these. Even though it is hard to say which factors are different, it is safe to say that there are some differences between how the different sizes of companies rate what is important for good reuse.

There are also some differences between how companies of different sizes share components among their developers. One of them is that company based sharing is used more in small and middle sized companies. For department based sharing to have an own database dedicated to reuse, or an intranet is used most by the small and middle sized companies. It can therefore be concluded that there is a difference in how reusable components are shared in the different sizes of companies.

As a final conclusion it can be said that there are some differences between reuse in the companies of different sizes, but there are also a lot of similarities.

Internal in a department and for the whole company.

To explore which of the companies applied reuse differently internal in the departments, and for the whole company, the subjects were asked how they rated reuse both for the department they worked in, and for the whole company. If the reusable components were shared department based or company based was also asked about so differences could be explored.

On the data from the survey it can be concluded that there is no difference between how well reuse work internally in the departments and for the whole company. The mean values for these two, for the companies that had departments, were the same. Therefore reuse works just as well internally in the department and for the whole company.

There is however some differences between how the reusable components are shared internally in the departments and for the whole company. From the answers it is apparent that configuration systems are used much more company based than department based. Also intranet and shared folders are used more company based. It can therefore be concluded that there is a difference between how reusable components are shared.

6.1.2 Product Families

General conclusion.

From the results of the survey it can be concluded that product families is a popular way to apply reuse, most of the companies who use it has also used it for some time. There were a lot of different artifacts that were shared through product families, and a lot of companies also used it to promote products through branding. The most used way of distinguishing the different products in a product family from each other was by using parameter adjusting. This means that there is a ready made product that only has to be configured and sold to different customers.

The reasons for using product families were many and different, but when asked it was answered that the two most important reasons were those of branding and defined from different customers. This means that the most important reasons are for profiling towards customers.

Companies of different sizes.

The answers from this part of the survey were also grouped after size to explore if there were any differences between how the companies of different sizes used product families. The numbers of answers for the small and the large group were however small, and it is therefore hard to make solid conclusions.

The first difference that is noticeable between the companies of different sizes is how long product families have been used. The bigger the company was, the longer it had used product families. It can therefore be concluded that there is a difference between how long product families have been used in companies of different sizes.

It can also be concluded that there were a lot of differences between what artifacts were shared in the product families. These are discussed more thoroughly in the discussion chapter.

When the product families were started was very similar for the companies of different sizes. There was however one noticeable difference; none of the small companies who used product families started product families after the creation of a system. Therefore there are some differences between when product families are started for the different groups.

There were also differences between how the companies distinguished between the different products. The biggest difference was on having a common framework/architecture and adapted modules.

The companies of different sizes used product families for different reasons. The largest difference was with having branding as a reason. Also companies of middle size (20-99) were the only ones that had to follow a line of hardware products as a reason.

There were a lot of differences between how the different sized companies used and saw product families, but as mentioned the size of the sample is however small, especially for the small (0-19) and the large group (≥100). It is therefore hard to make firm conclusions.

6.1.3 Related Research

It is apparent that the conclusions in the related research that is discussed also hold today even though they are old. Much has changed in software engineering in the 15 years since the oldest of the articles was written, but the success factors that then were seen as important are also those that the subjects in the thesis saw as important.

However the claim that having a repository doesn't create good reuse isn't of interest any more because today everyone who applies reuse has some sort of distributed library for sharing. This is because today, with the need for global communication and internet, all computers are connected together in networks in the companies. It is therefore standard to have a repository where data can be saved.

Based on the data from the thesis survey the claim that roles dedicated to reuse are necessary doesn't hold. The subjects in the thesis felt that this wasn't important for having good reuse. This is probably because repositories have become very advanced, and they are therefore easier to keep updated. Another factor that gives good repositories is that the personnel who use it are trained in using it and applying reuse. Therefore it is very important that everyone is trained in reuse, and that they use it.

6.2 Further Work

There are a lot of interesting aspects that can be studied further. These are parts that the surveys brought to mind, but can't conclude on.

What could be interesting to explore further about reuse is:

- Applying of reuse with own methods. A lot of the companies had their own methods for developing and it would be interesting to investigate if any of these are especially adapted for reuse or product families, and if they have improved reuse and product families in the companies.
- The subjects were asked how well they thought reuse worked in their companies and in their departments. This is however not a good measurement, as what the different subjects feel is good reuse can be very different. It would therefore be interesting to find some better way of measuring this, maybe through a more detailed study of the companies, or through experiments. This would make for data that was better to compare between the companies, and also between internal departments and the whole company. There are a lot of factors that can make reuse different between different types of companies.
- All of the companies in the survey were Norwegian. It would be interesting to see if the success factors and failure factors are the same in other countries.

The samples for the small (1-19 employees) and big (≥100 employees) groups were very small when asking about product families. The conclusions that have been made here are therefore somewhat vague. There were some interesting trends that were however discovered, and could be further investigated:

- From the survey it can be seen that the bigger the companies are the longer they have used product families. Is this because companies grow in size by time, or is there another explanation?
- When asked when their companies started product families none of the small companies answered after a system had been created. Is this really the case for all of the small companies who uses product families in Norway, and is the reason that they don't have the resources to do it another way?
- Do the medium sized companies really share so few artifacts while the small and large companies share more? And if they do, why?

Bibliography

[BL01] Magnus Bergquist & Jan Ljungberg, *The power of gifts: organizing social relationships in open source communities*. Information Systems Journal 11: 305-320, 2001.

[BR03] Andrea Bonaccorsi, Cristina Rossi, *Why Open Source Software can succeed.* Research policy, 32(7): 1243-1258, 2003.

[CHW97] Yufeng F. Chen, W. Gerry How and Nazir A.Warsi, *A Multimodeling Framework for Complex Software Reuse*. Proceedings on International Conference and Workshop on Engineering of Computer-Based Systems, 24-28 March 1997, pp. 81 – 87.

[DL00] David J. Lilja, *Measuring computer performance – A practitioner's guide*, Cambridge university press 2000.

[FF95] William B. Frakes and Christopher J. Fox, *Sixteen Questions About Software Reuse*, Communications of the ACM, 38(6): 75-87, June 1995.

[HK03] Eric Von Hippel, Georg von Krogh, *Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science*. Organization Science, 14(2):209-223, March–April 2003.

[JF98] R.E. Johnson, B. Foote, *Designing Reusable Classes*. Journal Object-Oriented Programming, 1(3): 26-49, 1998.

[JV98] Jeffrey Voas, *COTS: The Economical Choice?* IEEE Software, 15(2): 16-19, March 1998.

[LCK03] Len Bass, Paul Clements, Rick Kazman, *Software Architecture in Practice, Second edition*. Addison Wessley 2003.

[LCS05a] Jingyue Li, Reidar Conradi, Odd Petter N. Slyngstad, Christian Bunse, Umair Khan, Marco Torchiano, and Maurizio Morisio, *An Empirical Study on Off-the-Shelf Component Usage in Industrial Projects*. Proceedings of the 6th International Conference on Product Focused Software Process Improvement, Oulu, Finland, June 13 - 15, 2005. Springer Verlag LNCS Volume 3547, June 2005, pp. 54 - 68

[LCS05b] Jingyue Li, Reidar Conradi, Odd Petter N. Slyngstad, Christian Bunse, Umair Khan, Marco Torchiano, Maurizio Morisio, *Validation of New Theses on Off-the-Shelf Component Based Development*, Accepted at the 11th IEEE International Metrics Symposium (Metrics'05), 19-22, Sep. 2005, Como, Italy.

[MC03] Parastoo Mohagheghi and Reidar Conradi: *Different Aspects of Product Family Adoption*, In Frank van der Linden: "Proc. 5th International Workshop on Product Family Evolution (PFE-5)", Siena, Italy, 4-6 Nov., 2003, Springer LNCS 3014 (2004), pp. 459-464.

[MCK04] P. Mohagheghi, R. Conradi, O.M. Killi, H. Schwarz: *An Empirical Study of Software Reuse vs. Defect-Density and Stability*. Proc. of the International Conference on Software Engineering (ICSE'04), Edinburgh, Scotland, UK, 23-28 May 2004, ISSN 0270-5257, ISBN 0-7695-2163-0, IEEE Computer Society Order Number P2163, pp. 282-292.

[MET02] Mauruzio Morisio, Michel Ezran, and Colin Tully, *Success and Failure Factors in Software Reuse*, IEEE Transactions on software engineering, 28(4):340-357, April 1992.

[MS05] Marius Sommerseth, *Reuse through product-families and Frameworks*, Depth study for course System engineering at NTNU, pp. 55 Dec. 2005.

[NATO68] P. Naur and B. Randell (Eds.), *Software Engineering: Report on a Conference Sponsored by the NATO Science Committee*. Garmisch, Germany, 7-11 Oct. 1968. Scientific Affairs Division, NATO, Brussels (Jan. 1969). pp. 79 – 85.

[PDG05] Piergiorgio Di Giacomo, *COTS and Open Source Software Components: Are They Really Different on the Battlefield*, Proceedings of ICCBSS 2005, LNCS 3412, Springer-Verlag, pp. 301-310, 2005.

[WRH00] Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Björn Regnell, Anders Wesslén, *Experimentation In Software Engineering – An Introduction*, Kluwer Academic Publishers 2000

Glossary

Artifact	A component that is reused in a product family
Brønnøysundregistrene	The Norwegian Register of Business Enterprises. More info on http://www.brreg.no
CBSE	Component based software engineering, creating software from components.
Component	Part that can compose a bigger whole.
COTS	Commercial-off-the-shelf, a component created by a commercial company and sold to another company.
CVS	Concurrent Versions Systems, a system that keeps track of all of the changes done on files in a software project. More info on http://www.nongnu.org/cvs/
Database	A collection of records stored electronically in a systematic way on a computer.
Framework	A defined support structure for how a project can be organized and developed.
GOTS	Government-off-the-shelf, a component created by a government and distributed to companies for creation of its software.
IKT-Norge	A Norwegian research institution. More info on http://www.ikt-norge.no
Intranet	A private computer network that uses internet protocols for transferring data. Offers the users a web page for communicating among each others.
MDA	A software design methodology. More info on http://www.omg.org/mda/
NOTS	NASA-off-the-shelf, a component created by NASA and distributed to companies for creation of government software.
OSS	Open Source Software, software that is free for use and further development. More info on http://www.opensource.org
Pattern	A set of rules that can be used to generate something, or a part of something.

Product family	A line of many products that share a common core of equal components, but also have their own specialized functions.
Product line	Same as product family.
RUP	Rational Unified Process, a process for development. More info on http://www.rational.com
Server	A computer that provides services to other computers in a network. Either for storing data or running software.
Software Component	Software functionality that is encapsulated and can be put together with other software components to form a whole.
SPSS	Statistical Package for Social Science, a program for statistical analysis in social science. More info on http://www.spss.com
TTM	Time to market, the time from development starts to the product is out on the market.
UML	An object modeling and specification language used in software development. More info on http://www.uml.org
Wikipedia	A free internet encyclopedia where articles are written by volunteers, but approved by staff. More info on http://en.wikipedia.org/wiki/Wikipedia/
XP	A method for software programming. More info on http://www.extremeprogramming.org

Appendix A: The thesis questionnaire

Invitation letter

Denne spørreundersøkelsen er en del av en diplomoppgave i Datateknikk ved NTNU av Marius Sommerseth med professor Reidar Conradi som veileder. Diplomoppgaven skal identifisere likheter og forskjeller ved komponentbasert utvikling og gjenbruk i norske programvarebedrifter, herunder bedrifter som holder på med programvareutvikling og salg, bedrifter som driver med konsulenttjenester, og bedrifter som driver med intern utvikling av programvare til bruk i egen bedrift. Data som blir innsamlet skal tolkes og analyseres for å se om det er mulig å finne likheter mellom hvordan bedriftene utfører gjenbruk. Data vil også bli sammenlignet opp mot tidligere undersøkelser utført om gjenbruk i programvarebedrifter.

Diplomoppgaven bygger på en forstudie kalt ”Reuse through product-families and frameworks”[22] av Marius Sommerseth ved NTNU høsten 2005. Spørreskjemaet er definert ut fra informasjon innhentet med intervjuer i forstudiet, samt en spørreundersøkelse utført som en del av prosjektet ”Programvarebransjen i Norge 2005” utført av IKT-Norge[23].

Informasjonen innhentet vil bli anonymisert og behandlet konfidensielt. Det vil ikke bli gjengitt noen som helst knytning mellom firmaer og svar i rapporten.

Diplomoppgaven vil bli gjort tilgjengelig på nett i midten av juni 2006, og link til denne vil bli sendt til alle som har deltatt i spørreundersøkelsen.

Spørreundersøkelsen er delt i tre deler; en del med generelle spørsmål om bedriften, en med spørsmål om gjenbruk, og en med spørsmål om produktfamilier. Den er ikke lang og vil ta ca. 10-12 minutter å svare på.

Eventuelle spørsmål til skjemaet kan rettes til:

Marius Sommerseth
5-års siv. ing student datateknikk
E-post: mariusom@stud.ntnu.no
Telefon: xx xx xx xx
NTNU

[22] http://www.idi.ntnu.no/grupper/su/fordypningsprosjekt-2005/sommerseth-fordyp05.pdf
[23] http://www.ikt-norge.no/templates/Page.aspx?id=478

Spørsmål til undersøkelse om produktfamilier og gjenbruk hos bedrifter.

Generelt om bedriften:

1. Navn på bedriften:

2. Navn og stilling til intervjuobjekt:

3. E-post til intervjuobjektet:

4. Hvor mange ansatte er det i bedriften i Norge (IT/systemutviklere/totalt)?
____/____/____

5. Bedriftens hovedmålgruppe/forretningsdomene?
(ett kryss)
□ Programvareutvikling og salg av programvare
□ Programvareutvikling til intern bruk i bedriften
□ Hardwareutvikling
□ Konsulentvirksomhet
□ Annet (spesifiser ____________________)

6. Hvor mange større programutviklingsprosjekter utføres i året (større enn 2 årsverk)?
____

7. Antall programvare produkter/systemer som blir vedlikeholdt per dags dato?
____

Teknisk om bedriften:

8. Hvilket programmeringsspråk er det mest hyppige brukte?
(maks tre kryss)
□ C++
□ Java
□ C#
□ Cobol
□ C
□ Visual Basic
□ Fortran
□ PL/SQL
□ Annet (spesifiser ____________________)

9. Hvilke andre programmeringsspråk brukes?
(flere kryss)
□ C++
□ Java
□ C#
□ Cobol
□ C
□ Visual Basic
□ Fortran
□ PL/SQL
□ Annet (spesifiser ____________________)

10. Hvilke verktøy brukes i utviklingsprosessen?
(flere kryss)
□ .net
□ Powerbuilder
□ Oracle
□ Delphi
□ UML-verktøy (for eksempel Rational Rose)
□ J2EE
□ Annet (spesifiser ____________________)

11. Hvilke prosesser/metoder bruker bedriften ellers i sin utviklingsprosess?
(flere kryss)
□ Rational Unified Process (RUP)
□ Vannfall
□ Extreme Programming (XP)
□ Inkrementell
□ Prototyping
□ Annet (spesifiser ____________________)

Om gjenbruk:

12. Bruker din/deres bedrift gjenbruk basert på komponentbasert utvikling?
□ Ja
□ Nei

13. Hvis ja om bruk av gjenbruk basert på komponentbasert utvikling, hvor mange år har dette pågått?
□ Under 1 år
□ 1-2 år
□ 3-4 år
□ 5-6 år
□ Over 6 år

14. Hvis ja om gjenbruk basert på komponentbasert utvikling, hvordan synes du at dette fungerer i bedriften din?

Veldig dårlig	Dårlig	Ok	Bra	Veldig bra
□	□	□	□	□

15. Hvis ja om gjenbruk basert på komponentbasert utvikling, og bedriften din er delt opp i flere avdelinger, hvordan synes du at dette fungerer internt i avdelingen din?

Veldig dårlig	Dårlig	Ok	Bra	Veldig bra
□	□	□	□	□

16. Hvis ja om gjenbruk basert på komponentbasert utvikling hva slags komponenter bruker din bedrift?
(flere kryss)
□ Open-source (OSS)
□ Commercial-of-the-shelves (COTS)
□ Internutviklede
□ Outsourcede komponenter (komponenter bestilt spesifiserte fra andre firma)

17. Hvis ja om gjenbruk basert på komponentbasert utvikling, hva slags komponenter gjenbrukes?
(flere kryss)
□ Kravspesifikasjoner
□ Arkitektur/design
□ Kodemoduler
□ Testplaner
□ Dokumentasjon
□ Prosessmodeller, prosjektplanlegging
□ Annet (spesifiser ____________________)

18. Hvis ja om gjenbruk basert på komponentbasert utvikling, hvordan lagres komponentene på tvers av produkter - henholdsvis avdelingsbasert eller bedriftsbasert? Hvis bedriften ikke er delt opp i avdelinger kryss kun av for bedriftsbasert.
(flere kryss, maks ett kryss hver linje)

	Avdelingsbasert	Bedriftsbasert
Konfigurasjonssystem (versjonskontrollsystem)	□	□
Egen database bygget spesifikt for deling av komponenter	□	□
Intranett (lokalt intranett ikke bygget spesifikt for deling av komponenter)	□	□
Delte mapper/filkataloger på lokal server	□	□
Annet (spesifiser ____________________)	□	□

19. Hvis ja om gjenbruk basert på komponentbasert utvikling, hvor viktige er følgende begrunnelser for bedriften?
(maks ett kryss per linje)

	Uviktig	Litt viktig	Passe viktig	Meget viktig	Svært viktig	Vet ikke
Lavere kostnader	□	□	□	□	□	□
Kortere utviklingstid	□	□	□	□	□	□
Bedre programvarekvalitet	□	□	□	□	□	□
Mer standardiserte produkter	□	□	□	□	□	□
Mindre vedlikehold	□	□	□	□	□	□
Annet (spesifiser ____________________)	□	□	□	□	□	□
Annet (spesifiser ____________________)	□	□	□	□	□	□
Annet (spesifiser ____________________)	□	□	□	□	□	□

20. Hvis ja om gjenbruk basert på komponentbasert utvikling, hvor viktige er følgende mulige suksessfaktorer for bedriften?
(maks ett kryss per linje)

	Uviktig	Litt viktig	Passe viktig	Meget viktig	Svært viktig	Vet ikke
Bedriften har intern utviklingsprosess der gjenbruk er poengtert	□	□	□	□	□	□
Utviklere må se nytten av gjenbruk	□	□	□	□	□	□
De ansatte blir opplært i gjenbruk	□	□	□	□	□	□
Egne personer dedikert til gjenbruk	□	□	□	□	□	□
God gjenbruk blir belønnet	□	□	□	□	□	□

Ledere må se nytten av gjenbruk	□	□	□	□	□	□
Arkitekturen må være laget for gjenbruk	□	□	□	□	□	□
Komponentene er generalisert for gjenbruk	□	□	□	□	□	□
God dokumentasjon og tilgjengelighet av komponenter i eget gjenbruksbibliotek	□	□	□	□	□	□
Versjonskontroll av komponenter	□	□	□	□	□	□
Annet (spesifiser ____________________)	□	□	□	□	□	□
Annet (spesifiser ____________________)	□	□	□	□	□	□
Annet (spesifiser ____________________)	□	□	□	□	□	□

Om produktfamilier:

21. Bruker din bedrift produktlinjer/familier i utviklingen?
□ Ja
□ Nei

22. Hvis ja om produktlinjer/familier, hvor lenge har dette blitt brukt?
□ Under 1 år
□ 1-2 år
□ 3-4 år
□ 5-6 år
□ Over 6 år

24. Hvis ja om produktlinjer/familier, hva er felles i din bedrifts produktlinjer/familier? (flere kryss)
□ Branding (å bruke et felles navn på en linje av produkter for markedsgjenkjenning)
□ Felles krav
□ Felles arkitektur
□ Felles kode
□ Felles infrastruktur (rammeverket rundt og underliggende plattform)
□ Felles testsystem
□ Annet (spesifiser ____________________)

25. Hvis ja om produktlinjer/familier, hvordan startes produktfamilier? (flere kryss)
□ I forkant (reaktiv produktfamilie/linje, linjen blir opprettet først, og så blir systemene til linjen laget)

□ I etterkant (proaktiv produktfamilie/linje, gamle systemer gjenbrukes for å lage linjen)
□ Underveis/Inkrementelt (under utvikling av ett produkt, startes linjen)
□ Vet ikke

26. Hvis ja om produktlinjer/familier, hvordan skilles de forskjellige produktene i linjen fra hverandre?
(maks tre kryss)
□ Linjen/familien tilpasses vha. parametertilpasning (produktene er de samme men hvordan de er konfigurert er forskjellig)
□ Linjen/familien tilpasses vha. addware (det er en kjerne som er lik for alle produktene, men ekstra programvare er lagt til for å få forskjellig funksjonalitet)
□ Linjen/familien har et fast rammeverk/arkitektur, men modulene den settes sammen av må endres/konfigureres
□ Linjen/familien har faste moduler den settes sammen av, men rammeverket/arkitekturen og hvordan det settes sammen er forskjellige
□ Vet ikke

27. Hvis ja om produktlinjer/familier, hvor mange forskjellige produktlinjer av produkter har din bedrift?

____

28. Hvis ja om produktlinjer/familier, hvor mange forskjellige produkter har typisk hver linje?

____

29. Hvis ja om produktlinjer/familier, hva er hensikten bak å bruke dette?
(flere kryss)
□ Branding (å bruke et felles navn på en linje av produkter for markedsgjenkjenning)
□ Definert ut fra ulike kundegrupper, hver kundegruppe har en linje/familie
□ Definerer linjer/familier ut fra forskjellige standarder (linjen er laget for at produktene i den skal følge en gitt standard)
□ Følger en linje/familie av hardwareprodukter (din bedrift produserer linjer av hardware produkter, og programvaren følger denne familien)
□ Annet (spesifiser ____________________)

30. Har du aspekter som du føler har blitt utelatt om din bedrifts bruk av komponentbasert utvikling, gjenbruk og/eller produktfamilier?

Appendix B: The IKT-Norge questionnaire

Vi takker for at du er villig til å bidra til økt kunnskap om norsk programvaresektor. Undersøkelsen omfatter også tilgrensede områder som elektronisk innhold, e-læring, telekomtjenester over IP, samt driftstjenester.

Undersøkelsen gjennomføres i samarbeid med Norges forskningsråd, Innovasjon Norge, NTNU samt Microsoft, Linpro, MAMUT m.fl..

Undersøkelsen har 3 deler:

A)Hoveddelen: Datagrunnlag for næringsrapporten "Programvaresektoren 2005", herunder næringsvolum, eksport, import m.m.

B)Avklaringer ominteresse for bransjeregister, anbudsvarslinger, varsling om relevante prosjektmuligheter i EU og EUREKA.

C)Spesialundersøkelse fra NTNU, Sintef og Simulasenteret. Formålet er bl.a. forbedring av utviklingsprosesser og undervisningstilbud for programvareindustrien. (frivillig del).

Linken til dette spørreskjemaet er unik for ditt foretak og kan brukes forå komplettere og endre dine svar. Første uttrekk av svardata vil bli gjort i mars.
Alle som fullfører undersøkelsen vil få tilsendt den ferdige næringsrapporten. Individuelle svar vil ikke bli publisert.

q1_1_contact_v - Det foretaksnummeret som du har oppgitt tidligere, er registrert på foretaksnavnet under. Det er viktig at foretaksnr. er korrekt fordi vi vil koble dine svar til regnskapsdata for det oppgitte foretaksnr.

Dersom foretaksnummeret under er feil, må vi be deg finne riktig foretaksnr. i Enhetsregisteret http://www.brreg.no/oppslag/enhet.html. Send riktig foretaksnavn og nr. til support@infosector.net for programvarerapporten og avslutt undersøkelsen, så vil vi kontakte deg på nytt.

Dersom kontaktperson er feil eller mangler, ber vi deg korrigere dette direkte i feltene og gå videre. Dine svar blir lagret når du trykker på pilene nede på skjermen.

(Check all that apply)

□ Offisielt foretaksnavn

□ Foretaksnummer

□ Offisiell næringskode

□ Næringskode tekst

□ Bedriftens webside

□ Kontaktperson - fornavn

□ Kontaktperson - mellomnavn

□ Kontaktperson - etternavn

□ Epost kontaktperson

q1_2_comm_contact - (ikke påkrevd)

(Type text)

q1_3_stockex - 1.3 Børsnotert

(Check one)

□ Ikke børsnotert

□ Børsnotert (angi navn på børsen, eks. Oslo Børs)

q1_4export - 1.4 Aktiviteter i utlandet

(Check one alternative per row)

	Har aktivitet i dag	Planlegger	Planlegger ikke per idag
Markedsføringsaktiviteter i utlandet	□	□	□
Forretningsenheter i utlandet (datterselskap eller avdelinger)	□	□	□
Agenter/forhandlere i utlandet	□	□	□
Teknologisamarbeid med utenlandske aktører	□	□	□
Annet samarbeid i utlandet	□	□	□

q1_5_desc - Legg inn en tekst som beskriver din virksomhet. Bruke gjerne faguttrykk som beskriver foretakets kompetanse, da dette vil øke sjansene for treff ved fritekstsøk. Maks 250 ord.

(Type text)

q1_6_directory -

Vennligst angi hvorvidt du godtar at følgende informasjon publiseres i et bransjeregister:

Firmanavn, adresse, tlf. sentralbord, web-adresse, leverandørkategorier, aktiviteter i utlandet, virkomhetsbeskrivelse.

Bransjeregisteret vil bli publisert på Computerworld Norge sine sider www.computerworld.no fra sommeren. Endringer i bransjeregisteret gjøres ved å endre informasjonen i kap. 1 (spørsmål som begynner på 1.*) i denne undersøkelsen. Dette kan du gjøre ved å gå inn på den linken du bruker nå.

(Check one)

□ Vi ønsker ikke å stå oppført i bransjeregisteret.

□ Vi godkjenner publisering av informasjonen som er angitt over.

q2_1_1_econt - Elektronisk innhold

(Check all that apply)

□ Inntekt (mill. kr)

□ Herav eksport (%)

□ Kostnader (mill. kr)

□ Herav import (%)

□ Årsverk

q2_1_2_sw - Programvare

(Check all that apply)

□ Inntekt (mill. kr)

□ Herav eksport (%)

□ Kostnader (mill. kr)

□ Herav import (%)

□ Årsverk

q2_1_3_elearn - PC-støttetlæring (e-læring)

(Check all that apply)

□ Inntekt (mill. kr)

□ Herav eksport (%)

□ Kostnader (mill. kr)

□ Herav import (%)

□ Årsverk

q2_1_4_iptelecom - Internett-baserte telekom-tjenester

(Check all that apply)

□ Inntekt (mill. kr)
□ Herav eksport (%)
□ Kostnader (mill. kr)
□ Herav import (%)
□ Årsverk

q2_1_5_hosting - Drifts-tjenester for program-varetilgang over IP

(Check all that apply)

□ Inntekt (mill. kr)
□ Herav eksport (%)
□ Kostnader (mill. kr)
□ Herav import (%)
□ Årsverk

q2_1_6_other - Andre næringsområder

(Check all that apply)

□ Inntekt (mill. kr)
□ Herav eksport (%)
□ Kostnader (mill. kr)
□ Herav import (%)
□ Årsverk

q2_1_7_totals - Totalt (total inntekt hentet fra regnskapstall, Brønnøysundregisteret)

(Check all that apply)

□ Inntekt (mill. kr)
□ Herav eksport (%)
□ Kostnader (mill. kr)
□ Herav import (%)
□ Årsverk

q2_2_comm - 2.0.1 Kommentarer

(Type text)

i21 - Følgende spørsmål gjelder fordeling av inntektene fra salg av elektronisk innhold som du rapporterte i spørsmål 2.0.

q21_1 - %-vis andel av totale e-innholdsinntekter

(Check all that apply)

□ 1 Web- og rubrikkannonser - samlet
□ - Web-annonser (% av sum-tallet over)
□ - Rubrikk-annonser
□ - Andre annonsetjenester o.l. på web
□ 2 Medieobjekter, digitalt innhold - samlet
□ - Spill og underholdning
□ - Musikk, bilder, film
□ - Informasjon, nyheter
□ - Artikler, e-bøker
□ - Ringetoner etc.
□ - Andre medieobjekter
□ 3 Digitale sendinger, kringkasting - samlet
□ 4 Annet, spesifiser

q21_2 - Herav eksport (i % av tallet til venstre)

(Check all that apply)

□ 1 Web- og rubrikkannonser - samlet
□ - Web-annonser (% av sum-tallet over)
□ - Rubrikk-annonser
□ - Andre annonsetjenester o.l. på web
□ 2 Medieobjekter, digitalt innhold - samlet
□ - Spill og underholdning
□ - Musikk, bilder, film
□ - Informasjon, nyheter
□ - Artikler, e-bøker
□ - Ringetoner etc.
□ - Andre medieobjekter
□ 3 Digitale sendinger, kringkasting - samlet
□ 4 Annet, spesifiser

q21_2_econt_comm - 2.1.2 Kommentarer til fordelingen

(Type text)

q21_3_econt_markets - Gjør et anslag av fordelingen av de oppgitte e-innholdsinntekter i 2004 på markedssektorer. Omsetningen kan være direkte eller via forhandlere. NB: Samlet fordeling av salg skal være tilnærmet 100%.

(Check one alternative per row)

	0%	0-5%	5-30%	30-50%	50-80%	80-100%	100%
Salg til offentlig virksomheter i Norge	□	□	□	□	□	□	□
Salg til privat næringsliv i Norge	□	□	□	□	□	□	□
Salg til privatpersoner i Norge	□	□	□	□	□	□	□
Eksportandel av e-innholdsinntekter	□	□	□	□	□	□	□

i22 - Følgende spørsmål gjelder fordeling av de programvareinntektene som durapporterte i spørsmål 2.0.

q22_2_sw_prod - %-vis andel av totale programvareinntekter

(Check all that apply)

□ 1 Programvareprodukter og lisenser - salg og utleie ("hyllevare")

□ 2 Salg av passordbeskyttet tilgang til web-løsninger

□ 3 Utviklingsoppdrag, implementering, inkludert tilpasninger av tredjeparts løsninger ("skreddersøm"), samlet

□ a) utviklingsoppdrag på "Open Source Software"

□ 4 Vedlikeholdsinntekter, samlet

□ a) vedlikehold på "Open Source Software"

□ 5 Annet; kurs etc (spesifiser)

q22_3_sw_export - Herav eksport (i % av tallet til venstre)

(Check all that apply)

□ 1 Programvareprodukter og lisenser - salg og utleie ("hyllevare")

□ 2 Salg av passordbeskyttet tilgang til web-løsninger

□ 3 Utviklingsoppdrag, implementering, inkludert tilpasninger av tredjeparts løsninger ("skreddersøm"), samlet

□ a) utviklingsoppdrag på "Open Source Software"

□ 4 Vedlikeholdsinntekter, samlet

□ a) vedlikehold på "Open Source Software"

□ 5 Annet; kurs etc (spesifiser)

q22_2_sw_comm - 2.2.2 Kommentarer til fordelingen

(Type text)

q22_3_sw_markets - Anslå enfordelingen av de oppgitte programvareinntektene i 2004 på markedssektorer. Omsetningen kan være direkte eller via forhandlere. NB: Samlet fordeling av salg skal være tilnærmet 100%.

(Check one alternative per row)

	0%	0-5%	5-30%	30-50%	50-80%	80-100%	100%
Salg til offentlig virksomheter i Norge	□	□	□	□	□	□	□
Salg til privat næringsliv i Norge	□	□	□	□	□	□	□
Salg til privatpersoner i Norge	□	□	□	□	□	□	□
Eksportandel programvare	□	□	□	□	□	□	□

i23 - Fordeling av inntektene fra e-læring som ble rapportert i spørsmål 2.0

q23_2_elearn_prod - %-vis andel av totale e-læringsinntekter

(Check all that apply)

□ Lisenser, salg og utleie av læringsprogram (hyllevare)

□ Lisenser, salg og utleie av LMS-systemer og verktøy (hyllevare)

□ Utviklingsoppdrag, implementering (skreddersøm)

□ Vedlikeholdsinntekter (hyllevare og skreddersøm)

□ Annet; kurs etc (spesifiser)

q23_3_elearn_export - Herav eksport (i % av cellen til venstre)

(Check all that apply)

□ Lisenser, salg og utleie av læringsprogram (hyllevare)

□ Lisenser, salg og utleie av LMS-systemer og verktøy (hyllevare)

□ Utviklingsoppdrag, implementering (skreddersøm)

□ Vedlikeholdsinntekter (hyllevare og skreddersøm)

□ Annet; kurs etc (spesifiser)

q23_4_elearn_comm - 2.3.3 Kommentarer til fordelingen e-læring

(Type text)

q23_5_elearn_markets - Gjør et anslag av fordelingen av de oppgitte e-læringsinntektene i 2004 på markedssektorer. Omsetningen kan være direkte eller via forhandlere. NB: Samlet fordeling av salg skal være tilnærmet 100%.

(Check one alternative per row)

	0%	0-5%	5-30%	30-50%	50-80%	80-100%	100%
Salg til offentlig virksomheter i Norge	□	□	□	□	□	□	□
>> herav salg til offentlig utdanning i Norge	□	□	□	□	□	□	□
Salg til privat næringsliv i Norge	□	□	□	□	□	□	□
Salg til privatpersoner i Norge	□	□	□	□	□	□	□
Eksportomsetning totalt	□	□	□	□	□	□	□

i24 - Kan du fordele inntektene fra internettbasert telekom som du rapporterte i spørsmål 2.0?

q24_1_iptc_prod - %-vis andel av totale internettbasert telekom inntekter

(Check all that apply)

□ Trafikk- og abonnementsinntekter, IP-telefoni

□ Leveranser, tjenester/prosjekter for Web-TV

□ Annet; kurs etc (spesifiser)

q24_1_iptc_export - Herav eksport (i % av cellen til venstre)

(Check all that apply)

□ Trafikk- og abonnementsinntekter, IP-telefoni

□ Leveranser, tjenester/prosjekter for Web-TV

□ Annet; kurs etc (spesifiser)

q24_2_iptc_comm - 2.4.2 Kommentarer til fordelingen internettbasert telekom

(Type text)

q24_3_iptc_markets - Gjør et anslag av fordelingen av de oppgitte telekominntektene i 2004 på markedssektorer. Omsetningen kan være direkte eller via forhandlere. NB: Samlet fordeling av salg skal være tilnærmet 100%.

(Check one alternative per row)

	0%	0-5%	5-30%	30-50%	50-80%	80-100%	100%
Salg til offentlig virksomheter i Norge	□	□	□	□	□	□	□
Salg til privat næringsliv i Norge	□	□	□	□	□	□	□
Salg til privatpersoner i Norge	□	□	□	□	□	□	□
Eksportomsetning totalt	□	□	□	□	□	□	□

i25 - Kan du fordele inntektene fra driftstjenester som du rapporterte i spørsmål 2.0?

q25_1 - %-vis andel av totale inntekter fra driftstjenester

(Check all that apply)

□ Registrering og drift av domener

□ ASP, teknisk serverdrift

□ Videresalg av linjeleie

□ Salg og utleie av programvarelisenser i f.m. ASP-drift

□ Annet, spesifiser

q25_1_exp - Herav eksport (i % av cellen til venstre)

(Check all that apply)

□ Registrering og drift av domener
□ ASP, teknisk serverdrift
□ Videresalg av linjeleie
□ Salg og utleie av programvarelisenser i f.m. ASP-drift
□ Annet, spesifiser

q25_2 - 2.5.2 Kommentarer til fordelingen

(Type text)

q25_3_hosting_markets - Gjør et anslag av fordelingen av de oppgitte driftsinntektene i 2004 på markedssektorer. Omsetningen kan være direkte eller via forhandlere. NB: Samlet fordeling av salg skal være tilnærmet 100%.

(Check one alternative per row)

	0%	0-5%	5-30%	30-50%	50-80%	80-100%	100%
Salg til offentlig virksomheter i Norge	□	□	□	□	□	□	□
Salg til privat næringsliv i Norge	□	□	□	□	□	□	□
Salg til privatpersoner i Norge	□	□	□	□	□	□	□
Eksportomsetning totalt	□	□	□	□	□	□	□

i3_tech -

3.0 Teknologiplattformer

q3_1_tech_platforms - Hvilke plattformer utvikles eller leveres dine løsninger for?

(Check all that apply)

□ Windows Server
□ Unix
□ Linux
□ AS/400
□ IBM 0S390
□ Windows XP
□ Mac OS
□ Symbian (40/60/90)
□ Microsoft Pocket PC
□ Microsoft Windows Mobile Smartphone
□ Annet (spesifisér)
□ Vet ikke

q135 - %-vis andel av samlede inntekter fra e-innhold, programvare, e-læring, IP-telekom og drift.

(Check all that apply)

□ Windows Server
□ Unix
□ Linux
□ AS/400
□ IBM 0S390
□ Windows XP
□ Mac OS
□ Symbian (40/60/90)
□ Microsoft Pocket PC
□ Microsoft Windows Mobile Smartphone
□ Annet (spesifisér)
□ Vet ikke

q136 - Herav eksport (i % av cellen til venstre)

(Check all that apply)

□ Windows Server
□ Unix
□ Linux
□ AS/400
□ IBM 0S390
□ Windows XP
□ Mac OS
□ Symbian (40/60/90)
□ Microsoft Pocket PC
□ Microsoft Windows Mobile Smartphone
□ Annet (spesifisér)
□ Vet ikke

i4_trends -

4 Trender, satsinger m.v.

q4_1_trends_change - 4.1Hvordan vil du karakterisere etterspørselen i dag, sammenlignet med samme tid i fjor?

(Check one)

□ a. Mye bedre
□ b. Bedre
□ c. Uendret
□ d. Dårligere
□ e. Markert dårligere

q4_2_trends_revenue_dev - Hvilke endringer i omsetningen ser dere i de neste 12 mndr.?

(Check one)

□ mer enn 20% vekst
□ 5% til 20% vekst
□ -5% til +5%
□ -5% till -20% (reduksjon)
□ mer enn -20% (reduksjon)

q4_3_trends_growt_obst - 4.3Hvilke er de tre viktigste begrensende faktorene for vekst i de nærmeste 12 månedene?

(Check one alternative per row)

	Viktig	Litt viktig	Ikke viktig
a. Tilgang til kvalifisert personell	□	□	□
b. Tilgang til risikokapital	□	□	□
c. Sterk konkurranse	□	□	□
d. Ufordelaktig arbeidsmarkedsregulering	□	□	□
e. Svak investeringsvilje/etterspørsel	□	□	□
f. Annet (spesifisér)	□	□	□

q4_4_trends_comm - (ikke påkrevd)

(Type text)

q4_5_trends_advantage - 4.5Gradér Norges konkurransekraft internasjonalt innen de markedssektorer bedriften operer i (e-innhold, programvare, e-læring, internettbasert telekom, drift).

(Check one)

□ a. Meget bra

□ b. Bra

□ c. Hverken bra eller dårlig

□ d. Svak

□ e. Meget svak

q4_6_trends_advantage_comm - (ikke påkrevd)

(Type text)

q4_7_trends_potential_comm - Skriv gjerne inn forslag til satsninger der Norge bør ha gode muligheter for å lykkes.

(Type text)

q4_8_trends_int_advantage - Tallene i parantes er produktgruppekoder (CPV) som brukes ianbudsutlysninger.

(Check all that apply)

□ 1 Systemprogramvare generelt

□ System- og serverprogramvare (1500)

□ Operativsystemer (1400)

□ Databasesystemer (1100)

□ Annen systemprogramvare

□ 2 Utviklingsverktøy generelt

□ 3 Applikasjoner generelt

□ Driftstyring (4000)

□ Sikkerhet (2000)

□ Kommunikasjonsprogramvare (6000)

□ Programvare til medisinsk bruk (3000)

□ Undervisning (9300)

□ Finans- og økonomisystemer, (9400)

□ Kontorautomasjon, CRM (9500)

□ Multimediaprogramvare (1700)

□ Andre applikasjoner (spesifisér)

i5_expansion -

5 Innovasjon og internasjonalisering

q5_1_rd_participate - Angi virkemidler eller nettverk som ble benyttet.

(Check one alternative per row)

	Ja	Nei	Vet ikke
EU	□	□	□
EUREKA	□	□	□
Norges forskningsråd	□	□	□
Innovasjon Norge/SND	□	□	□
SkatteFUNN	□	□	□
Nordisk Innovasjon	□	□	□
Andre (spesifisér)	□	□	□

q5_2_dev - Hvordan rolle spiller innovasjonsprosjekter (EU, EUREKA, SkatteFUNN etc.) i utviklingen av bedriften?

(Check one alternative per row)

	Stor rolle	Middles rolle	Liten rolle	Ingen rolle	Vet ikke
Markedsutvikling	□	□	□	□	□
Eksport	□	□	□	□	□
Nettverksbygging	□	□	□	□	□
Finansiering	□	□	□	□	□
Produktutvikling	□	□	□	□	□

q5_3_rd_experience - Skriv inn erfaringer med innovasjonsprosjekter.

(Type text)

q5_4_rd_interest - □□Det forutsettes at prosjektene dekker kostnader til reiser, møter, utvikling av piloter, eksterne fageksperter samt en andel av medgått tid. Prioritér fra listen med prosjektteknologier. Svarene er ikke påkrevd, men vil gi et grunnlag for å diskutere initiativ for felles innovasjonsprosjekter.

(Check one alternative per row)

	Positivt interessert	Kan være interessert	Ikke interessert	Vet ikke (mangler informasjon)
Computer Games	□	□	□	□
Computer Software	□	□	□	□
Databases, Database Management, Data Mining	□	□	□	□
Electronic Commerce, Electronic Payment, Electronic Signature	□	□	□	□
Imaging, Image Processing, Pattern Recognition	□	□	□	□
Knowledge Management, Process Management	□	□	□	□
Simulation	□	□	□	□
Speech Processing/Technology	□	□	□	□
Applications for Health	□	□	□	□
Applications for Tourism	□	□	□	□
Applications for Transport and Logistics	□	□	□	□
e-Government	□	□	□	□
GIS Geographical Information Systems	□	□	□	□
Multimedia	□	□	□	□
Cultural Heritage	□	□	□	□
E-Learning	□	□	□	□
E-Publishing, Digital Content	□	□	□	□
Information Filtering, Semantics, Statistics	□	□	□	□
Visualisation, Virtual Reality	□	□	□	□
Telecommunications	□	□	□	□

q5_5_RD_comm - Her kan du legge inn spørsmål og kommentarer om innovasjonsprosjekter, program og virkemidler. Spørsmål som ikke besvares i rapporten, vil bli besvart i form av epost til avsender.

(Type text)

i6 - 6 Tilleggsavklaringer (del B)

q6_1_listing - Ønsker bedriften oppføring i et planlagt bransjeregister på web (engelsk og norsk tekst)?□Informasjon som registreres er profileringstekst, produkter, markeder m.m..

(Check one)

□ Ja, send meg mer informasjon om bransjeregisteret.

□ Ja, send mer informasjon om bransjeregisteret til (epostadresse):

□ Ønsker ikke å bli kontaktet angående bransjeregisteret.

q6_2_tenders - IKT-Norge vurderer å forhandle frem rabatter på anbudsvarslingfor sine medlemer. Vil dette være av interesse for din virksomhet?

(Check one)

□ Ja, dette er relevant. Vi er medlem og ønsker et tilbud på dette.

□ Ja, dette er relevant, men vi er ikke medlem. Vi ønsker tilbud på medlemskap med anbudsvarsling.

□ Nei, vi har anbudsvarsling og er fornøyd med den.

□ Nei, dette er ikke av interesse for oss.

□ Vil ikke ta stilling til dette nå.

q6_3_project_alert -

IKT-Norge vurderer å tilby varsling av relevante prosjektmuligheter ogpartnersøki EU og EUREKA, samt bistand til søknadsutforming for innovasjonsprosjekter.
Vil dette være av interesse?

(Check one)

□ Ja, send meg informasjon om dette.

□ Ja, send informasjon til følgende person (epostadresse)

□ Nei, dette er ikke av interesse for oss.

□ Vil ikke ta stilling til dette nå.

i7 - 7 Prosess- og kvalitetsforbedringer (del C)

q7_1_opning - NTNU er med i flere forskningsprosjekter, dels sammen med IKT-Norge, SINTEF, Simula-senteret, om forbedring av utviklingsprosesser/ programvareteknologier i norsk IKT-industri. Vi er derfor interessert i å undersøke holdninger til kvalitetssystemer og prosessforbedring rent generelt, samt faktisk bruk av komponentbasert utvikling med fokus på COTS/OSS. NTNU ønsker å bruke resultatene til å forbedre sin undervisning innen programvareutvikling. Vil du svare på ekstraspørsmålene?

(Check one)

□ Ja

□ Nei

q7_2_info - 7.2 Grunnleggende info

(Check one alternative per row)

	Ja	Nei
Har foretaket samarbeidet med en norsk FoU-institusjon om forbedrede utviklingsprosesser/ bedre programvarekvalitet de siste 5 år?	□	□
Har foretaket engasjert et konsulentfirma om forbedrede utviklingsprosesser/ bedre programvarekvalitet de siste 5 år?	□	□
Er foretaket ISO-9000 sertifisert?	□	□
Planlegger foretaket de neste 2 år å blir ISO-9000 sertifisert?	□	□
Har foretaket laget et internt kvalitetssystem?	□	□
Hvis nei, har foretaket planer om et slikt de neste 2 år?	□	□
Er foretaket del av et større firma som har laget et internt kvalitetssystem?	□	□

q7_3_kvalsyst_tilpass - Du svarte ja på at foretaket er en del av et større firma som har laget et internt kvalitetssystem.

(Check one alternative per row)

	ikke endret	litt endret	en god del endret	endret i meget stor grad	vet ikke
I hvor stor grad er dette blitt tilpasset lokalt?	□	□	□	□	□

q7_4_prosess_forbedring - 7.4 Prosessforbedring

(Check one alternative per row)

	Ja	Nei
Har foretaket et pågående initiativ innen prosessforbedring?	□	□

q7_5_prosess_forbedring_motiv - Hva er motivene bak dette? (kryss av alternativene som passer):

(Check all that apply)

□ lavere kostnader

□ kortere utviklingstid

□ bedre programvarekvalitet

□ økt renomme blant kundene

□ annet (angi nærmere) ....

q7_6_agile - 7.6. Agile utviklingsprosesser

(Check one alternative per row)

	Ja	Nei
Benytter foretaket s.k. smidige ("agile") utviklingsprosesser?	□	□

q7_7_agile_motiv - Hva er motivene bak dette? (kryss av alternativene som passer):

(Check all that apply)

□ lavere kostnader

□ kortere utviklingstid

□ bedre programvarekvalitet

□ annet (angi nærmere) ....

q7_8_komponent_b_cots - 7.8 Komponentbasert utvikling, COTS og OSS

(Check one alternative per row)

	Ja	Nei
Benytter foretaket komponentbasert utvikling vhj. COTS (Commercial-Off-The-Shelf)?	□	□
Benytter foretaket komponentbasert utvikling vhj. OSS (Open Source Software)?	□	□

q7_10_komponent_b_motiv - Hva er motivene for bruk av COTS eller OSS ovenfor ? (kryss av alternativene som passer):

(Check all that apply)

□ lavere kostnader

□ kortere utviklingstid

□ bedre programvarekvalitet

□ mer standardiserte produkter

□ annet (angi nærmere) ....

q7_11_gjenbruk - 7.11 Gjenbruk av komponenter

(Check one alternative per row)

	Ja	Nei
Benytter foretaket systematisk gjenbruk av egenutviklede komponenter eller andre arbeidsprodukter?	□	□

q7_12_gjenbruk_hva - Hvilke arbeidsprodukter gjenbrukes systematisk (kryss av alternativene som passer)?

(Check all that apply)

□ kravspesifikasjoner

□ design/arkitekturer

□ kodemoduler/komponenter

□ dokumentasjon

□ testplaner

□ annet (angi nærmere)

q7_13_gjenbruk_motiv - Hva er motivene bak dette? (kryss av alternativene som passer):

(Check all that apply)

□ lavere kostnader
□ kortere utviklingstid
□ bedre programvarekvalitet
□ mer standardiserte produkter
□ annet (angi nærmere) ....

q7_14_outsc - 7.14 Outsourcing

(Check one alternative per row)

	Ja	Nei
Benytter foretaket "outsourcing" til andre firmaer?	□	□

q7_15_outsc_region - Hvilke regioner kommer disse fra?

(Check all that apply)

□ Norge
□ Norden utenom Norge
□ Vest-Europa
□ Øst-Europa
□ India
□ Asia utenom India
□ USA
□ Amerika utenom USA
□ Annet (angi nærmere)

q7_16_outsc_motiv - Hva er motivene bak dette? (kryss av alternativene som passer):

(Check all that apply)

□ lavere kostnader
□ kortere utviklingstid
□ bedre programvarekvalitet
□ mer standardiserte produkter
□ annet (angi nærmere) ....

q99_Spoersmaal - Har du noen generelle spørsmål eller kommentarer til undersøkelsen?

(Type text)

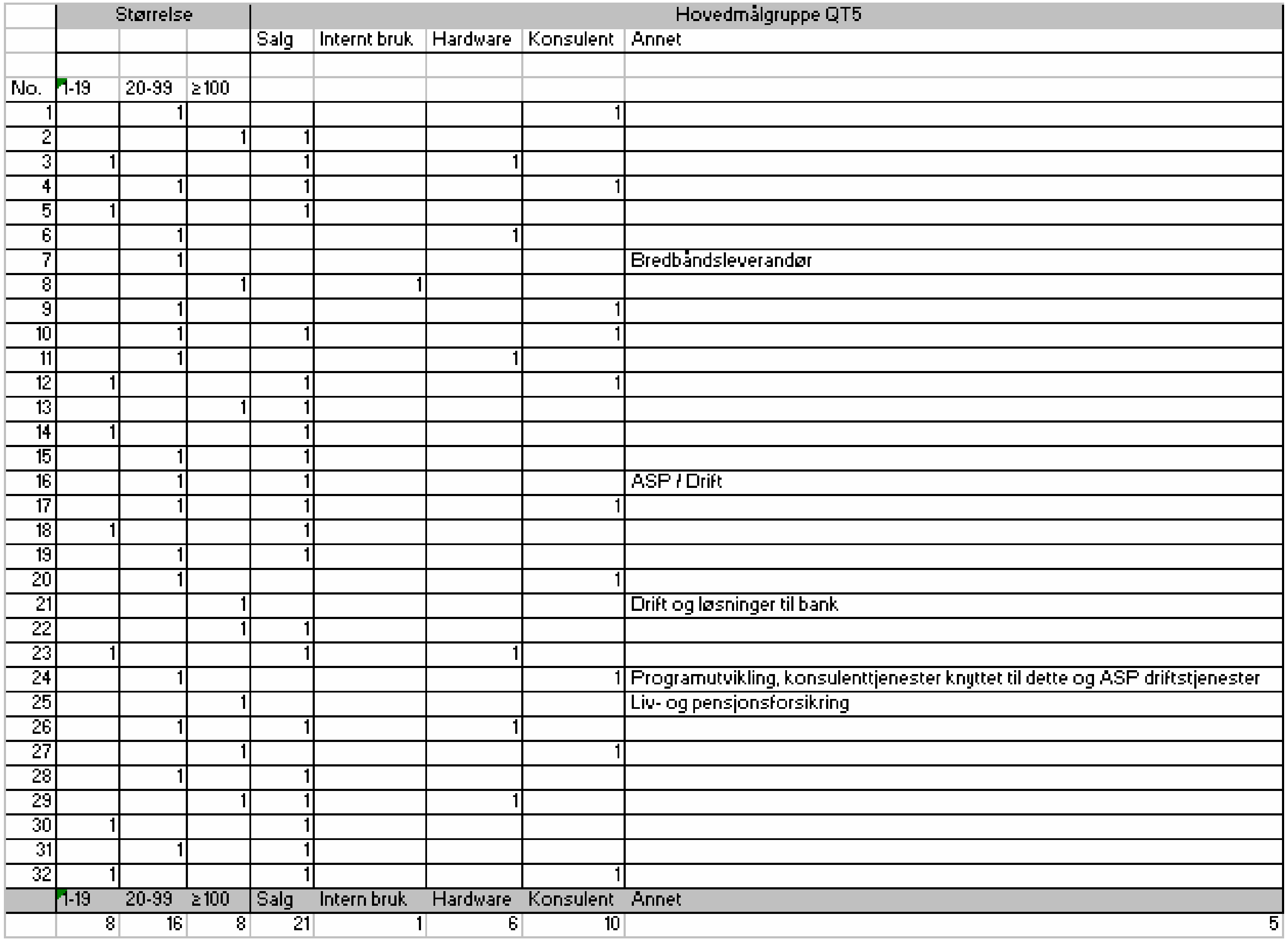
i70 - Du kan komplettere dine svar ved å bruke samme linken. Første uttrekk av svardata vil bli gjort 10. mars.

Første utgave av rapporten ventes ferdig i mars 2006. Rapporten sendes til alle som har levert komplette svar.

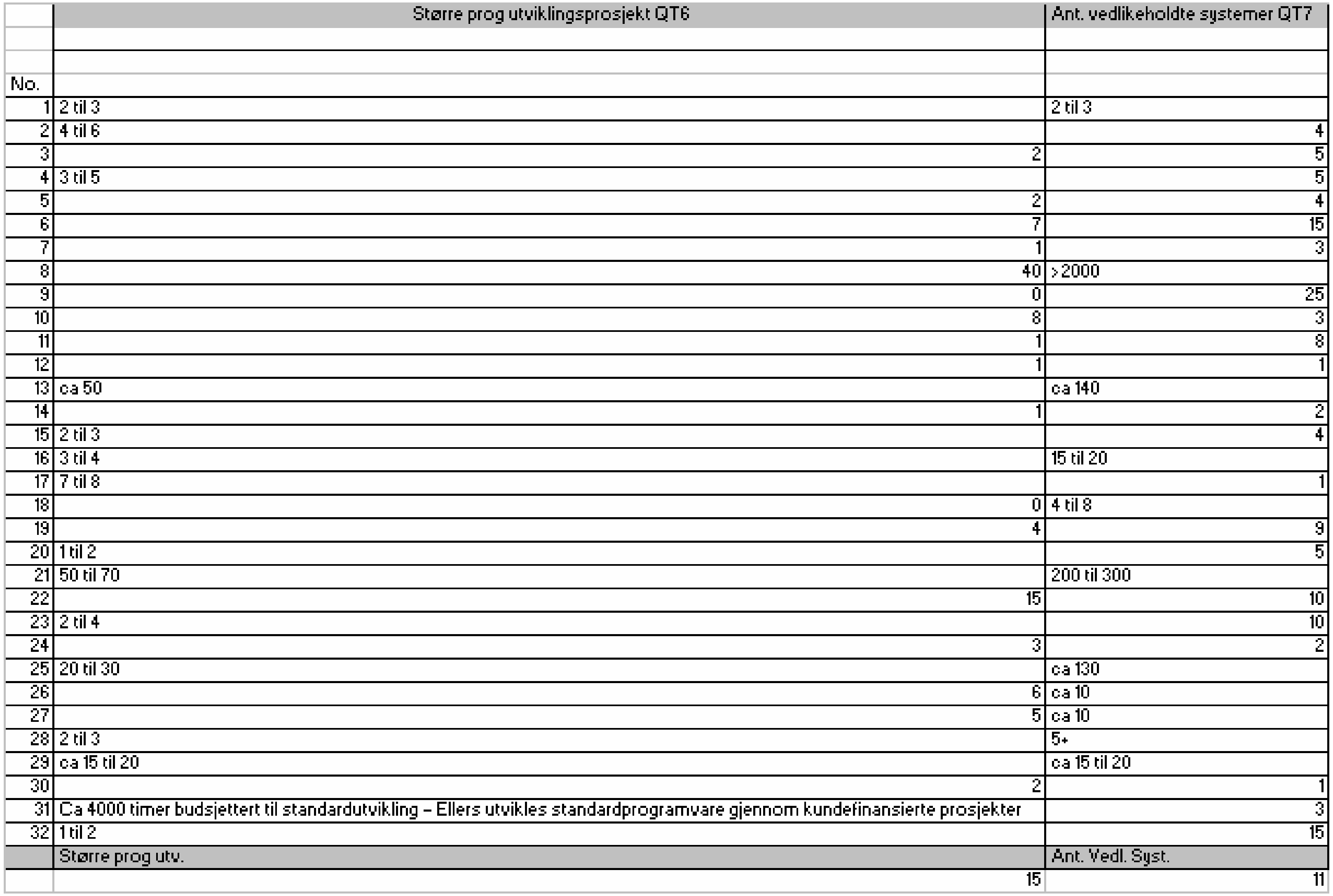
Med vennlig hilsen IKT-Norge

Appendix C: Data from the Thesis Survey

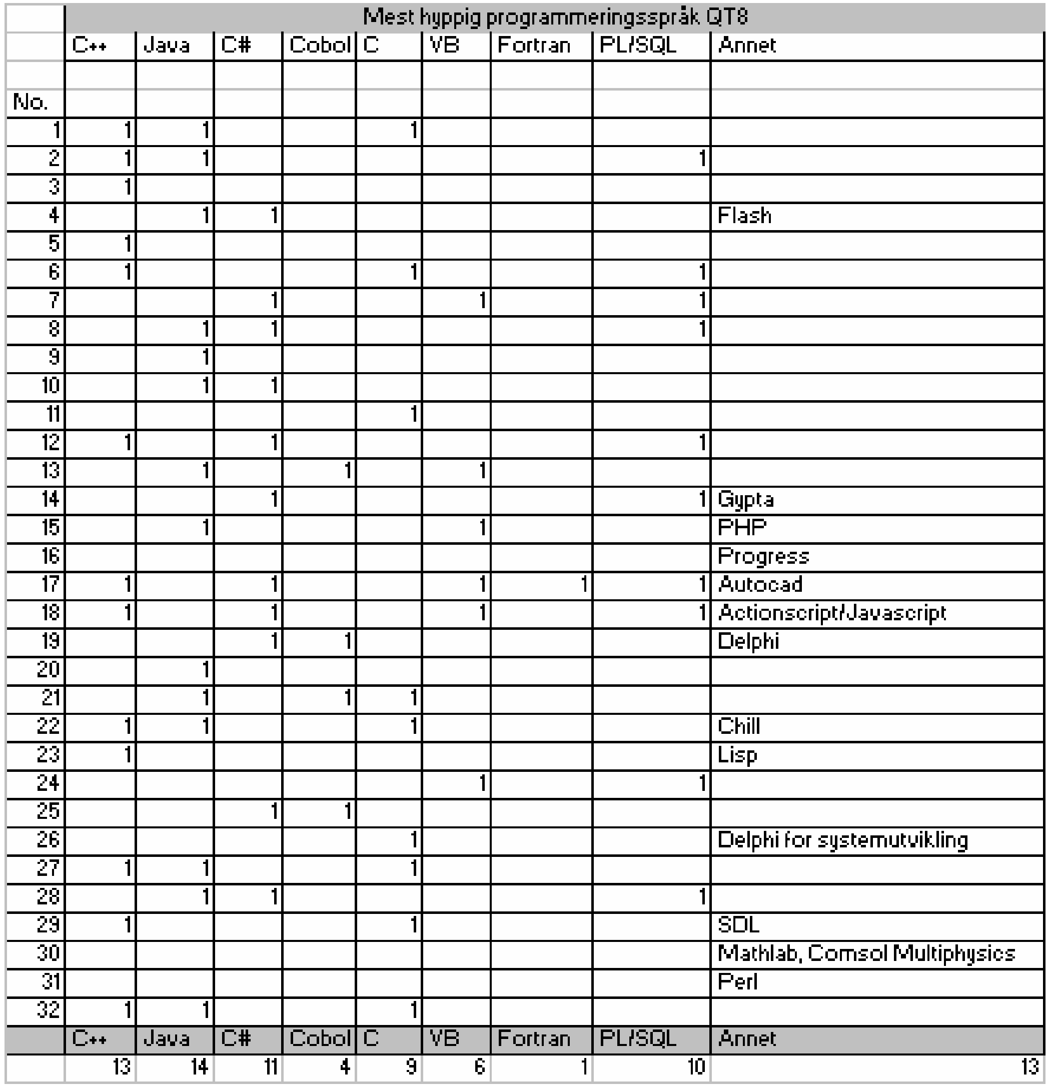
Here the data from the thesis survey is presented. The leftmost of the colons in every table shows which number company answered.



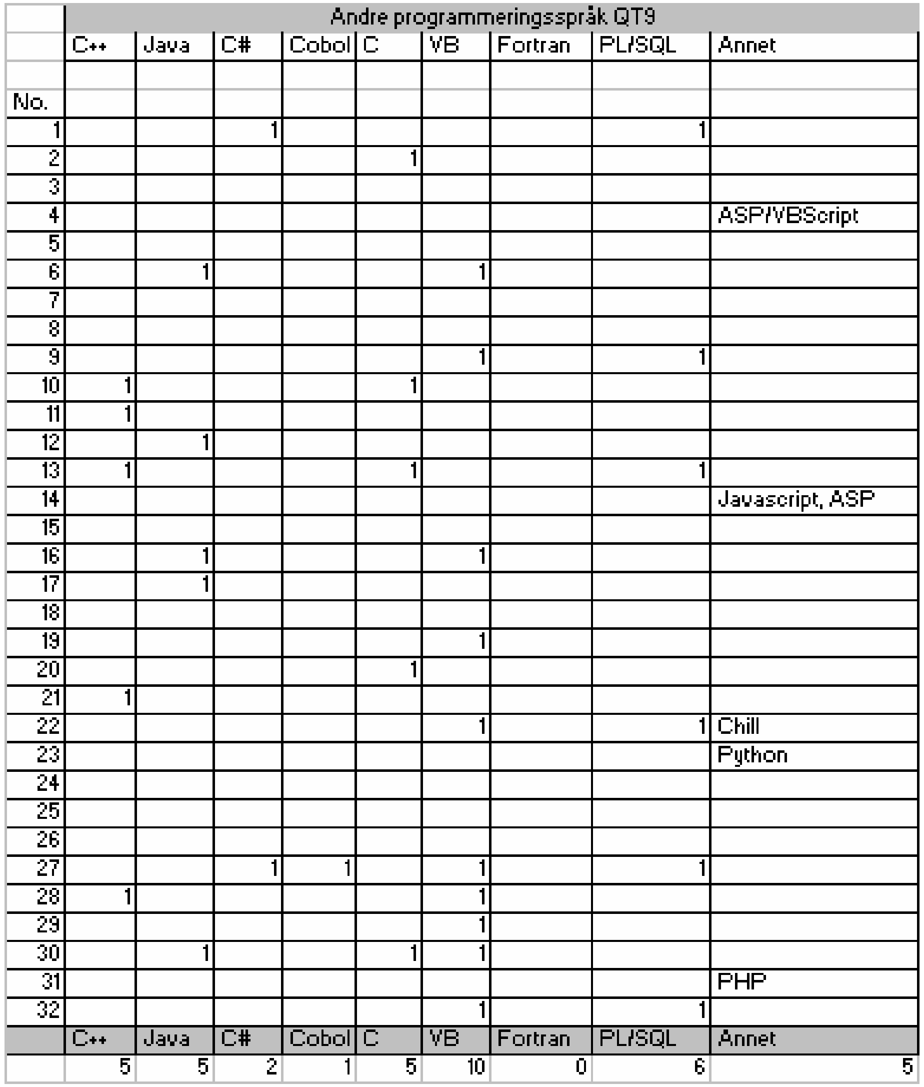
	Størrelse			Hovedmålgruppe QT5				
				Salg	Internt bruk	Hardware	Konsulent	Annet
No.	1-19	20-99	≥100					
1		1					1	
2			1	1				
3	1			1		1		
4		1		1			1	
5	1			1				
6		1				1		
7		1						Bredbåndsleverandør
8			1		1			
9		1					1	
10		1		1			1	
11		1				1		
12	1			1			1	
13			1	1				
14	1			1				
15		1		1				
16		1		1				ASP / Drift
17		1		1			1	
18	1			1				
19		1		1				
20		1					1	
21			1					Drift og løsninger til bank
22			1	1				
23	1			1		1		
24		1					1	Programutvikling, konsulenttjenester knyttet til dette og ASP driftstjenester
25			1					Liv- og pensjonsforsikring
26		1		1		1		
27			1				1	
28		1		1				
29			1	1		1		
30	1			1				
31		1		1				
32	1			1			1	
	1-19	20-99	≥100	Salg	Intern bruk	Hardware	Konsulent	Annet
	8	16	8	21	1	6	10	5



	Større prog utviklingsprosjekt QT6	Ant. vedlikeholdte systemer QT7
No.		
1	2 til 3	2 til 3
2	4 til 6	4
3	2	5
4	3 til 5	5
5	2	4
6	7	15
7	1	3
8	40	> 2000
9	0	25
10	8	3
11	1	8
12	1	1
13	ca 50	ca 140
14	1	2
15	2 til 3	4
16	3 til 4	15 til 20
17	7 til 8	1
18	0	4 til 8
19	4	9
20	1 til 2	5
21	50 til 70	200 til 300
22	15	10
23	2 til 4	10
24	3	2
25	20 til 30	ca 130
26	6	ca 10
27	5	ca 10
28	2 til 3	5+
29	ca 15 til 20	ca 15 til 20
30	2	1
31	Ca 4000 timer budsjettert til standardutvikling – Ellers utvikles standardprogramvare gjennom kundefinansierte prosjekter	3
32	1 til 2	15
	Større prog utv.	Ant. Vedl. Syst.
	15	11

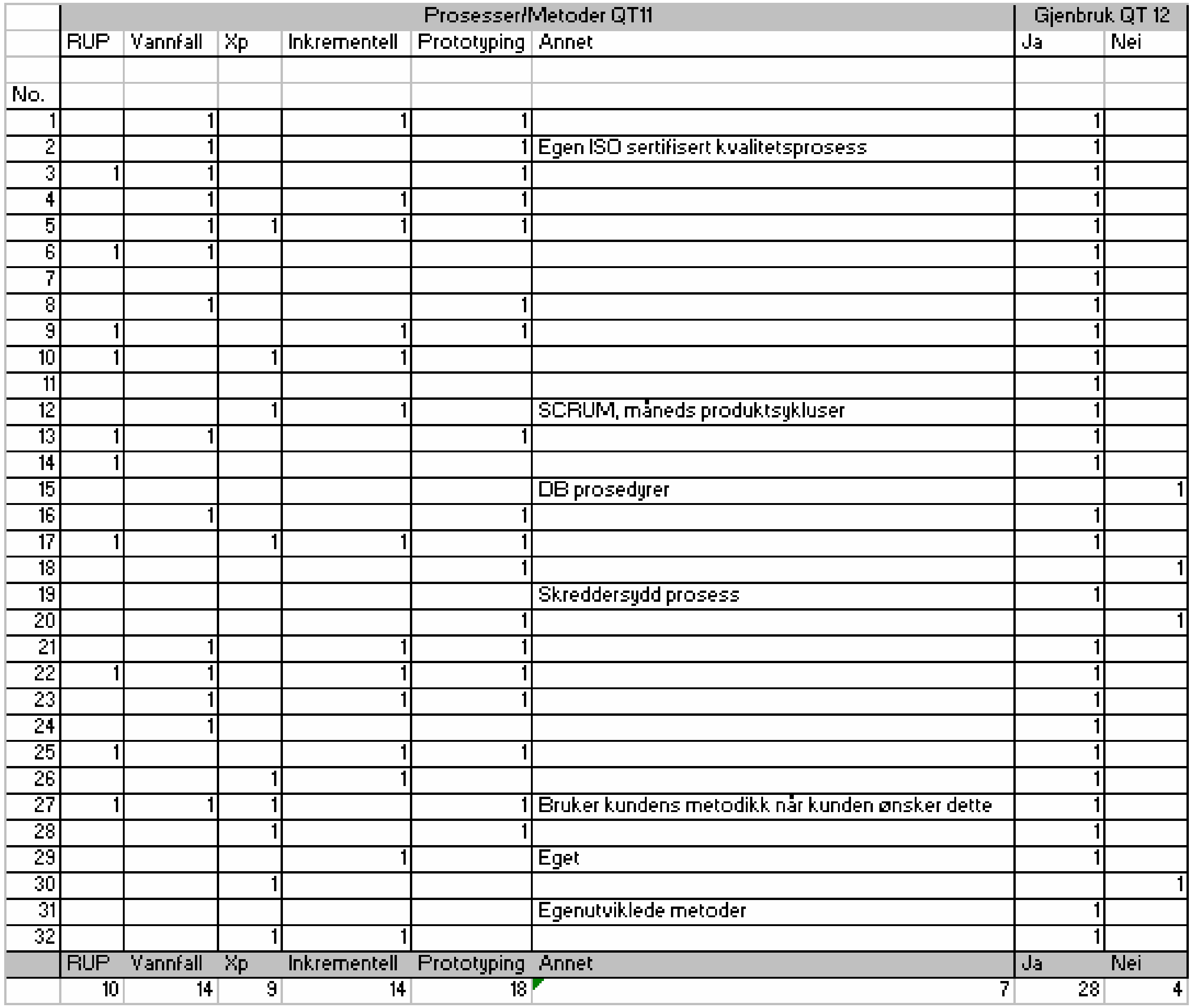


	Mest hyppig programmeringsspråk QT8								
	C++	Java	C#	Cobol	C	VB	Fortran	PL/SQL	Annet
No.									
1	1	1			1				
2	1	1						1	
3	1								
4		1	1						Flash
5	1								
6	1				1			1	
7			1			1		1	
8		1	1					1	
9		1							
10		1	1						
11					1				
12	1		1					1	
13		1		1		1			
14			1					1	Gypta
15		1				1			PHP
16									Progress
17	1		1			1	1	1	Autocad
18	1		1			1		1	Actionscript/Javascript
19			1	1					Delphi
20		1							
21		1		1	1				
22	1	1			1				Chill
23	1								Lisp
24						1		1	
25			1	1					
26					1				Delphi for systemutvikling
27	1	1			1				
28		1	1					1	
29	1				1				SDL
30									Mathlab, Comsol Multiphysics
31									Perl
32	1	1			1				
	C++	Java	C#	Cobol	C	VB	Fortran	PL/SQL	Annet
	13	14	11	4	9	6	1	10	13



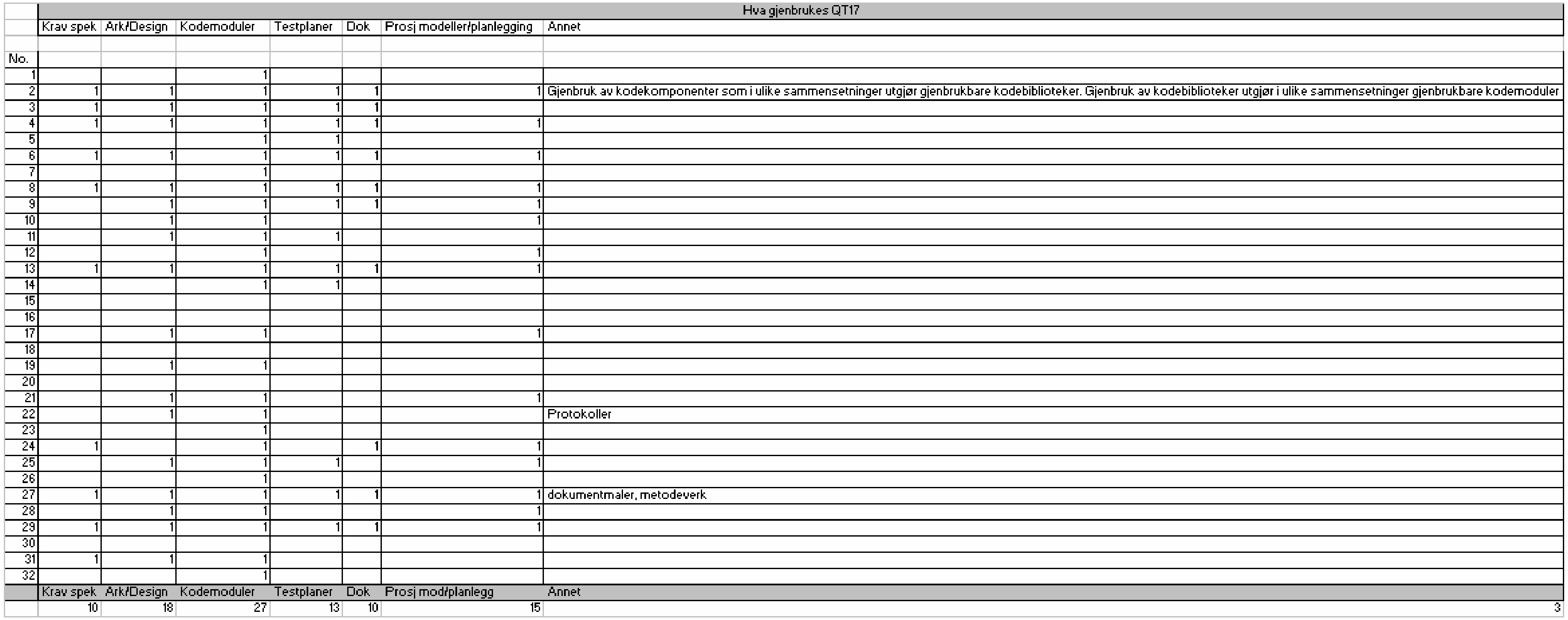
	Andre programmeringsspråk QT9								
	C++	Java	C#	Cobol	C	VB	Fortran	PL/SQL	Annet
No.									
1			1					1	
2					1				
3									
4									ASP/VBScript
5									
6		1				1			
7									
8									
9						1		1	
10	1				1				
11	1								
12		1							
13	1				1			1	
14									Javascript, ASP
15									
16		1				1			
17		1							
18									
19						1			
20					1				
21	1								
22						1		1	Chill
23									Python
24									
25									
26									
27			1	1		1		1	
28	1					1			
29						1			
30		1			1	1			
31									PHP
32						1		1	
	C++	Java	C#	Cobol	C	VB	Fortran	PL/SQL	Annet
	5	5	2	1	5	10	0	6	5

	Verktøy utviklings prosess QT10						
	.net	Powerbuilder	Oracle	Delphi	UML	J2EE	Annet
No.							
1	1					1	
2			1			1	Sun Studio 11 IDE, IntelliJ Java IDE, Diverse optimaliserings, test, support verktøy
3	1						
4	1		1			1	
5	1						QT
6		1			1		
7	1						
8	1		1		1	1	
9					1	1	WSAD/Eclipse
10	1				1	1	
11				1			
12	1						
13	1		1		1	1	
14	1		1		1		
15							
16	1						Progress
17	1		1		1		
18							
19	1			1	1		
20			1				
21		1	1		1		Minst 30 til
22			1		1		jBuilder, Teleuse
23							MS Visual studio, Lisp works
24	1		1				
25	1				1		
26					1		
27	1				1	1	
28	1		1	1		1	
29	1						Telelogic
30							Matlab
31							CVS (Concurrent Versioning System)
32	1				1	1	
	.net	Powerbuilder	Oracle	Delphi	UML	J2EE	Annet
	19	2	11	3	14	10	10

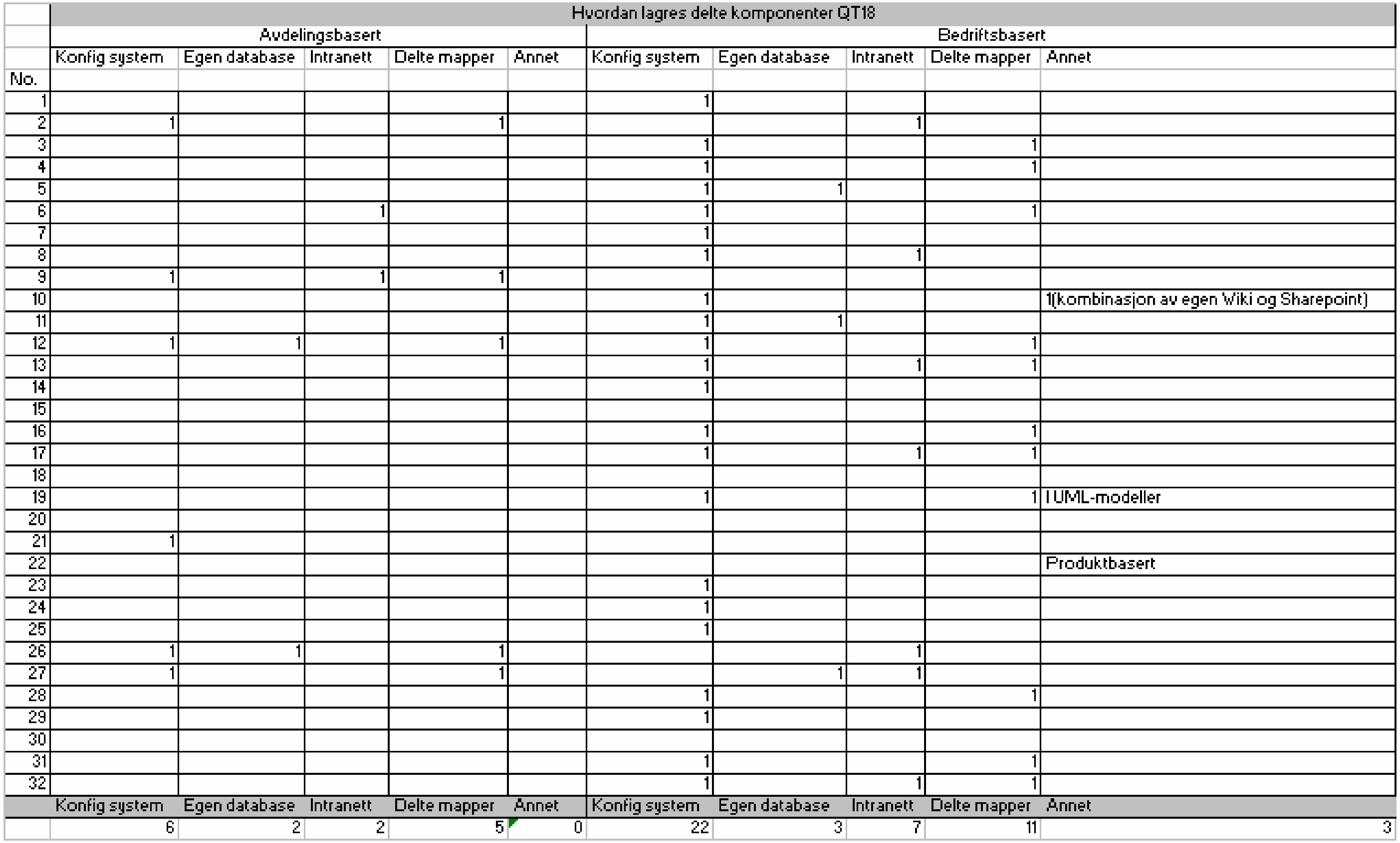


	Prosesser/Metoder QT11						Gjenbruk QT 12	
	RUP	Vannfall	Xp	Inkrementell	Prototyping	Annet	Ja	Nei
No.								
1		1		1	1		1	
2		1			1	Egen ISO sertifisert kvalitetsprosess	1	
3	1	1			1		1	
4		1		1	1		1	
5		1	1	1	1		1	
6	1	1					1	
7							1	
8		1			1		1	
9	1			1	1		1	
10	1		1	1			1	
11							1	
12			1	1		SCRUM, måneds produktsykluser	1	
13	1	1			1		1	
14	1						1	
15						DB prosedyrer		1
16		1			1		1	
17	1		1	1	1		1	
18					1			1
19						Skreddersydd prosess	1	
20					1			1
21		1		1	1		1	
22	1	1		1	1		1	
23		1		1	1		1	
24		1					1	
25	1			1	1		1	
26			1	1			1	
27	1	1	1		1	Bruker kundens metodikk når kunden ønsker dette	1	
28			1		1		1	
29				1		Eget	1	
30			1					1
31						Egenutviklede metoder	1	
32			1	1			1	
	RUP	Vannfall	Xp	Inkrementell	Prototyping	Annet	Ja	Nei
	10	14	9	14	18	7	28	4

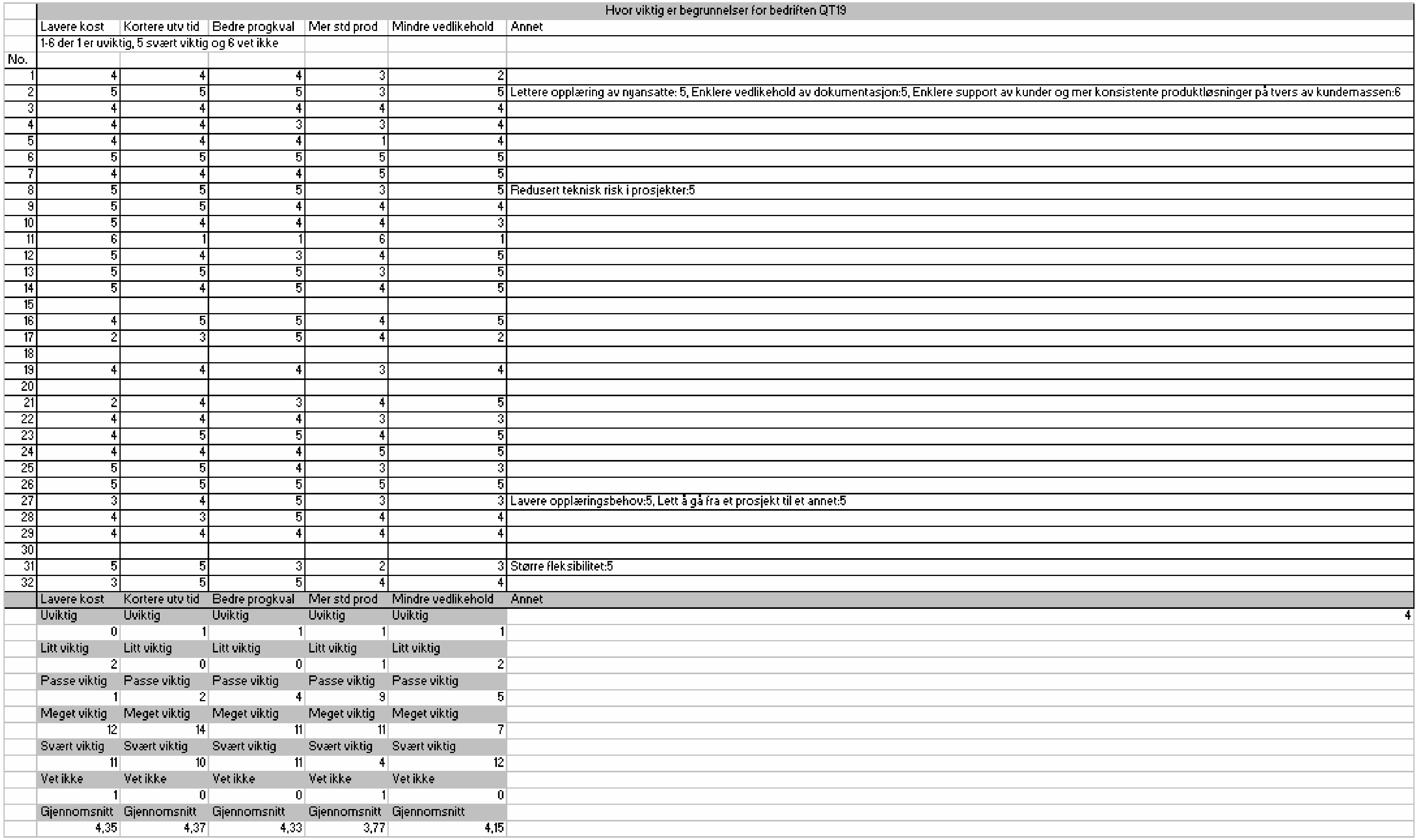
	Antall år QT13	Fungerer i bedriften QT14	Fungerer i avdelingen QT15	Hva slags komponenter QT16			
				OSS	COTS	Intern. utv.	Outsourced
		1 til 5 der 1 er dårligst	1 til 5 der 1 er dårligst				
No.							
1	over 6	3	3	1			
2	over 6	5	2	1		1	
3	over 6	4				1	
4	1 til 2	4	4		1	1	
5	3 til 4	5	5	1		1	
6	over 6	4	4	1	1	1	
7	3 til 4	4				1	1
8	5 til 6	3	4	1	1	1	
9	3 til 4	4	4	1		1	
10	3 til 4	4	4	1	1	1	
11	over 6	3			1		
12	1 til 2	4	4	1	1	1	
13	over 6	4	4		1	1	
14	3 til 4	3				1	
15							
16	over 6	3	3			1	
17	over 6	3		1	1	1	
18							
19	over 6	3	3		1	1	
20							
21	over 6	3	4	1	1	1	
22	over 6	3	3	1	1	1	
23	1 til 2	2	2			1	
24	over 6	3			1	1	
25	3 til 4	3	3			1	1
26	over 6	3	3			1	
27	5 til 6	3	4	1	1	1	
28	1 til 2	3	3	1	1	1	
29	over 6	3	3	1		1	
30							
31	over 6	5	5	1		1	
32	3 til 4	3	3	1	1	1	
	1 til 2	Fungerer i bedriften	Fungerer i avdelingen	OSS	COTS	Intern. utv.	Outsourced
	4	Veldig dårlig	Veldig dårlig	16	15	26	2
	3 til 4	0	0				
	7	Dårlig	Dårlig	Bare COTS			
	5 til 6	1	2	5			
	2	Ok	Ok	Bare OSS			
	over 6	16	9	6			
	15	Bra	Bra	Begge			
		8	9	10			
		Veldig bra	Veldig bra	Ingen			
		3	2	11			
		Gjennomsnitt	Gjennomsnitt				
		3,46	3,5				



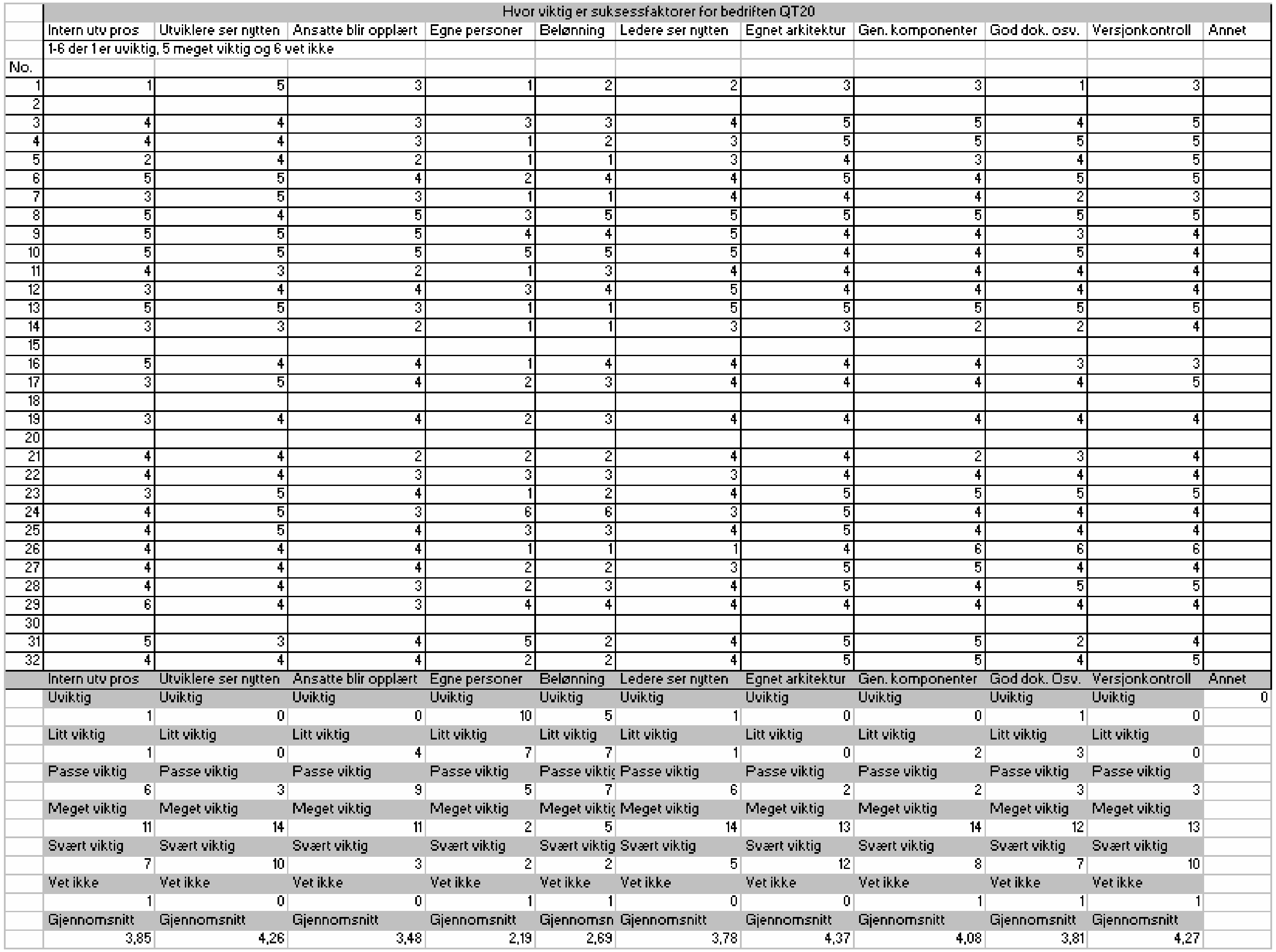
	Hva gjenbrukes QT17						
	Krav spek	Ark/Design	Kodemoduler	Testplaner	Dok	Prosj modeller/planlegging	Annet
No.							
1			1				
2	1	1	1	1	1	1	Gjenbruk av kodekomponenter som i ulike sammensetninger utgjør gjenbrukbare kodebiblioteker. Gjenbruk av kodebiblioteker utgjør i ulike sammensetninger gjenbrukbare kodemoduler
3	1	1	1	1	1		
4	1	1	1	1	1	1	
5			1	1			
6	1	1	1	1	1	1	
7			1				
8	1	1	1	1	1	1	
9		1	1	1	1	1	
10		1	1			1	
11		1	1	1			
12			1			1	
13	1	1	1	1	1	1	
14			1	1			
15							
16							
17		1	1			1	
18							
19		1	1				
20							
21		1	1			1	
22		1	1				Protokoller
23			1				
24	1		1		1	1	
25		1	1	1		1	
26			1				
27	1	1	1	1	1	1	dokumentmaler, metodeverk
28		1	1			1	
29	1	1	1	1	1	1	
30							
31	1	1	1				
32			1				
	Krav spek	Ark/Design	Kodemoduler	Testplaner	Dok	Prosj mod/planlegg	Annet
	10	18	27	13	10	15	3



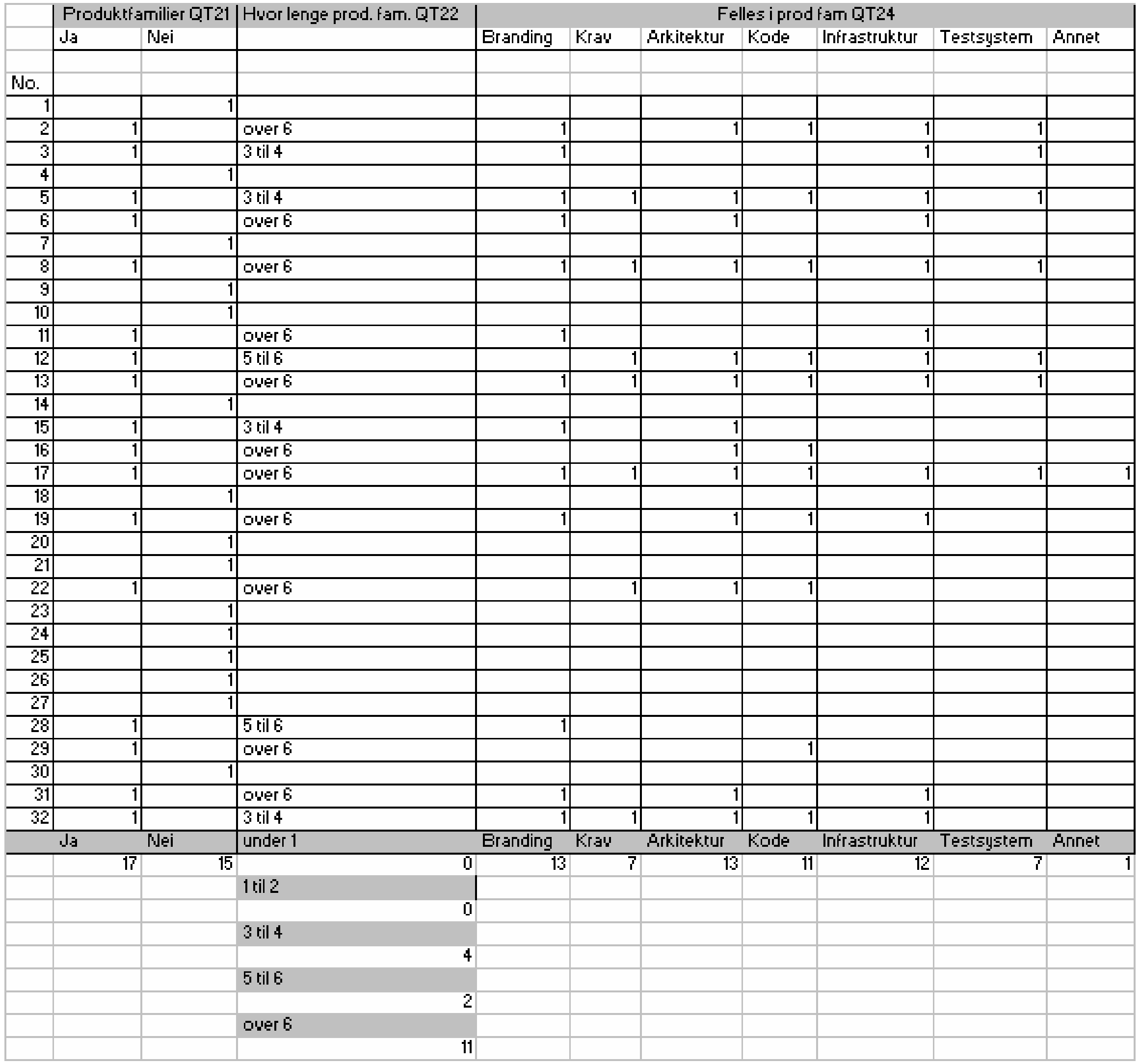
	Hvordan lagres delte komponenter QT18									
	Avdelingsbasert					Bedriftsbasert				
	Konfig system	Egen database	Intranett	Delte mapper	Annet	Konfig system	Egen database	Intranett	Delte mapper	Annet
No.										
1						1				
2	1			1				1		
3						1			1	
4						1			1	
5						1	1			
6			1			1			1	
7						1				
8						1		1		
9	1		1	1						
10						1				1(kombinasjon av egen Wiki og Sharepoint)
11						1	1			
12	1	1		1		1			1	
13						1		1	1	
14						1				
15										
16						1			1	
17						1		1	1	
18										
19						1			1	I UML-modeller
20										
21	1									
22										Produktbasert
23						1				
24						1				
25						1				
26	1	1		1				1		
27	1			1			1	1		
28						1			1	
29						1				
30										
31						1			1	
32						1		1	1	
	Konfig system	Egen database	Intranett	Delte mapper	Annet	Konfig system	Egen database	Intranett	Delte mapper	Annet
	6	2	2	5	0	22	3	7	11	3



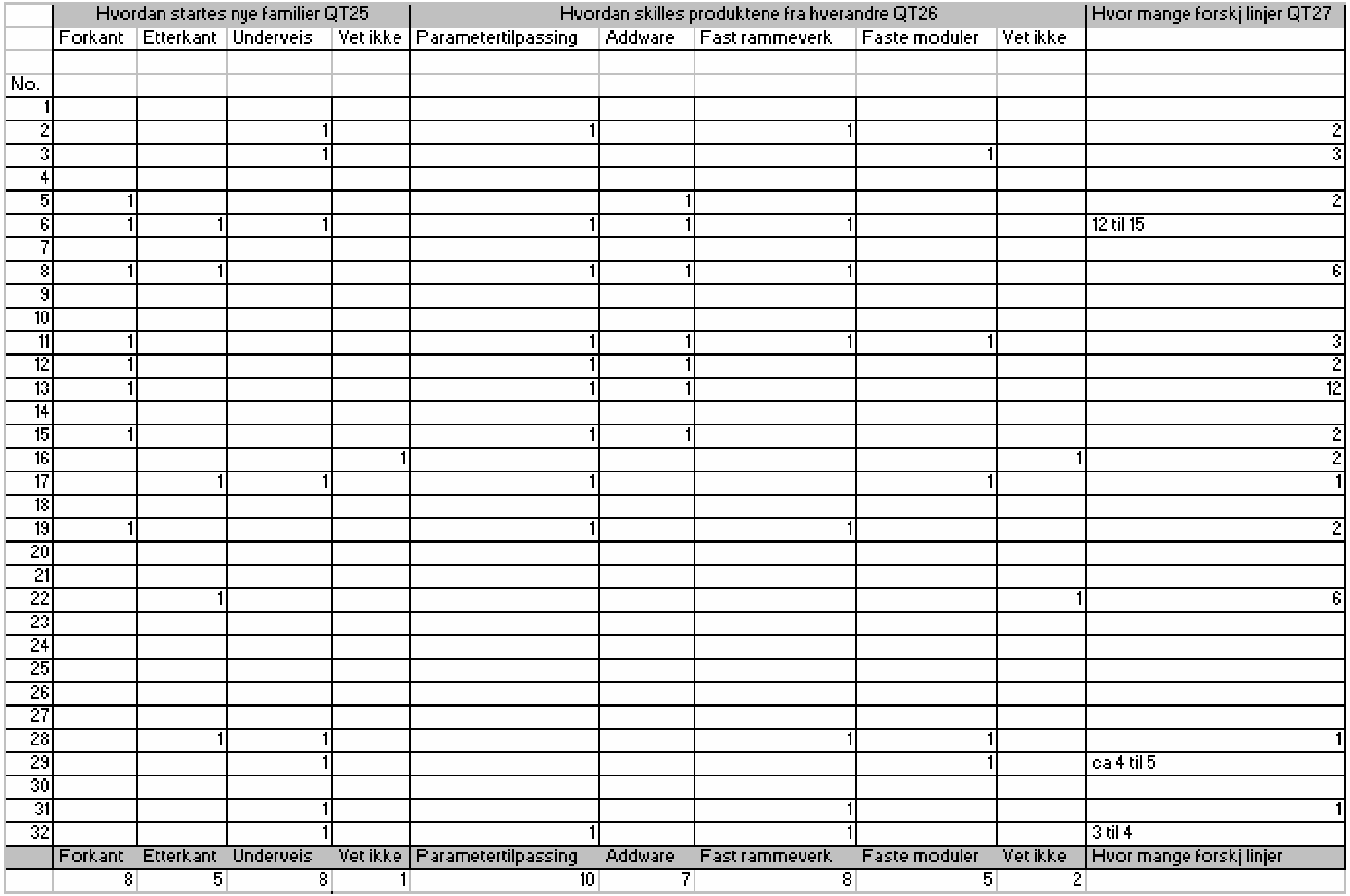
	Hvor viktig er begrunnelser for bedriften QT19					
	Lavere kost	Kortere utv tid	Bedre progkval	Mer std prod	Mindre vedlikehold	Annet
	1-6 der 1 er uviktig, 5 svært viktig og 6 vet ikke					
No.						
1	4	4	4	3	2	
2	5	5	5	3	5	Lettere opplæring av nyansatte: 5, Enklere vedlikehold av dokumentasjon:5, Enklere support av kunder og mer konsistente produktløsninger på tvers av kundemassen:6
3	4	4	4	4	4	
4	4	4	3	3	4	
5	4	4	4	1	4	
6	5	5	5	5	5	
7	4	4	4	5	5	
8	5	5	5	3	5	Redusert teknisk risk i prosjekter:5
9	5	5	4	4	4	
10	5	4	4	4	3	
11	6	1	1	6	1	
12	5	4	3	4	5	
13	5	5	5	3	5	
14	5	4	5	4	5	
15						
16	4	5	5	4	5	
17	2	3	5	4	2	
18						
19	4	4	4	3	4	
20						
21	2	4	3	4	5	
22	4	4	4	3	3	
23	4	5	5	4	5	
24	4	4	4	5	5	
25	5	5	4	3	3	
26	5	5	5	5	5	
27	3	4	5	3	3	Lavere opplæringsbehov:5, Lett å gå fra et prosjekt til et annet:5
28	4	3	5	4	4	
29	4	4	4	4	4	
30						
31	5	5	3	2	3	Større fleksibilitet:5
32	3	5	5	4	4	
	Lavere kost	Kortere utv tid	Bedre progkval	Mer std prod	Mindre vedlikehold	Annet
	Uviktig	Uviktig	Uviktig	Uviktig	Uviktig	4
	0	1	1	1	1	
	Litt viktig	Litt viktig	Litt viktig	Litt viktig	Litt viktig	
	2	0	0	1	2	
	Passe viktig	Passe viktig	Passe viktig	Passe viktig	Passe viktig	
	1	2	4	9	5	
	Meget viktig	Meget viktig	Meget viktig	Meget viktig	Meget viktig	
	12	14	11	11	7	
	Svært viktig	Svært viktig	Svært viktig	Svært viktig	Svært viktig	
	11	10	11	4	12	
	Vet ikke	Vet ikke	Vet ikke	Vet ikke	Vet ikke	
	1	0	0	1	0	
	Gjennomsnitt	Gjennomsnitt	Gjennomsnitt	Gjennomsnitt	Gjennomsnitt	
	4,35	4,37	4,33	3,77	4,15	



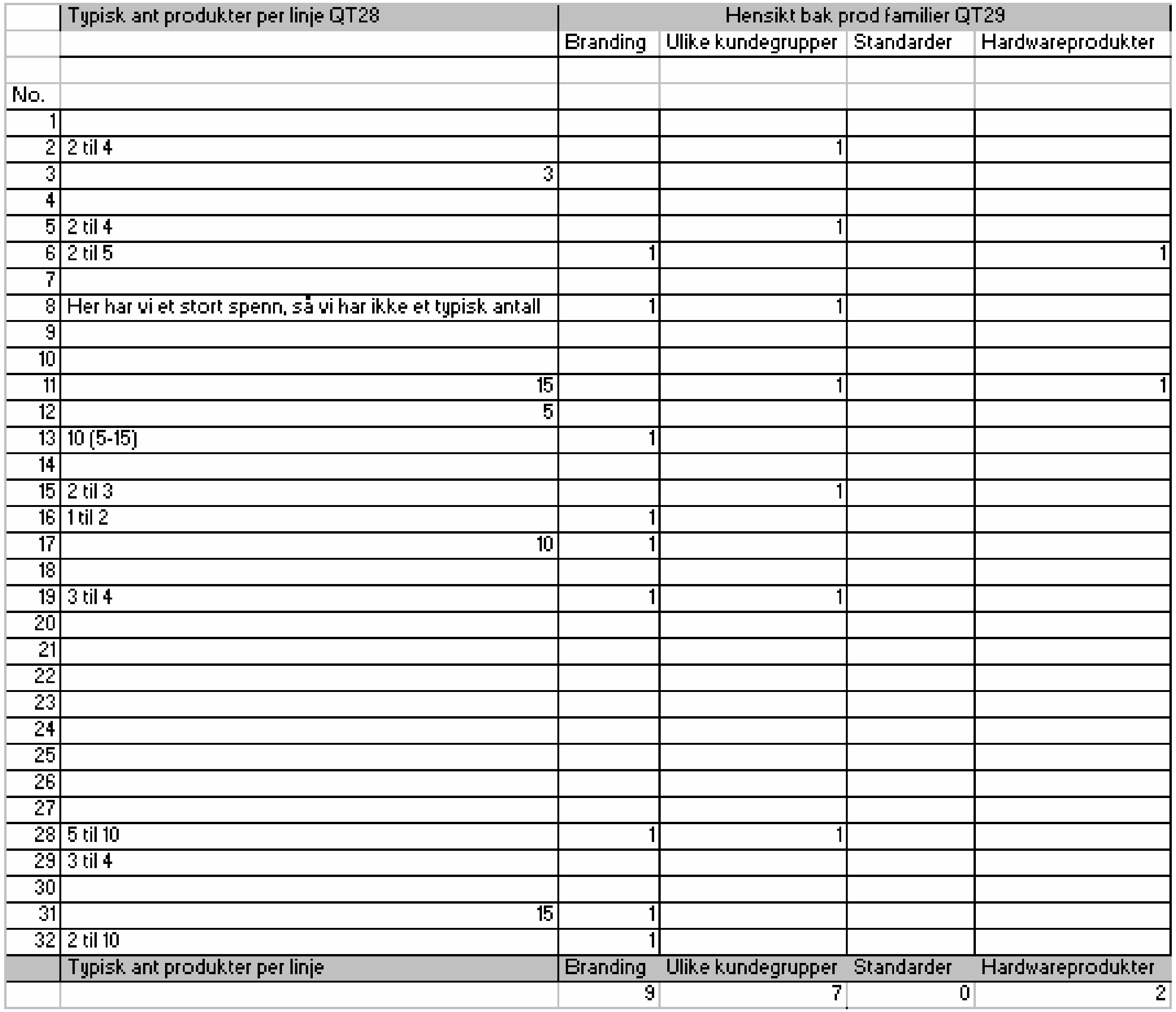
	Hvor viktig er suksessfaktorer for bedriften QT20										
	Intern utv pros	Utviklere ser nytten	Ansatte blir opplært	Egne personer	Belønning	Ledere ser nytten	Egnet arkitektur	Gen. komponenter	God dok. osv.	Versjonkontroll	Annet
	1-6 der 1 er uviktig, 5 meget viktig og 6 vet ikke										
No.											
1	1	5	3	1	2	2	3	3	1	3	
2											
3	4	4	3	3	3	4	5	5	4	5	
4	4	4	3	1	2	3	5	5	5	5	
5	2	4	2	1	1	3	4	3	4	5	
6	5	5	4	2	4	4	5	4	5	5	
7	3	5	3	1	1	4	4	4	2	3	
8	5	4	5	3	5	5	5	5	5	5	
9	5	5	5	4	4	5	4	4	3	4	
10	5	5	5	5	5	5	4	4	5	4	
11	4	3	2	1	3	4	4	4	4	4	
12	3	4	4	3	4	5	4	4	4	4	
13	5	5	3	1	1	5	5	5	5	5	
14	3	3	2	1	1	3	3	2	2	4	
15											
16	5	4	4	1	4	4	4	4	3	3	
17	3	5	4	2	3	4	4	4	4	5	
18											
19	3	4	4	2	3	4	4	4	4	4	
20											
21	4	4	2	2	2	4	4	2	3	4	
22	4	4	3	3	3	3	4	4	4	4	
23	3	5	4	1	2	4	5	5	5	5	
24	4	5	3	6	6	3	5	4	4	4	
25	4	5	4	3	3	4	5	4	4	4	
26	4	4	4	1	1	1	4	6	6	6	
27	4	4	4	2	2	3	5	5	4	4	
28	4	4	3	2	3	4	5	4	5	5	
29	6	4	3	4	4	4	4	4	4	4	
30											
31	5	3	4	5	2	4	5	5	2	4	
32	4	4	4	2	2	4	5	5	4	5	
	Intern utv pros	Utviklere ser nytten	Ansatte blir opplært	Egne personer	Belønning	Ledere ser nytten	Egnet arkitektur	Gen. komponenter	God dok. Osv.	Versjonkontroll	Annet
	Uviktig	Uviktig	Uviktig	Uviktig	Uviktig	Uviktig	Uviktig	Uviktig	Uviktig	Uviktig	0
	1	0	0	10	5	1	0	0	1	0	
	Litt viktig	Litt viktig	Litt viktig	Litt viktig	Litt viktig	Litt viktig	Litt viktig	Litt viktig	Litt viktig	Litt viktig	
	1	0	4	7	7	1	0	2	3	0	
	Passe viktig	Passe viktig	Passe viktig	Passe viktig	Passe viktig	Passe viktig	Passe viktig	Passe viktig	Passe viktig	Passe viktig	
	6	3	9	5	7	6	2	2	3	3	
	Meget viktig	Meget viktig	Meget viktig	Meget viktig	Meget viktig	Meget viktig	Meget viktig	Meget viktig	Meget viktig	Meget viktig	
	11	14	11	2	5	14	13	14	12	13	
	Svært viktig	Svært viktig	Svært viktig	Svært viktig	Svært viktig	Svært viktig	Svært viktig	Svært viktig	Svært viktig	Svært viktig	
	7	10	3	2	2	5	12	8	7	10	
	Vet ikke	Vet ikke	Vet ikke	Vet ikke	Vet ikke	Vet ikke	Vet ikke	Vet ikke	Vet ikke	Vet ikke	
	1	0	0	1	1	0	0	1	1	1	
	Gjennomsnitt	Gjennomsnitt	Gjennomsnitt	Gjennomsnitt	Gjennomsn	Gjennomsnitt	Gjennomsnitt	Gjennomsnitt	Gjennomsnitt	Gjennomsnitt	
	3,85	4,26	3,48	2,19	2,69	3,78	4,37	4,08	3,81	4,27	



	Produktfamilier QT21		Hvor lenge prod. fam. QT22	Felles i prod fam QT24						
	Ja	Nei		Branding	Krav	Arkitektur	Kode	Infrastruktur	Testsystem	Annet
No.										
1		1								
2	1		over 6	1		1	1	1	1	
3	1		3 til 4	1				1	1	
4		1								
5	1		3 til 4	1	1	1	1	1	1	
6	1		over 6	1		1		1		
7		1								
8	1		over 6	1	1	1	1	1	1	
9		1								
10		1								
11	1		over 6	1				1		
12	1		5 til 6		1	1	1	1	1	
13	1		over 6	1	1	1	1	1	1	
14		1								
15	1		3 til 4	1		1				
16	1		over 6			1	1			
17	1		over 6	1	1	1	1	1	1	1
18		1								
19	1		over 6	1		1	1	1		
20		1								
21		1								
22	1		over 6		1	1	1			
23		1								
24		1								
25		1								
26		1								
27		1								
28	1		5 til 6	1						
29	1		over 6				1			
30		1								
31	1		over 6	1		1		1		
32	1		3 til 4	1	1	1	1	1		
	Ja	Nei	under 1	Branding	Krav	Arkitektur	Kode	Infrastruktur	Testsystem	Annet
	17	15	0	13	7	13	11	12	7	1
			1 til 2							
			0							
			3 til 4							
			4							
			5 til 6							
			2							
			over 6							
			11							

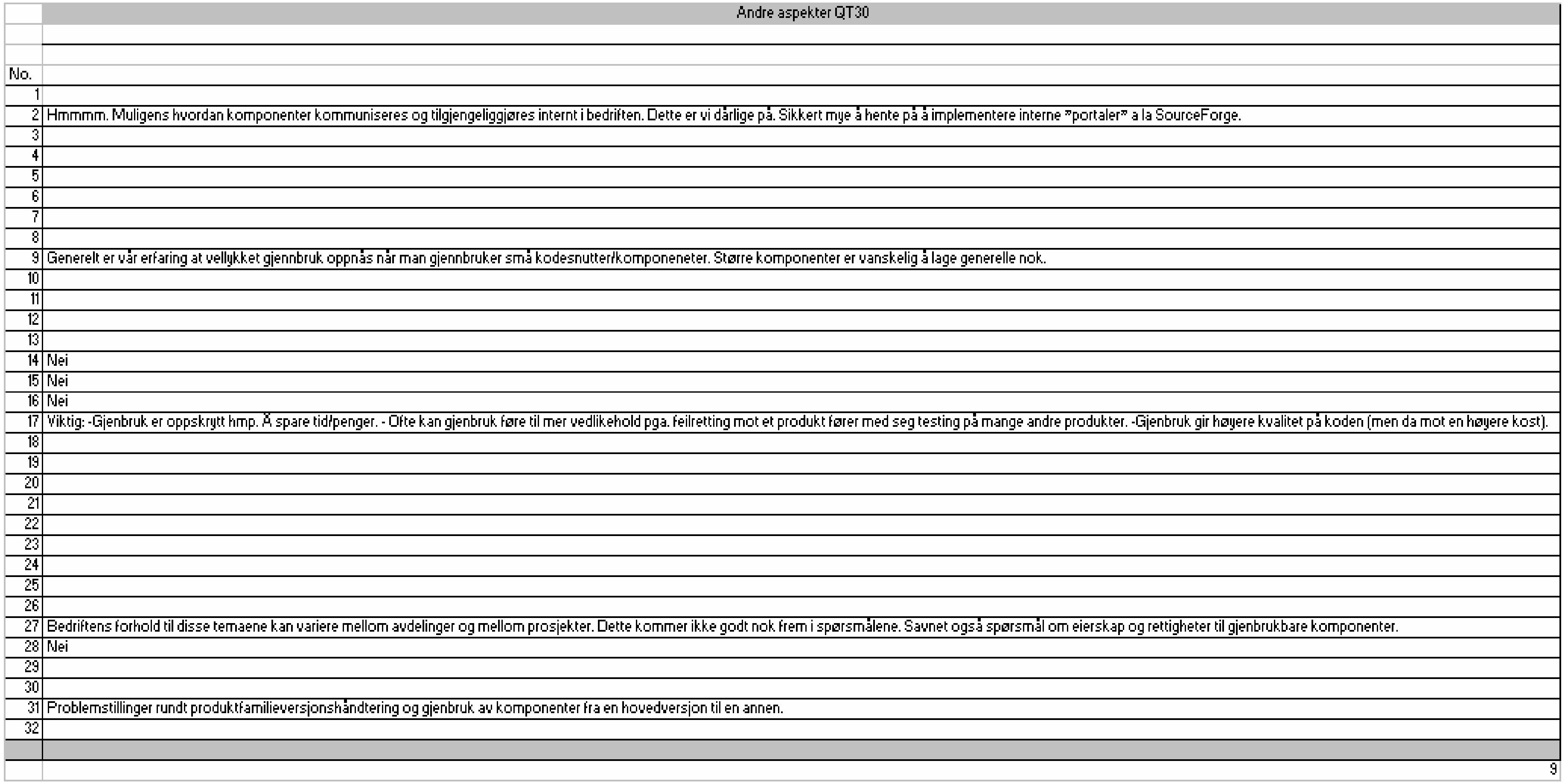


	Hvordan startes nye familier QT25				Hvordan skilles produktene fra hverandre QT26					Hvor mange forskj linjer QT27
	Forkant	Etterkant	Underveis	Vet ikke	Parametertilpassing	Addware	Fast rammeverk	Faste moduler	Vet ikke	
No.										
1										
2			1		1		1			2
3			1					1		3
4										
5	1					1				2
6	1	1	1		1	1	1			12 til 15
7										
8	1	1			1	1	1			6
9										
10										
11	1				1	1	1	1		3
12	1				1	1				2
13	1				1	1				12
14										
15	1				1	1				2
16				1					1	2
17		1	1		1			1		1
18										
19	1				1		1			2
20										
21										
22		1							1	6
23										
24										
25										
26										
27										
28		1	1				1	1		1
29			1					1		ca 4 til 5
30										
31			1				1			1
32			1		1		1			3 til 4
	Forkant	Etterkant	Underveis	Vet ikke	Parametertilpassing	Addware	Fast rammeverk	Faste moduler	Vet ikke	Hvor mange forskj linjer
	8	5	8	1	10	7	8	5	2	



	Typisk ant produkter per linje QT28	Hensikt bak prod familier QT29			
		Branding	Ulike kundegrupper	Standarder	Hardwareprodukter
No.					
1					
2	2 til 4		1		
3	3				
4					
5	2 til 4		1		
6	2 til 5	1			1
7					
8	Her har vi et stort spenn, så vi har ikke et typisk antall	1	1		
9					
10					
11	15		1		1
12	5				
13	10 (5-15)	1			
14					
15	2 til 3		1		
16	1 til 2	1			
17	10	1			
18					
19	3 til 4	1	1		
20					
21					
22					
23					
24					
25					
26					
27					
28	5 til 10	1	1		
29	3 til 4				
30					
31	15	1			
32	2 til 10	1			
	Typisk ant produkter per linje	Branding	Ulike kundegrupper	Standarder	Hardwareprodukter
		9	7	0	2

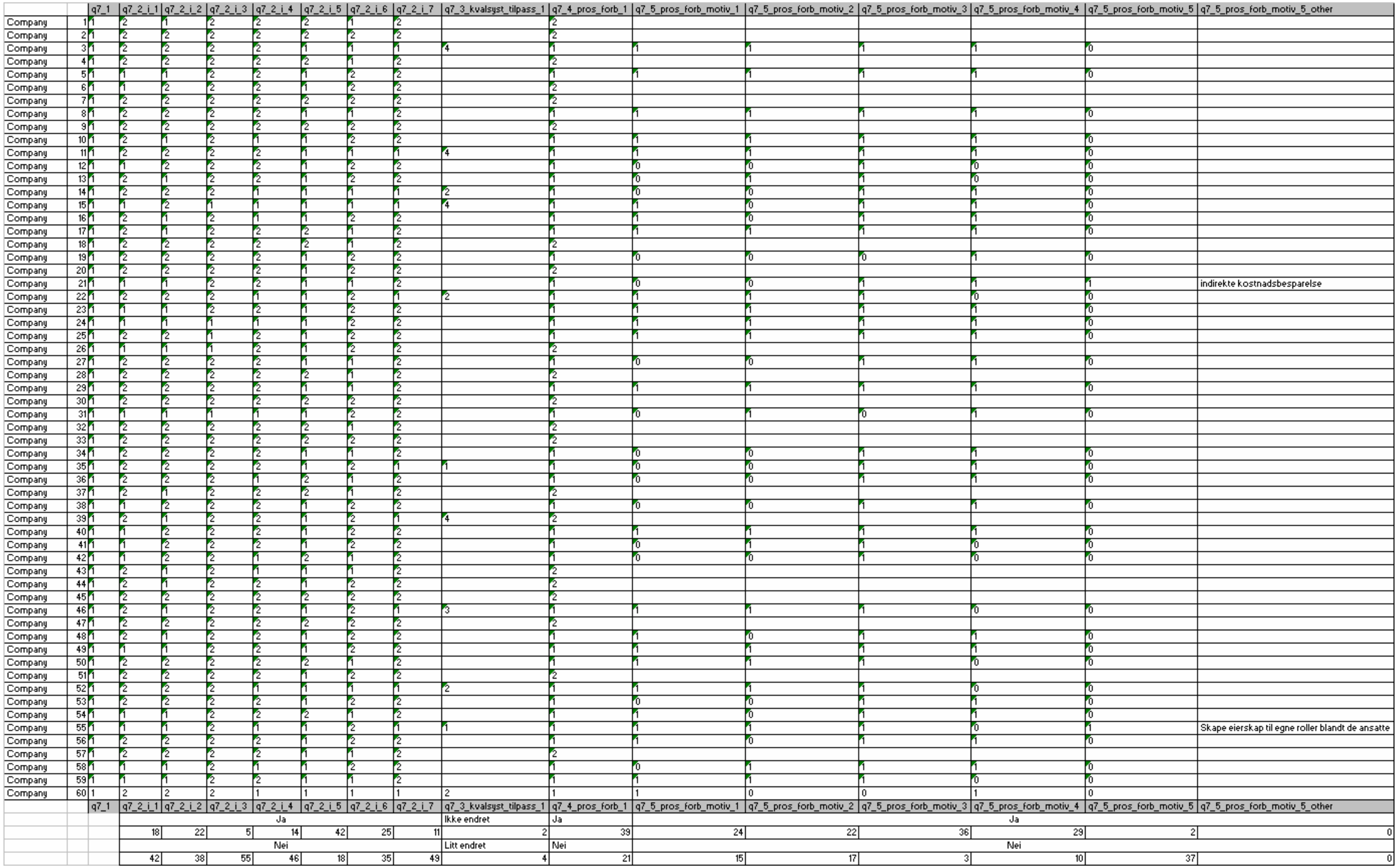
	Hensikt bak prod familier QT29 (forts.)
	Annet
No.	
1	
2	Pristaktisk. Ulike markeder eller segmenter innenfor samme marked. Ved å separere i ulike familier eller ulike varianter innenfor en familie så kan en etablere en mye mer fleksibel prismodell
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	støtte av flere operativsystem
13	Treffer forretningsområder/behov I markedet
14	
15	Gjenbruk av moduler på meta-nivå
16	
17	Brukeren kjøper de modulene han trenger
18	
19	
20	
21	
22	Ulike typer produkter
23	
24	
25	
26	
27	
28	
29	
30	
31	Modulene i rammeverket (eller en sammensetning av en eller flere av disse) kan benyttes til ulike formål avhengig av konfigurasjon, derfor ulike produkter innenfor en familie
32	
	Annet
	7



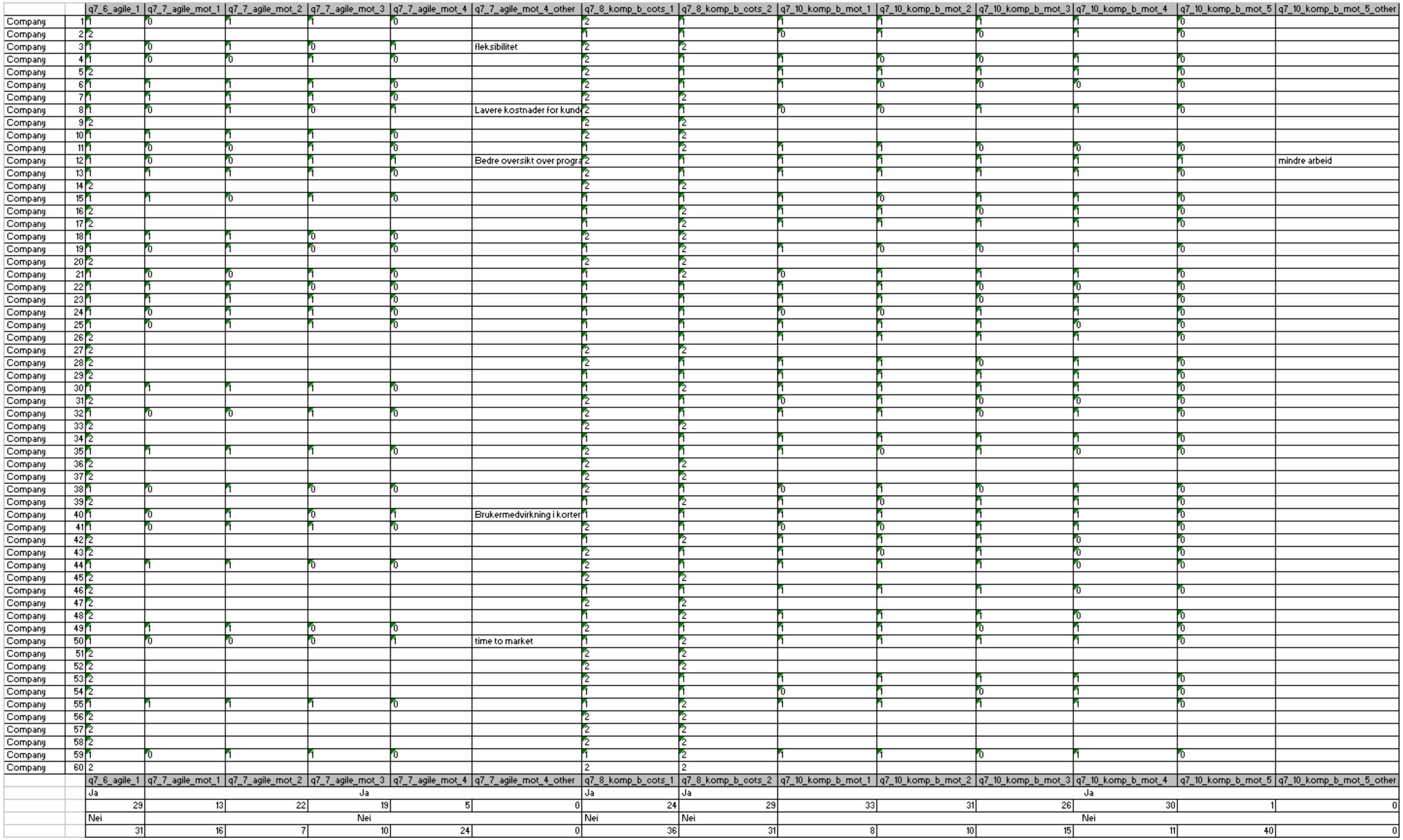
	Andre aspekter QT30
No.	
1	
2	Hmmmm. Muligens hvordan komponenter kommuniseres og tilgjengeliggjøres internt i bedriften. Dette er vi dårlige på. Sikkert mye å hente på å implementere interne "portaler" a la SourceForge.
3	
4	
5	
6	
7	
8	
9	Generelt er vår erfaring at vellykket gjennbruk oppnås når man gjennbruker små kodesnutter/komponeneter. Større komponenter er vanskelig å lage generelle nok.
10	
11	
12	
13	
14	Nei
15	Nei
16	Nei
17	Viktig: -Gjenbruk er oppskrytt hmp. Å spare tid/penger. - Ofte kan gjenbruk føre til mer vedlikehold pga. feilretting mot et produkt fører med seg testing på mange andre produkter. -Gjenbruk gir høyere kvalitet på koden (men da mot en høyere kost).
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	Bedriftens forhold til disse temaene kan variere mellom avdelinger og mellom prosjekter. Dette kommer ikke godt nok frem i spørsmålene. Savnet også spørsmål om eierskap og rettigheter til gjenbrukbare komponenter.
28	Nei
29	
30	
31	Problemstillinger rundt produktfamilieversjonshåndtering og gjenbruk av komponenter fra en hovedversjon til en annen.
32	
	9

Appendix D: Data from the IKT-Norge Survey

Here the data from the extra questions added by NTNU in the IKT-Norge survey are presented. The names of the questions have been shorted to make the tables more readable. The companies are numbered in the left colons on each page.



		q7_1	q7_2_i_1	q7_2_i_2	q7_2_i_3	q7_2_i_4	q7_2_i_5	q7_2_i_6	q7_2_i_7	q7_3_kvalsyst_tilpass_1	q7_4_pros_forb_1	q7_5_pros_forb_motiv_1	q7_5_pros_forb_motiv_2	q7_5_pros_forb_motiv_3	q7_5_pros_forb_motiv_4	q7_5_pros_forb_motiv_5	q7_5_pros_forb_motiv_5_other
Company	1	1	2	1	2	2	2	1	2		2						
Company	2	1	2	2	2	2	2	2	2		2						
Company	3	1	2	2	2	2	1	1	1	4	1	1	1	1	1	0	
Company	4	1	2	2	2	2	2	1	2		2						
Company	5	1	1	1	2	2	1	2	2		1	1	1	1	1	0	
Company	6	1	1	2	2	2	1	2	2		2						
Company	7	1	2	2	2	2	2	2	2		2						
Company	8	1	2	2	2	2	1	1	2		1	1	1	1	1	0	
Company	9	1	2	2	2	2	2	2	2		2						
Company	10	1	2	1	2	1	1	2	2		1	1	1	1	1	0	
Company	11	1	2	2	2	2	1	1	1	4	1	1	1	1	1	0	
Company	12	1	1	2	2	2	1	2	2		1	0	0	1	0	0	
Company	13	1	2	1	2	2	1	2	2		1	0	1	1	0	0	
Company	14	1	2	2	2	1	1	1	1	2	1	0	0	1	1	0	
Company	15	1	1	2	1	1	1	1	1	4	1	1	0	1	1	0	
Company	16	1	2	1	2	1	1	2	2		1	1	0	1	1	0	
Company	17	1	2	1	2	2	2	1	2		1	1	1	1	1	0	
Company	18	1	2	2	2	2	2	1	2		2						
Company	19	1	2	2	2	2	1	2	2		1	0	0	0	1	0	
Company	20	1	2	2	2	2	1	2	2		2						
Company	21	1	1	1	2	2	1	1	2		1	0	0	1	1	1	indirekte kostnadsbesparelse
Company	22	1	2	2	2	1	1	2	1	2	1	1	1	1	0	0	
Company	23	1	1	1	2	2	1	2	2		1	1	1	1	1	0	
Company	24	1	1	1	1	1	1	2	2		1	1	1	1	1	0	
Company	25	1	2	2	1	2	1	2	2		1	1	1	1	1	0	
Company	26	1	1	1	1	2	1	2	2		2						
Company	27	1	2	2	2	2	1	1	2		1	0	0	1	1	0	
Company	28	1	2	2	2	2	2	1	2		2						
Company	29	1	2	2	2	2	1	1	2		1	1	1	1	1	0	
Company	30	1	2	2	2	2	2	2	2		2						
Company	31	1	1	1	1	1	1	2	2		1	0	1	0	1	0	
Company	32	1	2	2	2	2	2	1	2		2						
Company	33	1	2	2	2	2	2	2	2		2						
Company	34	1	2	2	2	2	1	1	2		1	0	0	1	1	0	
Company	35	1	2	2	2	2	1	2	1	1	1	0	0	1	1	0	
Company	36	1	2	2	2	1	2	1	2		1	0	0	1	1	0	
Company	37	1	2	1	2	2	2	1	2		2						
Company	38	1	1	2	2	2	1	2	2		1	0	0	1	1	0	
Company	39	1	2	1	2	2	1	2	1	4	2						
Company	40	1	1	2	2	2	1	2	2		1	1	1	1	1	0	
Company	41	1	1	2	2	2	1	2	2		1	0	1	1	0	0	
Company	42	1	1	2	2	1	2	1	2		1	0	0	1	0	0	
Company	43	1	2	1	2	1	1	1	2		2						
Company	44	1	2	1	2	2	1	2	2		2						
Company	45	1	2	2	2	2	2	2	2		2						
Company	46	1	2	1	2	2	1	2	1	3	1	1	1	1	0	0	
Company	47	1	2	2	2	2	2	2	2		2						
Company	48	1	2	1	2	2	1	2	2		1	1	0	1	1	0	
Company	49	1	1	1	2	2	1	2	2		1	1	1	1	1	0	
Company	50	1	2	2	2	2	2	1	2		1	1	1	1	0	0	
Company	51	1	2	2	2	2	1	2	2		2						
Company	52	1	2	2	2	1	1	1	1	2	1	1	1	1	0	0	
Company	53	1	2	2	2	2	1	2	2		1	0	0	1	1	0	
Company	54	1	1	1	2	2	2	1	2		1	1	0	1	1	0	
Company	55	1	1	1	2	1	1	2	1	1	1	1	1	1	0	1	Skape eierskap til egne roller blandt de ansatte
Company	56	1	2	2	2	2	1	2	2		1	1	0	1	1	0	
Company	57	1	2	2	2	2	1	1	2		2						
Company	58	1	1	1	2	1	1	2	2		1	0	1	1	1	0	
Company	59	1	1	1	2	2	1	1	2		1	1	1	1	0	0	
Company	60	1	2	2	2	1	1	1	1	2	1	1	0	0	1	0	
		q7_1	q7_2_i_1	q7_2_i_2	q7_2_i_3	q7_2_i_4	q7_2_i_5	q7_2_i_6	q7_2_i_7	q7_3_kvalsyst_tilpass_1	q7_4_pros_forb_1	q7_5_pros_forb_motiv_1	q7_5_pros_forb_motiv_2	q7_5_pros_forb_motiv_3	q7_5_pros_forb_motiv_4	q7_5_pros_forb_motiv_5	q7_5_pros_forb_motiv_5_other
			Ja							Ikke endret	Ja	Ja					
			18	22	5	14	42	25	11	2	39	24	22	36	29	2	0
			Nei							Litt endret	Nei	Nei					
			42	38	55	46	18	35	49	4	21	15	17	3	10	37	0



		q7_6_agile_1	q7_7_agile_mot_1	q7_7_agile_mot_2	q7_7_agile_mot_3	q7_7_agile_mot_4	q7_7_agile_mot_4_other	q7_8_komp_b_cots_1	q7_8_komp_b_cots_2	q7_10_komp_b_mot_1	q7_10_komp_b_mot_2	q7_10_komp_b_mot_3	q7_10_komp_b_mot_4	q7_10_komp_b_mot_5	q7_10_komp_b_mot_5_other
Company	1	1	0	1	1	0		2	1	1	1	1	1	0	
Company	2	2						1	1	0	1	0	1	0	
Company	3	1	0	1	0	1	fleksibilitet	2	2						
Company	4	1	0	0	1	0		2	1	1	0	0	1	0	
Company	5	2						2	1	1	1	1	1	0	
Company	6	1	1	1	1	0		2	1	1	0	0	0	0	
Company	7	1	1	1	1	0		2	2						
Company	8	1	0	1	0	1	Lavere kostnader for kund	2	1	0	0	1	1	0	
Company	9	2						2	2						
Company	10	1	1	1	1	0		2	2						
Company	11	1	0	0	1	0		1	2	1	1	0	0	0	
Company	12	1	0	0	1	1	Bedre oversikt over progra	2	1	1	1	1	1	1	mindre arbeid
Company	13	1	1	1	1	0		2	1	1	1	1	1	0	
Company	14	2						2	2						
Company	15	1	1	0	1	0		1	1	1	0	1	1	0	
Company	16	2						1	2	1	1	0	1	0	
Company	17	2						1	2	1	1	1	1	0	
Company	18	1	1	1	0	0		2	2						
Company	19	1	0	1	0	0		1	2	1	0	0	1	0	
Company	20	2						2	2						
Company	21	1	0	0	1	0		1	2	0	1	1	1	0	
Company	22	1	1	1	0	0		1	1	1	1	0	0	0	
Company	23	1	1	1	1	0		1	1	1	1	0	1	0	
Company	24	1	0	1	1	0		1	1	0	0	1	1	0	
Company	25	1	0	1	1	0		1	1	1	1	1	0	0	
Company	26	2						1	1	1	1	1	1	0	
Company	27	2						2	2						
Company	28	2						2	1	1	1	0	1	0	
Company	29	2						1	1	1	1	1	1	0	
Company	30	1	1	1	1	0		1	2	1	1	1	1	0	
Company	31	2						2	1	0	1	0	0	0	
Company	32	1	0	0	1	0		2	1	1	1	0	1	0	
Company	33	2						2	2						
Company	34	2						1	1	1	1	1	1	0	
Company	35	1	1	1	1	0		2	1	1	0	1	0	0	
Company	36	2						2	2						
Company	37	2						2	2						
Company	38	1	0	1	0	0		2	1	0	1	0	1	0	
Company	39	2						1	2	1	0	1	1	0	
Company	40	1	0	1	0	1	Brukermedvirkning i korter	1	1	1	1	1	1	0	
Company	41	1	0	1	1	0		2	1	0	0	1	1	0	
Company	42	2						1	2	1	1	1	0	0	
Company	43	2						2	1	1	0	1	0	0	
Company	44	1	1	1	0	0		2	1	1	1	1	0	0	
Company	45	2						2	2						
Company	46	2						1	1	1	1	1	0	0	
Company	47	2						2	2						
Company	48	2						1	2	1	1	1	0	0	
Company	49	1	1	1	0	0		2	1	1	1	0	1	0	
Company	50	1	0	0	0	1	time to market	1	2	1	1	1	1	0	
Company	51	2						2	2						
Company	52	2						2	2						
Company	53	2						2	1	1	1	1	1	0	
Company	54	2						1	1	0	1	0	1	0	
Company	55	1	1	1	1	0		1	2	1	1	1	1	0	
Company	56	2						2	2						
Company	57	2						2	2						
Company	58	2						2	2						
Company	59	1	0	1	1	0		1	2	1	1	0	1	0	
Company	60	2						2	2						
		q7_6_agile_1	q7_7_agile_mot_1	q7_7_agile_mot_2	q7_7_agile_mot_3	q7_7_agile_mot_4	q7_7_agile_mot_4_other	q7_8_komp_b_cots_1	q7_8_komp_b_cots_2	q7_10_komp_b_mot_1	q7_10_komp_b_mot_2	q7_10_komp_b_mot_3	q7_10_komp_b_mot_4	q7_10_komp_b_mot_5	q7_10_komp_b_mot_5_other
		Ja	Ja					Ja	Ja	Ja					
		29	13	22	19	5	0	24	29	33	31	26	30	1	0
		Nei	Nei					Nei	Nei	Nei					
		31	16	7	10	24	0	36	31	8	10	15	11	40	0

q7_11_gjenbruk_1	q7_12_gjenbruk_hva_1	q7_12_gjenbruk_hva_2	q7_12_gjenbruk_hva_3	q7_12_gjenbruk_hva_4	q7_12_gjenbruk_hva	q7_12_gjenbruk	q7_12_gjenbruk	q7_13_gjenbruk_motiv_1	q7_13_gjenbruk_motiv_2	q7_13_gjenbruk_motiv_3	q7_13_gjenbruk_motiv_4	q7_13_gjenbruk	q7_13_gjenbruk
1	1	1	1	0	0	0		1	1	1	1	0	
2													
1	1	1	1	1	1	0		1	1	1	1	0	
1	1	1	1	0	1	0		1	1	1	1	0	
1	1	1	1	1	1	0		1	1	1	1	0	
1	0	0	1	0	0	0		1	1	0	0	0	
1	0	1	1	1	0	0		1	1	1	1	0	
1	0	1	1	1	1	0		1	0	1	1	0	
2													
1	1	1	1	1	1	0		1	1	1	1	0	
1	1	1	1	0	0	0		1	1	1	1	0	
1	0	1	1	0	1	0		0	0	1	1	0	
1	1	1	1	1	1	0		1	1	1	1	0	
1	1	1	1	0	1	0		1	0	1	1	0	
1	0	1	0	0	0	0		1	0	1	1	0	
1	1	1	1	1	1	0		1	1	1	0	0	
1	1	1	1	1	1	0		1	1	1	1	0	
1	0	1	1	1	0	0		1	1	1	0	0	
1	1	0	0	0	1	0		1	0	0	0	0	
2													
1	0	1	1	1	1	0		1	1	1	1	0	
1	0	0	1	0	0	0		1	1	0	0	0	
1	1	1	1	0	1	0		1	1	1	1	0	
1	1	1	1	1	1	0		1	1	1	1	0	
1	0	1	1	0	0	0		1	1	0	0	0	
1	1	1	1	1	1	0		1	1	1	1	0	
1	0	0	1	0	0	0		1	1	1	0	0	
2													
1	0	0	1	1	1	0		1	1	1	1	0	
1	0	0	1	1	1	0		1	1	1	0	0	
1	0	1	0	1	1	0		1	1	1	1	0	
1	1	1	1	0	0	0		1	1	1	1	0	
1	1	0	1	1	0	0		1	1	1	1	0	
1	0	0	1	1	1	0		1	1	1	1	0	
1	1	1	1	1	1	0		1	1	1	1	0	
1	0	0	1	0	0	0		1	1	0	1	0	
1	0	1	1	1	0	0		1	1	1	1	0	
1	0	0	1	0	0	0		1	1	1	0	0	
1	0	1	1	0	1	0		1	1	1	1	0	
1	1	1	1	1	1	0		1	1	1	1	0	
1	0	0	1	1	1	0		0	1	1	1	0	
1	1	1	1	1	0	0		1	1	1	0	0	
1	1	1	0	1	1	0		0	1	1	0	0	
1	0	1	1	0	1	0		1	1	1	1	0	
1	0	1	1	0	0	0		0	1	0	0	0	
1	0	1	1	0	1	0		1	1	1	1	0	
1	0	0	1	1	1	0		1	1	1	0	0	
1	1	1	1	1	1	0		1	1	1	1	0	
1	0	1	1	0	0	0		1	1	1	1	0	
1	0	0	1	0	0	0		1	1	1	1	0	
1	1	1	0	1	1	0		0	0	1	0	0	
1	0	1	1	0	0	0		1	1	1	1	0	
1	0	1	1	1	0	0		1	1	1	1	0	
2													
1	1	1	1	1	1	1	Forretningslog	1	1	1	1	1	Enklere vedlike
1	1	0	0	1	1	0		1	0	1	1	0	
1	0	1	1	1	1	0		1	1	1	1	0	
1	0	1	1	0	0	0		0	1	1	1	0	
1	0	0	1	1	0	0		1	1	1	1	0	
1	1	0	1	1	0	0		1	1	1	1	0	
q7_11_gjenbruk_1	q7_12_gjenbruk_hva_1	q7_12_gjenbruk_hva_2	q7_12_gjenbruk_hva_3	q7_12_gjenbruk_hva_4	q7_12_gjenbruk_hva	q7_12_gjenbruk	q7_12_gjenbruk	q7_13_gjenbruk_motiv_1	q7_13_gjenbruk_motiv_2	q7_13_gjenbruk_motiv_3	q7_13_gjenbruk_motiv_4	q7_13_gjenbruk	q7_13_gjenbruk
Ja	Ja							Ja					
55	25	39	49	32	32	1	0	49	48	49	41	1	
Nei	Nei							Nei					
5	30	16	6	23	23	54	0	6	7	6	14	54	

		q7_14_outsc_1	q7_15_outsc_r_1	q7_15_outsc_r_2	q7_15_outsc_r_3	q7_15_outsc_r_4	q7_15_outsc_r_5	q7_15_outsc_r_6	q7_15_outsc_r_7	q7_15_outsc_r_8	q7_15_outsc_r_9	q7_15_outsc_r_9_o	q7_16_outsc_mot_1	q7_16_outsc_mot_2	q7_16_outsc_mot_3	q7_16_outsc_mot_4	q7_16_outsc_mot_5	q7_16_outsc_mot_5_o
Company	1	2																
Company	2	2																
Company	3	2																
Company	4	2																
Company	5	1	1	0	0	0	0	0	0	0	0		1	0	0	0	0	
Company	6	2																
Company	7	2																
Company	8	1	1	0	0	0	0	0	0	0	0		0	1	0	0	0	
Company	9	2																
Company	10	1	1	0	0	0	0	0	0	0	0		1	1	0	0	0	
Company	11	1	1	0	1	0	0	0	0	0	1	sør-amerika	1	0	0	0	1	Tilgang til ressurser
Company	12	2																
Company	13	2																
Company	14	2																
Company	15	1	0	0	1	1	1	1	1	0	0		1	0	1	0	0	
Company	16	2																
Company	17	1	1	0	0	0	0	0	0	0	0		1	0	0	0	0	
Company	18	2																
Company	19	2																
Company	20	1	1	0	0	0	0	0	0	0	0		0	1	0	0	0	
Company	21	1	1	0	0	0	0	0	0	0	0		0	0	0	0	1	Kapasitetsspørsmål
Company	22	2																
Company	23	1	0	0	0	0	0	1	0	0	1	Vietnam	1	1	0	0	1	Mulighet for å skalere raskere, samt dårlig tilgang på kvalifiserte ressurser i Norge.
Company	24	1	1	0	1	0	0	0	1	0	0		0	0	1	1	0	
Company	25	1	1	0	1	0	0	1	0	0	0		1	1	0	0	1	Kapasitet
Company	26	1	1	0	0	0	1	0	0	0	0		1	0	0	0	0	
Company	27	1	0	0	0	0	1	0	0	0	0		1	0	0	0	0	
Company	28	1	1	0	0	0	0	0	0	0	0		1	0	0	0	0	
Company	29	1	1	0	0	0	0	0	0	0	0		0	1	0	0	0	
Company	30	1	0	0	0	0	1	0	0	0	0		1	1	0	0	1	mere fleksibel arbeidskraft uten langvarige forpliktelser, raske endringer i bemanningen=kostnadskontroll
Company	31	1	1	0	0	0	0	0	0	0	0		0	0	0	1	0	
Company	32	2																
Company	33	2																
Company	34	2																
Company	35	2																
Company	36	2																
Company	37	2																
Company	38	2																
Company	39	2																
Company	40	2																
Company	41	2																
Company	42	2																
Company	43	1	0	0	1	0	0	0	0	0	0		0	1	0	0	0	
Company	44	2																
Company	45	2																
Company	46	1	1	1	0	0	0	0	0	0	0		1	0	0	0	0	
Company	47	2																
Company	48	1	1	0	0	0	1	0	0	0	0		1	1	1	0	0	
Company	49	2																
Company	50	2																
Company	51	2																
Company	52	2																
Company	53	1	0	0	0	1	1	0	0	0	0		1	1	1	1	0	
Company	54	2																
Company	55	2																
Company	56	1	1	0	0	0	0	0	0	0	0		1	0	1	1	0	
Company	57	2																
Company	58	2																
Company	59	1	0	0	0	0	1	0	0	0	0		1	1	0	0	1	tilgang på personell
Company	60	1	1	0	0	0	0	0	0	0	0		1	0	0	0	0	
		q7_14_outsc_1	q7_15_outsc_r_1	q7_15_outsc_r_2	q7_15_outsc_r_3	q7_15_outsc_r_4	q7_15_outsc_r_5	q7_15_outsc_r_6	q7_15_outsc_r_7	q7_15_outsc_r_8	q7_15_outsc_r_9	q7_15_outsc_r_9_o	q7_16_outsc_mot_1	q7_16_outsc_mot_2	q7_16_outsc_mot_3	q7_16_outsc_mot_4	q7_16_outsc_mot_5	q7_16_outsc_mot_5_o
		Ja	Ja										Ja					
		24	17	1	5	2	7	3	2	0	2	0	17	11	5	4	6	0
		Nei	Nei										Nei					
		36	7	23	19	22	17	21	22	24	22	0	7	13	19	20	18	0