Aalto University
School of Business

Implementation Strategies in Software Development - Case XBRL

MSc program in Information and Service Management

Master's thesis

Antti Karjalainen

2016

Department of Information and Service Economy
Aalto University
School of Business

Aalto University
School of Business

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi

Abstract of master's thesis

Author Karjalainen Antti

Title of thesis Implementation Strategies in Software Development – Case XBRL

Degree Master of Science in Economics and Business Administration

Degree programme Information and Service Management

Thesis advisor(s) Penttinen Esko

Year of approval 2016 **Number of pages** 72 **Language** English

Abstract

The main objective of this study is to identify and describe the ways in which companies approach implementing inter-organizational software functionalities into their software products and services and to identify the factors that affect their choice of a software implementation strategy.

The study is a descriptive multiple case study that focuses on four companies that are all doing a similar XBRL functionality implementation at the same time. The theoretical framework of the study consists of multiple factors, out of which network effect and path dependence are the most prominent theories that are used to guide the case study design. The reviewed literature categorizes software implementation strategies according to how deeply the new functionality is integrated into existing systems, categorizing software implementations into bolt-on, built-in, and deeply embedded implementation strategies. The findings from the case study indicate support for this kind of categorization in the observed cases, although the categories should be considered to be more of a continuum than set of discrete classes.

The results of the case study also indicate that, in the cases that were observed, network effects and path dependence play a role in the software implementation strategy selection but a significant factor in the implementation strategy decisions is also the uncertainty of the future adoption, development, and use of the technology in question. In an inter-organizational context, companies weigh in the perceived benefits of a software implementation against their confidence in the future use of that technology, forming an estimate of the expected value of the future benefit of the software implementation. This expected value will then in turn factor into the software implementation strategy decision and partly contribute to how deeply the company is willing to integrate the functionality into their existing systems.

Keywords software implementation strategy, inter-organizational software functionality, network effect, path dependence, XBRL

Acknowledgements

I would like to thank my supervisor Esko Penttinen for the great help and guidance with this project. I would also like to thank all the people in the case organizations that participated into this study and provided excellent insights without which this study would not have been possible.

Lastly, I would like to thank my family that has been very supportive for me during my studies, and especially my two little girls, Kukka and Kiia, and my wife, Alisa.

Table of Contents

List of Figures

List of Tables

1 Introduction

This thesis studies the different strategies that companies use when approaching software implementation project. More specifically, the focus is on software implementation projects that include inter-organizational aspects, such as pulling in information from outside of the organization or publishing information to external parties through a commonly agreed interface and by using a standard way to represent data. This topic is studied through multiple case studies of Finnish companies that are implementing support for a new data reporting standard in the financial administration and accounting domain.

In this introductory section, first the background and the motivation of the study are presented. Second, the research question and objectives are defined followed by introduction to key terminology that is used in this thesis. Lastly, a central technology that is used in the case studies, XBRL (eXtensible Business Reporting Language), is introduced in brief.

1.1 Background and motivation

Submitting XBRL reports that are based on Finnish accounting law and the standard business reporting (SBR) code set becomes available in Finland in April 2016. With the new standard, companies will have an option to start reporting, among other data, their financial statements and annual tax statements in a completely electronic form. This new reporting method will require that the information systems that are used to generate these reports will implement support for the new Finnish XBRL/SBR taxonomy. The required software addition is a fairly complex piece of functionality that incorporates translating accounting data into XBRL format, validating it, and sending it out to the receiving systems of the tax administration and the patent and registration office.

Software implementation strategies for inter-organizational functionalities are affected by many complex factors such as the state of data interchange standards, network externalities, and the internal capabilities of the implementing organization. In order to make knowledgeable decisions in software projects that involve inter-organizational functionalities, it is good to understand what kinds of implementation strategies exist, what kinds of factors affect the selection of an implementation strategy, and what the expected differences between the different strategies are. The adoption of XBRL/SBR taxonomy based reporting in Finland provides a unique opportunity to study these kinds of different software implementation

strategies that companies are choosing when building new inter-organizational functionalities into their products.

1.2 Research Question and Objectives

The research problem is to identify and describe the ways in which companies approach implementing inter-organizational functionalities into their software products and services and to identify the factors that affect their choice of a software implementation strategy.

In the age of information sharing and interconnection, companies frequently develop new software functionalities that incorporate inter-organizational aspects and the inter-organizational nature of software is only going to only increase in the future. The ways in which companies approach their implementation projects and the strategies they choose for them is guided by multiple different factors, some internal some external. The aim of this study is to provide insight into the general classification of software implementation strategies for inter-organizational functionality through the study of representative cases of XBRL/SBR implementation. The aim is to also to identify the factors that affect the selection of software implementation strategy and to explore what challenges and benefits each identified strategy has.

1.3 Terminology

Inter-organizational software functionality: Software functionality or feature that involves integrating external functionality to an organization's own software application or service. This can be done either through web services, remote APIs (application programming interface), or similar type of interfaces. Examples of an inter-organizational software functionality are e.g. an online banking functionality in a financial administration software package (pulling in banking functionality through a bank's remote API) or a 3rd party provided authentication method in a consumer facing mobile application (calling e.g. Google's or Facebook's web service interfaces to provide user authentication). Often in order to communicate with the external services, a standard way to represent data such as XML, JSON, or SOAP is used.

Software implementation strategy: The overall strategy that an organization uses when approaching the task of creating new software functionality. This term is close to, but should not be confused with software implementation methods or software adoption strategies. The former of these terms means the actual technical methodology of engineering new software

that can for example be an agile method, such as Scrum or SAFe (scaled agile framework), or an organization specific software project management methodology, such as PLM Value Delivery Method that is used in Siemens PL or ASAP delivery method that is used in SAP (Varadaraj & Goud, 2012). The latter term on the other hand refers to transferring from an old system to a new system. The main strategies involved in such projects are big bang, phased, parallel, and process line as summarized by Malhotra & Temponi (2010).

1.4 XBRL

Understanding what XBRL is and how it works is important when assessing the software implementation strategies in the XBRL projects that are described in the case studies in section 5.1. In the introduction of an XBLR special issue of the Journal of Information Systems, XBRL is described as "open standard-based reporting language that allows companies to electronically report and exchange financial and nonfinancial information in a standardized, machine-readable format". XBRL is designed to facilitate reporting processes on many different levels of an organization. It can be used to handle, analyze and understand data that comes from various sources, in different languages, or from different accounting standards. On technical level XBRL is an XML (eXtensible Markup Language) extension that consists of machine-readable tags for individual data elements. (Srivastava & Liu, 2012)

XBRL allows the creation of reusable taxonomies that describe all the meaning contained in reporting terms. These taxonomies can be developed by regulators and government agencies. XBRL also supports business rules that can be used to constrain what can be reported. This allows for an automated method to enforce data quality in reports already when a report is being created. (XBRL International, 2016)

XBRL is being used worldwide in multiple different types of applications. In the U.S. and Asia, XBRL is used in the capital markets while in Europe it is also used in governmentwide and cross-border applications (Kernan, 2008). In Finland, XBRL is currently used by financial institutions for filing Common Reporting (COREP) reports that are mandated by the European Banking Authority, EBA (Moody's Analytics, 2011).

More recently, a new XBRL taxonomy has been developed based on Finnish accounting law and the standard business reporting (SBR) code set that is intended to be used by companies to report their financial statements to the Finnish Tax Administration and the Finnish Patent and Registration Office. Submitting XBRL reports that are based on the XBRL/SBR taxonomy is available starting from April 2016. Companies that are in the front line of

implementing this new taxonomy are accounting information system (AIS) vendors and companies that offer financial administration services. (Finnish Tax Administration, 2015)

1.5 Structure of the Thesis

This thesis is structured as follows. Chapter 2 presents a literature review on key theories that are used to frame the study. In Chapter 3, a theoretical framework is formulated based on the reviewed literature that is then used in Chapter 4 to present the design of the case study, along with discussion on study methodology selection and definition. The results from the case studies are presented in Chapter 5 and lastly, the conclusions of the study are presented in Chapter 6.

2 Literature Review

There is very little existing research on software implementation strategies outside of very specific domains such as implementing large-scale ERP (enterprise resource planning) systems. The adoption of EDI (electronic data interchange) systems is widely studied but the actual software implementation aspects of such projects are usually overlooked. More general project managerial aspects of software implementation projects, e.g. success and risk factors, are also widely studied. Similarly, the network effects that affect platform innovations, including the XBRL standard, have been extensively studied in previous literature.

In this section, a literature review on relevant theories regarding software implementation strategies and inter-organizational software functionality are presented.

2.1 Software Implementation Strategies

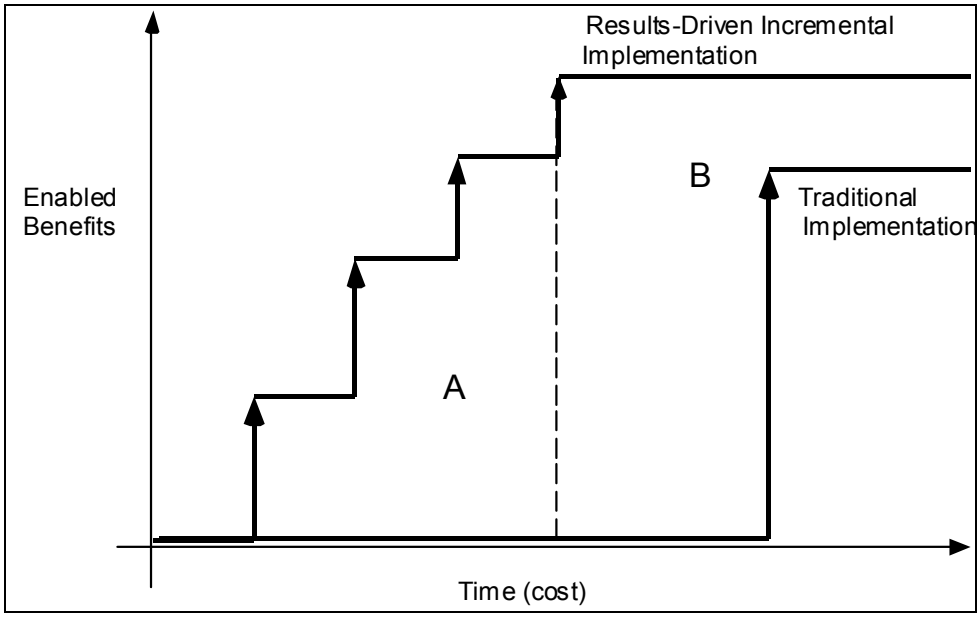
General literature on the topic of software implementation strategies in not widely published. Plenty of studies can be found on the adoption of new software, the actual engineering methodologies of implementing software, and in general technical software architecture and design. However, in the context of this study, the term software implementation strategy is used to indicate the overall strategy that and organization uses when approaching the task of creating new software functionality (see Section 1.3). Next, existing literature that explores topics that are close to the definition of software implementation strategy used here are reviewed.

In the world of agile software development, iterative and incremental development (IID) approach to software implementations is often considered as the best practice (Petersen & Wohlin, 2010). Based on the previously discussed software implementation strategy definition, the IID approach can also be seen as a software implementation strategy and not only as a method of organizing software development work. If the organization in question is knowledgeable enough, it can make a conscious strategy decision and choose to implement software incrementally instead of attempting to fulfill every requirement all at once.

In their article, Fichman & Moses (1999) introduce a software implementation strategy called the “results-driven incremental” (RDI) strategy in which software implementation is divided into a series of self-contained segments that each achieves a measurable business result. This implementation strategy can reduce scope creep and over-engineering by keeping the small and manageable sub-goals constantly visible. It will also help in maintaining

momentum in the implementation project by offering the team reoccurring achievements of visible results. Figure 1 illustrates the difference between the traditional approach and the RDI approach to software implementation. (Fichman & Moses, 1999)

Figure 1, Results-Driven Incremental vs. Traditional Implementation. (Fichman & Moses, 1999)



In their article, Fichman & Moses mix discussion between software adoption and implementation and they present the RDI method in the context of packaged software configuration projects. However, the conclusions from the RDI method are clearly extendable to more project that focus on developing new software as well and the incremental approach can be considered as one category of software implementation strategies.

In the context of XBRL, Garbellotto discusses in his three article series multiple approaches that an organization can take when implementing XBRL functionality. Since XBRL implementation projects can be seen as normal software development undertakings, the findings of Garbellotto can be extended to other software domains, at least with certain reservations. Garbellotto three strategies are called bolt-on, built-in, and deeply embedded (Garbellotto, 2009a; Garbellotto, 2009b; Garbellotto, 2009c). Each of these three strategies is described here briefly along with the associated level of cost, benefits, and limitation.

The bolt-on approach is just as it sounds, an afterthought. In this strategy, XBRL functionality is added by the means of data post-processing, either by using Excel spreadsheets or an external tool. The cost of the initial implementation is low but the indirect cost of maintenance might be relatively high. The main benefits of this strategy is that it can produced immediate results and the work is easily outsourced to a third party since there is a

clear interface from the main application to the bolt-on functionality. The main limitation of the bolt-on strategy is that the resulting implementation will require a lot of maintenance if the underlying requirements evolve over time. Garbellotto concludes that a bolt-on solution might be a justified way for an organization to buy more time, to do a proof-of-concept, before implementing a more long-term solution. (Garbellotto, 2009a)

A more lasting XBRL implementation strategy is the built-in approach. In this approach, the reporting application will understand the data that the XBRL report will require and it will be able to present it by using the appropriate abstractions and semantics. Whereas in the bolt-on approach XBRL reporting might be an additional cost for the organization, using the built-in strategy might actually lead into cost saving as is proposed by Stantial in his article "Roi on XBRL" (Stantial, 2007). According to Garbellotto, the built-in approach is easier to maintain than a bolt-on solution but it also lacks certain capabilities and benefits that are only achieved by going even further in the level of integration. (Garbellotto, 2009b)

The deeply embedded XBRL implementation strategy goes beyond having XBRL inside the reporting system and actually makes the XBRL format itself an integral part of the application's architecture. A standard way of representing data, such as the XBRL, enables aggregating, sharing, validating and analyzing data in ways that would otherwise be unfeasible. Garbellotto however points out that this type of implementation strategy is something that should be approached gradually over a long period of time, going from process to process, instead of rushing into replacing all existing IT infrastructure with XBRL enhanced versions. (Garbellotto, 2009c)

2.2 Network Effects

Network effect, also often called network externality, in simplicity means that the utility that a user derives from a product or a service is dependent on the number of other agents using the same product (Katz & Shapiro, 1985). These types of dependencies are abundant in modern day world of inter-organizational systems (IOS) and open communication standards (Zhu, Kraemer, Gurbaxani, & Xin Xu, 2006). Katz and Shapiro note the importance of compatibility in determining the scope of the network that creates network externalities (Katz & Shapiro, 1985). They also note that compatibility tends to be undersupplied by the market and emphasize the role that a sponsor, who is willing to make investments in order to promote a technology, can have in the adoption of the technology in the presence of network externalities (Katz & Shapiro, 1986).

In their 1985 article, Katz & Shapiro formulate a mathematical model of a market that is affected by network effects and identify three different sources for positive consumption externalities. A direct network effect means that the number of users is directly related to the usefulness of a product. The telephone is a good example of a product for which the value of the core feature has a utility that is directly linked to the number of users on the telephone network. An indirect network effect on the other hand is something that is dependent on the number of users but not in a directly linked manner. Katz & Shapiro use the personal computer as an example of this category; the PC will be more valuable to its users when more people are using it, not because any direct mechanism, but rather through the increase of compatible software products that are available on the market. Thirdly, Katz & Shapiro note a category of network effects for durable goods that results from the size of the service network for that product. A car for instance has more value for the user if it has an extensive service network – something that is only possible for a brand that has a lot of users. (Katz & Shapiro, 1985)

Based on their empirical research, Zhu et al. conclude that network effects and expected benefits are significant drivers behind migration to open-standard inter-organizational systems (Zhu et al., 2006). The same study also shows that the strength of the network effects in open-standard IOS adoption is determined by the strength of the community that is using the same technology. These results indicate that network effects should be examined closely when assessing any choice made by an individual company about the adoption, integration, or development of functionality that is affected by inter-organizational aspects, especially when that functionality is mediated by an open standard.

2.3 Path Dependence

Path dependence in essence explains how, for a given set circumstances, the future set of decisions is limited by the decisions that have been made previously, even though the circumstances that governed those decisions would no longer be relevant. In the context of innovations, technology and organizations this concept is intuitively easy to understand but it can lead into counter-intuitive scenarios where a company that has advanced technological infrastructure can end up in economically inferior innovation diffusion. Farrell and Saloner (1985) explain how standardization benefits can trap an industry in an obsolete or inferior standard even when there exists a better alternative. In their model, Farrell and Saloner

explain that this "excess inertia" is a result of incomplete-information combined with lack of coordination or consensus for the adoption of the new standard.

Zhu et al. show in their empirical study how migration to open-standards IOS is path dependent (Zhu et al., 2006). Previous technology adoption path will determine the marginal cost of adopting a new technology, define the marginal utility of the technology, and also define the level of managerial knowledge on similar projects inside the organization. This result is in line with the views of Dosi who argues that technical progress follows certain "technological trajectories" that determine the scope of possible new development (Dosi, 1982). Path dependency is also explained by Cohen and Levinthal who analyze a firm's "absorptive capacity", that is the ability to recognize, assimilate and apply new external information, and conclude that this capability is largely determined by prior related knowledge (Cohen & Levinthal, 1990).

Path dependence can be an explaining factor when determining why a company has adopted a certain technology that would in a static analysis seem to be inferior to competing choices. A dynamic approach that acknowledges the effects of historical events on present-day decisions will be able to describe a technical decision more accurately and, for example, explain situations where a company is locked-in to an inferior technology path (Arthur, 1989). This type of point-of-view is important when determining the factors that may affect a company's choice of technological approach to a given challenge. The reasons behind the choice may be either purely technical, for example interoperability of software standards, or driven by human factors like the familiarity with previously tested technologies, or a combination of both. An important thing to notice is that, in addition to affecting the scope of available decisions, path dependence may also affect the perceived marginal cost and utility when a company is considering a new technology.

2.4 Risk and Success Factors in Software Projects

Success factors and risks in projects are a widely researched topic and understanding them is important when making of observations on the progression of any type of project, including software projects. In the context of software implementation strategies, it is relevant to understand whether the outcome of a software implementation project is described as either a success or a failure due to the actual choice of the implementation strategy or the realization of critical success or risk factors.

Describing definitive success factors for a given type of project is difficult since they are contingent on the actual definition of the success of the project. There is no clear consensus on how project success should be judged and what factors are the most crucial for project success (Wateridge, 1995). However, in their paper Fortune and White provide an interesting meta-analysis of project success factors across 63 different publications that gives a broad sense of the most commonly regarded generic project success factors in academic publications (Fortune & White, 2006). The thirteen highest-ranking success factors are presented in Table 1.

Table 1, Critical Success Factors for Projects Fortune & White, 2006

Critical success factor	**Count of citations**
Support from senior management	39
Clear realistic objectives	31
Strong/detailed plan kept up to date	29
Good communication/feedback	27
User/client involvement	24
Skilled/suitably qualified/sufficient staff/team	20
Effective change management	19
Competent project manager	19
Strong business case/ sound basis for project	16
Sufficient/well allocated resources	16
Good leadership	15
Proven/familiar technology	14
Realistic schedule	14

When examining the literature on software projects specifically, one important factor that is not explicitly present in Fortune and White's list emerges; namely the importance of requirements planning and management (Berntsson-Svensson & Aurum, 2006; Hofmann & Lehner, 2001; Reel, 1999). Requirements might be considered in some ways part of project objectives and planning, so they are, in a way, implicitly present in Table 1 in the second and third most frequently mentioned success factors. Also, an empirical study by Dvir et al. suggests a strong positive relationship between project success and "the amount of effort invested in defining the goals of the project and the functional requirements and technical specifications of the product at hand" (Dvir, Raz, & Shenhar, 2003). The definition of project planning that is used by Dvir et al. in essence captures the meaning of requirements engineering in software projects.

The world of agile software development on the other hand is a bit less studied in terms of project success factors than the more traditional project management space. A survey study by Chow and Cao (2008) suggests that agile software projects might have a different set of critical success factors. Based on data from 109 agile software projects, most strikingly, Chow and Cao did not find support for the need to have a strong executive commitment in order for an agile software project to be successful. This is unexpected since management support is so prominently present in other project success literature. Instead, their study proposes that the correct delivery strategy, proper practice of agile software engineering techniques, and a high-caliber team are the most critical success factors for an agile software project. Other success factors identified in the study are good agile project management, agile-friendly team environment, and strong customer involvement. (Chow & Cao, 2008)

On the risk side of software development projects, Keil et al. (1998) study the perceived risk factors and their relative importance in a panel study consisting of three panels of software project managers from different countries: Finland, Hong Kong, and the U.S. All panels independently identified the same 11 common risk factors, for which the relative importance ratings are presented in Table 2. Keil et al. note that most of the important risk factors are outside of the direct control of the project manager and only one of the risks involve technology. (Keil et al., 1998)

Table 2, Risk Factors in Software Development Projects Keil et al., 1998

Risk factor	**Relative importance**
Lack of top management commitment to the project	1
Failure to gain user commitment	2
Misunderstanding the requirements	3
Lack of adequate user involvement	4
Failure to manage end user expectations	5
Changing scope/objections	6
Lack of required knowledge/skills in the project personnel	7
Lack of frozen requirements	8
Introduction of new technology	9
Insufficient/inappropriate staffing	10
Conflict between user departments	11

Results from the panel study of Keil et al. are seemingly in line with the previously presented project success factors in the sense that the most important identified risks are often the inverse of the most important project success factors; e.g. support from senior management

versus lack of management commitment or good user involvement versus lack of adequate user involvement. The need for good software requirements is also prominently visible in the list of important software project risk factors. This indicates that when evaluating the aspects that factor into the success or the failure of a software project, focusing on the success factors alone can be enough since if the most critical success factors are not visible in the project, then the inverse of that success factor will count as a critical risk factor.

3 Theoretical Framework

This chapter describes the theoretical framework that is used to support the design of the case study and the formulation and analysis of the case study results. The framework is based on the literature that is reviewed in the previous chapter. The framework consists of factors that affect the choice of software implementation strategy when developing inter-organizational functionalities. The influencing factors can be divided into two main categories: external factors and internal factors. Next, both categories are described in detail along with justification for the incorporation of the chosen theories. In the end of this section, both categories are combined into a cohesive theoretical framework that is then extended with factors that are expected to affect the outcome of a software implementation project.

3.1 External Factors

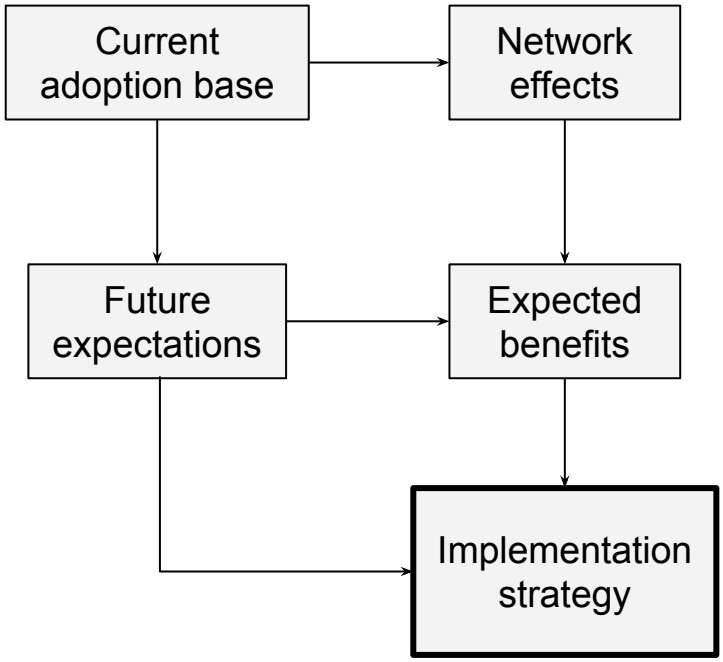
The main external factor that is expected to affect the choice of an implementation strategy in a software project with inter-organizational functionalities is the expected benefits that the company has for the functionality. According to the innovation diffusion theory, perceived benefits are an important aspect of new technology adoption (Rogers, 2010) so it is not farfetched to expect that the level of benefits a company is expecting to gain from a new functionality would be proportional to the level of effort they are willing to spend implementing the feature and this would in turn affect the choice of the implementation strategy.

Empirical evidence from an open-standard inter-organizational systems migration study by Zhu et al. strongly indicates that network effects affect the level of expected benefits and are also a major driver of migration to open-standards IOSs (Zhu et al., 2006). The same study by Zhu et al. also shows that the rate of peer adoption and trading community influence are key determinants for the strength of the network effects. The strength and size of the current adoption base for the inter-organizational technology in question can be an influencing factor in how the implementing company sees the future expectations for the use of that technology. Katz and Shapiro suggest that the level of current adoption of a technology acts as a signal for the future expectations for that technology (Katz & Shapiro, 1994). It is easy to see how this type of mechanism would then affect the software implementation strategy choice; if

there is reason to expect that a technology will be dominant in the future and there are major benefits through network effects associated to that technology, then it is safe to justify heavy investments into that technology rather than just doing a more superficial implementation.

Adding up these influencing mechanisms, a chain of determining external factors for software implementation strategy choice is formed (Figure 2). The factors that are presented in the figure below are not a definitive description of relationships between these different components; the level of future expectations is surely determined by other factors than just the level of current adoption base alone and network effect might affect the implementation strategy choice directly instead of indirectly through the expected benefits. Nevertheless, this framework is a starting points for evaluating the different external aspects that will shape the software implementation strategies that companies choose and apply when creating software products and services that include inter-organizational functionalities.

Figure 2, External Factors of Software Implementation Strategy Choice



3.2 Internal Factors

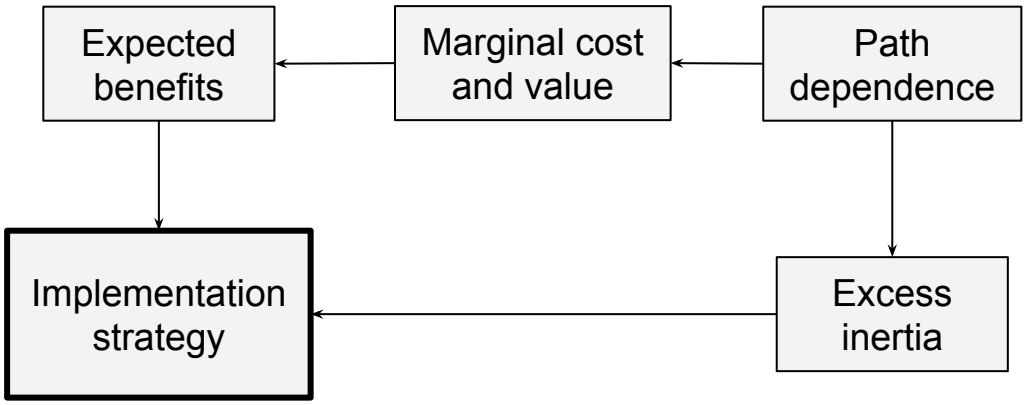
The internal factors of the theoretical framework, presented in Figure 3, start with the same expected benefits factor as does the external factors framework. In addition to network effects and future expectations, the level of benefits a company is expecting to receive from a software project, whether it is inter-organizational or intra-organizational, is shaped by the marginal cost and marginal value of adding the technology to the company's existing

offering; i.e. how much does the project cost and how much does the company expect to receive value out of it. An investment of any type will typically adhere to some form of basic net present value analysis that states that the combined present value of all the future cash flows from the project should be positive while taking into account the initial investments and time value of money (Remer & Nieto, 1995a; Remer & Nieto, 1995b).

The marginal cost and marginal value of a software project is dependent on the technical path dynamics of the implementing company as described by the path dependence theory (section 2.3). When a company has experience on similar technologies from earlier projects, it tends to better estimate the cost and value of similar future projects. Also if a company has postponed investing in new technology, the marginal cost of a new technology implementation project will most likely be high because of shortcomings in the existing technical infrastructures. On the other hand, if a company has been previously active in developing new technologies, the marginal value from a new technology project may be comparatively small since the company has less to gain from adding new technologies. (Zhu et al., 2006)

Based on the work of Farrell and Saloner, excess inertia is added into the framework of internal factors. In their article they conclude that the existing path of standards adoption will create excess inertia that affects the present-day choices of a company (Farrell & Saloner, 1985). The converse of excess inertia in turn is called “excess momentum” that can lead to a premature technology adoption (Farrell & Saloner, 1986). Excess inertia or excess momentum are determining factors when a firm decides whether to invest in or abstain from a new technology. When the firm then makes the investment decision, the level or eagerness involved in the decision will also affect the level of commitment the company has for the new technology and through this mechanism it will also affect the choice of the implementation strategy and how much the company is willing to invest in the implementation.

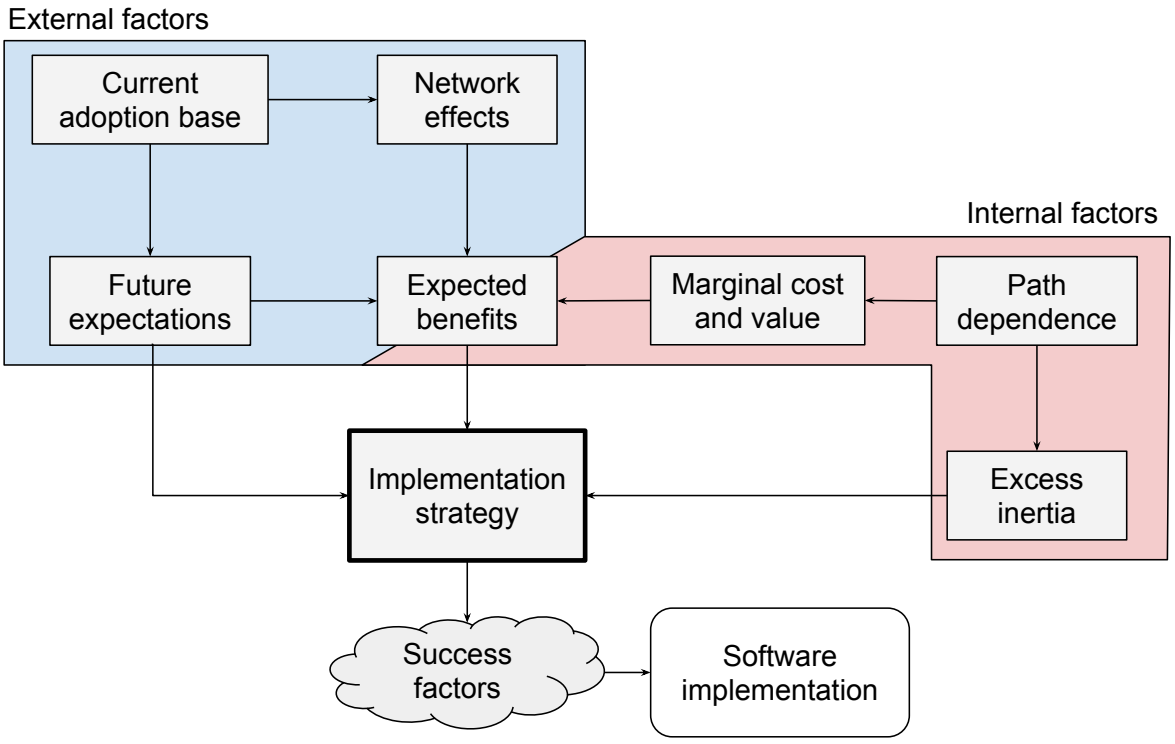
Figure 3, Internal Factors of Software Implementation Strategy Choice



3.3 Combined Theoretical Framework

When the internal and external factors affecting software implementation strategy selection in inter-organizational context are combined, a combined theoretical framework can be developed. The result of combining the internal and the external factors into one framework is presented in Figure 4. Companies are expected to make implementation strategy decision based on their future expectations and expected benefits for the particular technology as well as based on the excess inertia that originates from its previous technology and standards related choices. Network effects are an important driver for the expected benefits and the strength of the network effects in turn are shaped by the inter-organizational nature of the technology in question and the size of the community that is using the technology. Expected benefits are also shaped through the marginal cost and value of the implementation project that are in turn determined by the technological and organizational decision path that the company has taken.

Figure 4, Combined Theoretical Framework



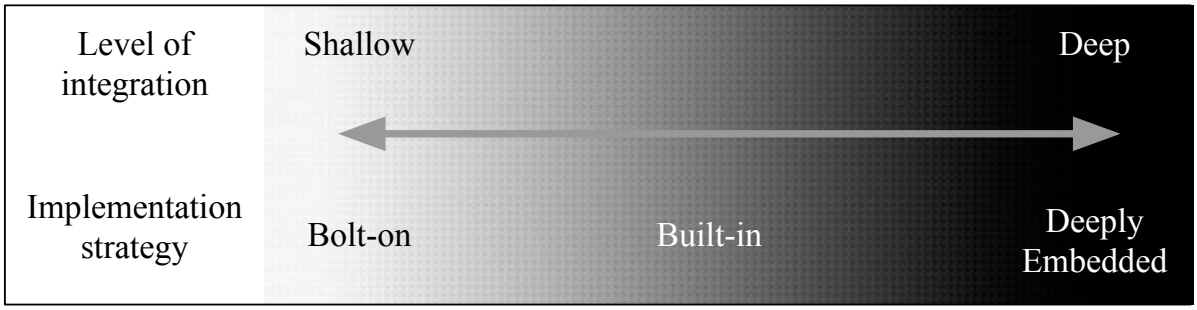
Next, focusing on the software implementation strategies themselves; although generic software implementation strategies in inter-organizational projects are not widely researched, some idea on the possible types of strategies can be obtained from XBRL specific implementation literature. Even though ERP system implementations are widely published, such examples might not be generalizable enough because such projects usually focus more on business process engineering than software implementation itself. Table 3 illustrates the different XBRL implementation strategies described by Garbellotto (Garbellotto, 2009a; Garbellotto, 2009b; Garbellotto, 2009c). These categories – bolt-on, built-in, and deeply embedded – match with the adoption level definitions that are used by Garner, Henderson, Sheetz, & Trinkle (2013) in a survey study to describe XBRL adoption levels: non-adoption, low adoption, medium adoption, and high adoption.

Table 3, XBRL Implementation Strategies

Implementation strategy (Garbellotto)	Adoption level (Garner et al.)	Initial cost	Long term cost	Use of external resources	Benefits	Limitations
Bolt-on	Low adoption	Low	High	High	Immediate results	High maintenance, negative ROI in the long run
Built-in	Medium adoption	Medium	Medium	Medium	Maintainability, positive ROI	Limited capabilities
Deeply Embedded	High adoption	Medium	Low	Low	High added value	Complex and time consuming transition

Other reviewed literature on software implementation focuses on the iterative and incremental software implementation strategies (Fichman & Moses, 1999; Petersen & Wohlin, 2010). Combining these views on software implementation strategies, a spectrum of implementation strategies can be formulated as shown in Figure 5. On the "shallow" end of the spectrum there is the bolt-on category that is the most superficial implementation, where the functionality is not integrated inside existing systems but it is rather built outside of them. On the "deep" end of the spectrum there is the deeply embedded category that means a very high level of integration with existing systems, possibly even taking the new implementation as part of the technical foundation of other functionality.

Figure 5, Levels of Integration and Software Implementation Strategies



It should be noted that an implementation strategy can be a mix between different categories and the end result does not necessarily have to be realized in one go. A company that might start out with a bolt-on solution, can iterate over longer periods of time, adding more integrated functionality as it goes and eventually move to a built-in solution. Also, a company that starts out with a decidedly deep integration strategy in mind, might at first start from just one part of their existing system and move from there on to other parts gradually, instead of

doing the deep integration with one big architecture refactoring. Iterative practices are starting to become the norm as more and more companies adopt agile software development practices, which generally means moving from heavy waterfall projects to working with smaller and more easily manageable units of work.

Lastly, the final piece of the theoretical framework is the success factors in software development projects. These determinants of the outcome of software projects are necessary to account for in the theoretical framework since they are important when forming a comprehensive understanding of software implementations. If, for instance, all the right success factors are in place in a project but the project team still encounters significant obstacles in their implementation, it gives grounds to investigate further what unexpected factors are causing the issues. In inter-organizational context, there might be multiple factors that are outside of the direct control of the implementation project that are not accounted for in the success factor literature. Identifying these factors can lead into important new realizations about software implementation strategies and how they are suitable for different types of projects.

Based on the literature reviewed in the previous chapter, a combined list of five success factors in software implementation projects is formulated. This list is illustrated in Table 4. The listing is certainly not complete and the items in it are any in no particular order. It is created with a focus towards agile software projects and it emphasizes the importance of requirements planning, as suggested in multiple studies (Berntsson-Svensson & Aurum, 2006; Hofmann & Lehner, 2001; Reel, 1999). Even though some views argue that having management support is not a necessity in agile software projects (Chow & Cao, 2008), it is still such prominent factor in tens of project success factor publications (Fortune & White, 2006) that it is not easy to pass in the listing.

Table 4, Success Factors in Agile Software Implementation Projects

Success factors in agile software implementation projects
Strong management support for the project
Good requirements planning
Sufficient communication between stakeholders, including the users
Adequate software development practices
Skilled team members

4 Methodology

This chapter describes the selection of research methodology that is used for the study and discusses the design of the study, the study protocol, data collection methods, and the validity and reliability of the study.

4.1 Study Method Selection and Definition

The aim of this thesis study is to identify, describe and compare software implementation strategies that companies use when developing inter-organizational functionalities and to identify the factors that affect the choice of a software implementation strategy. Combining this goal with the case study definition of Yin (2003), the research objective can be written in such way that it invites the use of case study as the study method: The aim of this research is to empirically investigate a contemporary phenomenon (software implementation strategies) within its real-life context (inter-organizational project inside companies). Moreover, due to the nature of the subject of the study (behavior of organizations), the boundaries of the phenomenon under investigation are not clearly evident before the study is conducted and there is no possibility to conduct controlled experiments on the topic.

The case study research method has no single uniformly accepted definition. A concise definition of the case study method that is combined from multiple sources is formulated by Benbasat, Goldstein, & Mead (1987) as follows: “A case study examines a phenomenon in its natural setting, employing multiple methods of data collection to gather information from one or a few entities (people, groups, or organizations). The boundaries of the phenomenon are not clearly evident at the outset of the research and no experimental control or manipulations are used.”

Three main categories of case study research can be identified: exploratory, descriptive, and explanatory. Exploratory study focuses on initial research and tries to identify patterns in the collected data without having any predefined model that would define the analysis. Descriptive case study moves deeper from the exploratory study and tries to obtain information on more well features of an issue. Both exploratory and descriptive case studies focus on “what” questions whereas the explanatory case study deals with “how” and “why” questions, analyzing the reasons behind a phenomenon instead of just observing it. (Yin, 2011)

The research question in this study is distinctly a "what": What types of software implementation strategies are used when undertaking software projects with inter-organizational functionalities? Various aspects surrounding the phenomenon under investigation have been previously studied as illustrated by the theoretical framework presented in the previous chapter. Also, the research question focuses more on specific matters instead of generalities, such as, what factors affect the selection of a particular software implementation strategy. These aspects of this case study advocate that the study method in question is best characterized as a descriptive case study.

4.2 Study Design

According to Yin (2003), a case study research design is "the logic that links the data to be collected (and the conclusions to be drawn) to the initial questions of study". Continuing with Yin's definition, a case study design should include the following five distinct components:

1. Study questions
2. Study propositions
3. Unit(s) of analysis
4. The logic linking the data to the propositions
5. Criteria for interpreting the findings

The study question is, as described in earlier chapters, to find out how companies approach implementing new inter-organizational functionalities into their software products and services, and why do they choose the particular implementation strategy. To emphasize the selection of case study as the study method, the study question is written here knowingly in the typical case study "how" and "why" format. The key propositions for the study are that network effects will play a role in the implementation strategy selection along with the technical and organizational path of the company. Also, it is proposed that software implementations should exhibit a categorization between shallow and deep in terms of the level of integration.

Unit of analysis for the study is logically a software implementation process, starting from the initial decision to implement and ending with the production release of the functionality. Alternatively, it could be argued that a sufficient unit of analysis could be just the initial implementation and design phase of a software project. However, by incorporating the whole implementation process into the unit of analysis, a broader understanding of the different software implementation strategies can be formulated and the categorization of the found

implementation strategies does not have to rely only on the descriptions of the implementation strategies given by the companies themselves, but rather it can be built on the base of empirically observed project outcomes.

The logic that links the data to the propositions is a form of pattern matching; if the propositions are to be correct, companies are expected to describe signs of network externalities when describing their implementation strategy decisions, to relate their decisions to their existing technology, and to categorize their implementation strategies in terms of the level of integration. However, no strict criteria can be asserted on what type of findings will be interpreted as positive confirmation of the study propositions, but rather the results have to be interpreted within their context.

Since the unit of study is the software implementation process and one of the propositions is that the implementation strategy selection is affected by path dependence, it is logical to select multiple-case study as the research method. The focus is on comparing different types of implementation processes and it is expected that companies will end up selecting different implementation strategies if the external and internal circumstances that are related to the implementation project decision are different. The aim is to find companies that are implementing similar projects at the same time, so that the external factors of the projects could be somewhat similar. If such projects can be selected for the cases, it is expected that the drivers behind the companies' implementation decisions should be the same in term of external factors, meaning that the cases follow a literal replication as defined by Yin (2003). The internal aspects of the implementation strategy selection are of course different of each company, which means also that the results can provide a theoretical replication (Yin, 2003).

4.3 Case Selection

Reporting of financial statements in XBRL form is available in Finland starting from April 2016. The first wave of accounting information system (AIS) companies that are implementing the new Finnish XBRL/SBR reporting capabilities into their software products offers a unique set of inter-organizational software implementation cases to study. These implementation cases will offer a partly literal replication of the results because the external factors of the projects are mostly same for every company. Each company uses the same underlying standard, the XBRL/SBR taxonomy, so they have the same level of knowledge about he current state of XBRL reporting Finland and they are participating in the same pilot program. Some aspects of the study will also provide a theoretical replication in the sense

that the selected companies will most likely have different implementation strategies due to differences in their internal factors that are expected to affect the strategy selection process, such as their individual unique technical foundations and different internal organizational dynamics.

It can be argued that a richer set of cases could be obtained by observing companies doing different types of software implementation projects in different business domains, rather than focusing on a single event in which multiple companies undertake a projects with almost identical goals at the same time. To counter this argument, such variety in the selected cases could possibly lead into difficulties in identifying all the countless factors that affect such a complex organizational process. With a certain level of homogeneity between the selected cases, the differences between the companies and their implementation strategies are more likely to be more pronounced. With the selected approach, all the companies will have products that are fairly similar, the business domains will be the same, the network effects are likely as strong, or as weak, for all companies at the time of the study, and every company has roughly the same goal for their projects.

To summarize the case study design, case selection, and the focus of the study: The research method of this study is descriptive multiple-case study with a comparative focus. Case study subjects are AIS software companies that are finalizing their first Finnish XBRL/SBR pilot implementations and the focus of the study is to establish a descriptive storyline for each XBRL implementation project, identify the implementation strategy in each project, find differences between the observed strategies, and investigate the reasons behind the implementation strategy decisions.

4.4 Data Collection

After the initial case selection, a group of seven companies was identified to be fitting targets for case interviews. These companies formed the first wave of adopters that had started to implement XBRL/SBR functionality into their products in late 2015 and were aiming to be ready to test their implementations when the Finnish Tax Administration would open up for receiving reports in April 2016. The companies were identified with help from TIEKE, the party that organizes and coordinates activity around the XBRL/SBR taxonomy, to ensure that each candidate case would fit with the purpose of the study. A protocol for the interviews was developed to ensure that each case would be approached and handled in the same way and a set of semi-structured interview questions were developed based on reviewed literature.

4.4.1 Data Sources

The interviews form the main body of data for each case. The interviews were recorded and transcribed for later analysis and notes were kept during the interviews. Yin (2003) suggests strongly that case studies should use as many sources of evidence as possible in order to achieve data triangulation that can help in supporting the results of the study. However, finding other data sources, in addition to interviews and field notes, for studying the software implementation strategy decisions in XBRL/SRB projects is problematic since all the relevant data is contained mostly inside the companies and it is not publically available. In addition to the direct interviews, supporting information, such as basic information about the companies and their products were researched online before the interview so that the interviews would not have to deal with basic facts about the companies. Also, details that came up in the interviews were fact-checked afterwards by using online resources whenever possible. In order to create a foundation of knowledge about general XBRL/SRB topics, multiple discussion sessions with two key representatives from TIEKE and XBRL Finland were conducted. In addition, background information on the XBRL/SBR reporting project was collected from multiple online resources: Finnish Tax Administration, 2015; Rintala, 2015a; Rintala, 2015b; XBRL Finland, 2015.

4.4.2 Case Study Protocol

The communication with each of the candidate companies started by approaching contacts that were provided by TIEKE with an email for a request for an interview with one or more people who are familiar with the XBRL/SBR project and its origins, preferably a senior manager, a project manager, or a product owner. Each email was sent with the same wording that described the purpose of the study, how long the interview was expected to take (1.5 hours), what kind of person would be suitable for the interview, and where the contact information for the company had been received. Interview times were scheduled for December 2015 and face-to-face meetings were preferred over telephone interviews, although two of the five interviews had to be performed over telephone due to geographical reasons.

Altogether five individuals from four companies agreed to be interviewed. The interviews were audio recorded and notes were taken during the meetings. The audio recordings were later transcribed for further analysis. Each interview started with an introduction script that described the purpose of the study, how the interview material would be handled, how the

semi-structured interview was organized, and that the interviewee could offer their own comments and remarks at any time they felt that something relevant should be added or some topic outside of the interview frame should be discussed.

4.4.3 Interview Question Design

The design of the interview questions was guided by the theoretical framework that is described in chapter 3 and discussions with XBRL experts from TIEKE and XBRL Finland. All the interviews followed the same semi-structured format with questions that were designed to capture the timeline of the XBRL implementation project, to create an understanding of the software implementation strategy that was used in the project, to highlight key decision-making processes, and to focus on the major difficulties and successes during the implementation. The questions emphasize the reasons behind the decisions that were made during the project and explore what factors were considered when decisions were being made and how well the typical software project success factors were present in the projects.

A part of the interview focused also on the use of external help or experts in the project and evaluation on how well XBRL, XML, and other key technologies were known and understood in the organization before the project started. Each interviewed person was also asked to give their own post-project evaluation of the implementation and to suggest things that they would do differently if they would start the same project over again. A timeline was established that follows the different stages of the project: initial decision-making, requirement definition, software implementation, validation, and production.

The interview questions are presented in Appendix A. In the actual interview situations, the initially defined questions were purposely followed rather loosely, leaving room for open discussion but still making sure that each of the topics were discussed during the interview.

4.4.4 Research Ethics

Before the case studies, ethical considerations were made to ensure that the interviews stay confidential and that every interviewee knows in advance what they are agreeing to when they give the permission for the interview. The email that was used to approach the case subjects was crafted so that it explicitly tells about the aims of the study, who is conducting the study, where it will be published, and what information will be discussed in the interview. Permissions for interviews were all acquired in writing.

Each interview started with the same introduction that reiterated the same topics mentioned in the email and also stated that the interviews will be handled confidentially; the audio recordings from the interviews are transcribed but no direct excerpts will be published. Also, the interviewees were told that their names and the names of their companies would not be published.

4.5 Validity and reliability

There are four commonly used tests for the quality of empirical research, three of which are relevant for this study. Yin (2003) defines these tests as shown for the applicable parts in Table 5 and offers tactics to ensure that the validity and reliability requirements for a case study are met.

Table 5, Case Study Tactics for Applicable Design Tests

Test	Case study tactic
Construct validity	• Multiple sources of evidence • Establishing a chain of evidence
External validity	• Using replication logic in multiple-case studies
Reliability	• Using case study protocol • Developing a case study database

To summarize the different types of tests, construct validity means establishing correct operational measures for the concepts that are being studies. External validity is the process of establishing the domain in which the study's results can be generalized. Reliability is the means that ensure that the results of a study can be replicated. (Yin, 2003)

As previously stated in section 4.4.1, finding multiple sources of evidence that would enable proper triangulation is challenging due to the fact that the information about the companies' internal processes that are being studied are only available inside the company; some aspects of this information might even be considered as trade secrets. In order to pass the construct validity tests, a chain of evidence must be established within the bounds of the ethical considerations that are presented in section 4.4.4. Direct transcriptions of the interviews cannot be published in verbatim but descriptions of the cases have to be presented with sufficient level of detail. Providing the list of interviews questions in addition to describing

the case study protocol accurately enough will provide a chain of evidence that can be followed without violating the confidentiality of the interviews.

For external validity, replication logic is established as described in section 4.3. Both literal and theoretical replication logic is used, meaning that the cases are likely to corroborate each other's in terms of the effects of the external factors and also cover different theoretical conditions in terms of the internal factors. With this kind of replication, the effects of the internal factors can be better compared between the cases because the external factors are almost identical for every company.

Lastly, the reliability of the study is addressed by establishing a case study protocol that is followed accurately for each case, as described in section 4.4.2. The communication with the case subjects and the data collection procedures were deliberately kept as uniform as possible between the different cases. Also, a case study database has been developed containing the interview audio recordings, transcriptions, field notes, and downloaded online materials. Although, again due to the same confidentiality reasons already mentioned, most of the material in the case study database cannot be published as such.

5 Results

In this chapter, the results from the case studies will be described along with cross-case comparison and analysis.

5.1 Case Reports

Based on recommendations from TIEKE, altogether six companies out of seven XBRL pilot participants were approached with a request for interviews. Each of the companies that were approached was an active participant in the XBRL/SBR pilot and the companies are expected to be able to produce electronic financial statement reports in April 2016 when the Tax Administration starts accepting them in the new format. All of the companies in question develop some form of accounting information system software in-house and offer it to their customers either as a cloud service, in a customized software package, or by selling a service that utilizes the software in its delivery.

The interview requests were sent out in September 2015, leaving enough time to schedule the interviews for December 2015. Four out of the six companies that were approached agreed for interviews. Out of these companies, two offered one person to be interviewed and the other two offered two persons. From these interviews, one was later cancelled, leaving a total of five people to be interviewed from four different software implementation projects. Each interview was scheduled to take at maximum one and a half hours.

Given that these four projects were all active concurrently and all of them were aiming to implement the same type of inter-organizational software functionality, the level of access is impressive. The companies in the cases are labeled with letters from A to D. For reference, general characteristics of each of the companies are listed in the chart below.

Table 6, Summary of Interviewed Companies

Company	Characteristics	Product
Company A	Accounting software vendor, part of an international concern	Cloud AIS, sold directly to customers
Company B	Large national accounting firm	Internally used AIS
Company C	Big global technology and consulting corporation	Customizable reporting tool, packaged software
Company D	Small startup, cloud software vendor	Cloud AIS with focus on digitalization, sold directly to customers

5.1.1 Case 1, Company A

Company A develops and sells financial administration software as a cloud service. The interviewee is the CTO of the company who is in charge of software development, R&D, and keeping up with the technology trends in the field of financial administration. The company's customers include both accounting firms that offer their services to other companies and direct end-users. After over 15 years in the market, the user base of the product is around 450 accounting firms and more than 12,000 end-user companies, which makes Company A a large player in the Finnish financial administration software scene. The company handles their R&D, software development, and software operations in-house, employing around 40 people in development and devops (software operations) tasks.

The main product of Company A, the cloud based financial administration application, is implemented mainly in Java and it uses third party open-source and commercially licensed components where applicable. The software is mature and feature rich and it has a lot of integrations to external systems, such as online banking, printing and scanning services, and e-invoicing. Integrating to new external systems and implementing new data interchange standards is familiar for the company.

Starting from already five years ago, the CTO had been in talks with the tax administration and had identified that the current way of operating, where the tax official defines its own forms and companies then implement them, is not the most efficient way moving forward. The data that is handled in the accounting systems and by the tax administration is the same but the way in which it is represented is different. When the standard business reporting code set and XBRL started to emerge as a way to transfer this data, the CTO was active to follow the topic. At first it looked like that the standard business reporting code set might even become the default chart of accounts in Finland but now the CTO doesn't believe that this will be the case since the accounting act was renewed in 2016 and there is no mention about the chart of accounts.

The CTO identifies XBRL/SBR to be an important development in the field of electronic financial administration and says that their company will definitely support it when the market is ready for it. The company decided already in early 2014 to do a proof-of-concept implementation of XBRL/SBR in order to prove that they are ready to support the new standard when the time is right. At that time, they didn't have any particular business driver for the implementation decision, rather they wanted to contribute to their field and also do the

"research part in R&D". They felt that doing an actual implementation of the new standard would be the best way to keep up with the latest trends and to really understand what kind of benefits XBRL would bring to their product. The company chose to do the proof-of-concept as an external component that takes in data in the standard business reporting code set format and translates it into XBRL reports. The component wasn't developed as an integral part of their product at this stage of development, rather they want to wait and see how XBRL/SBR is taking off before they make the final decision on how they will approach the technology. The initial push to implement XBRL/SBR came when TIEKE, the facilitating organization in XBRL/SRB, and asked whether the company would be interested to participate in the consortium of companies around the standard. A decision to do a proof-of-concept was made when the CEO of the company, briefed by the CTO, agreed to invest the necessary funds to hire a summer trainee to explore the topic.

The whole proof-of-concept implementation was done by one summer trainee software developer over the time period of two months. The work was organized along the lines of a Scrum process but instead of a team, there was only one person doing the work. The CTO acted as a product owner for the project, provided the initial top-level requirements and periodically every two weeks checked in to see how the project was progressing. The software developer trainee had previous experience in working in an accounting office so there was little need for detailed specifications and orientation to the domain specific knowledge in the beginning of the project. The software developer also had access to TIEKE in XBRL related issues during the implementation process. Before the project started, XBRL was not known particularly well for anyone inside the company. The developer had not even heard about the technology before the project, however XLM was well known and understood by everyone involved.

In addition to the one software developer, the team can be said to be consisted of two external contributors from two partnering accounting firms. These partners provided the project with test data form real companies that was pre-mapped into standard business reporting code set accounts. The implementation itself didn't perform the mapping but rather took in data that was already in the correct format. The project was finalized with a brief validation phase where XBLR file instances were created from the data of real limited companies and these XBLR documents were validated by TIEKE with an XBLR validator application that also contained logic for validating the SBR taxonomy. After a few iterations of the validation process, the project was concluded to be completed.

Regarding the success factors in the implementation, the CTO notes that already early on in the project, it was realized that they would have to get a developer that has both the technical knowledge in software development and the right type of domain knowledge in accounting. Otherwise they would have to spend a significant amount of time defining specifications and the complex XBRL format would most likely seem intimidating for someone who is not familiar with accounting concepts. During the implementation the main difficulties that were faced were related to the complexity of the XBRL format itself. The CTO had talked about XBRL for years before the project but said that only when the project started, they actually really looked at the technical specifications in detailed. They found out that due to its highly expressive nature, XBRL can be complex and difficult to understand at first. The CTO says that had the requirements planning phase been more thorough, they would have been able to address some of the issues that rose from the complexity of XBRL format earlier in the project. However, these issues were related to mostly detail level matters and had little to do with the higher-level design of the project. The CTO and the developer had to do some learning and designing as they went forward with the implementation but the delays didn't end up being major.

According to the CTO, one of the key challenges facing the XBRL/SBR adoption is the diversity in the charts of accounts that are being used in Finland. If companies are not using the standard business reporting code set when creating their accounting data, there might be situations where the XBRL report would need more information, or more resolution power, from the source material than there is available. In this kind of situation, a simple mapping from the used chart of accounts to standard business reporting code set is not possible. If the chart of accounts has enough information however, a competent accountant can do the mapping between it and the standard business reporting code set in just a few hours.

Company A's view is that the issue of mapping accounts to XBRL/SBR tags should be solved with regulation; either the regulators should enforce the use of XBRL in financial reporting or they should make sure that every acceptable chart of accounts has enough detail so that it would be translatable into the standard business reporting code set. An option for this type of new regulation would be that XBRL adoption would happen within the accounting industry in a market-driven way. The CTO however doesn't believe that this will happen since it would require that a company that is using XBLR would gain a significant competitive advantage over companies that are not using it. The CTO says that there has been talks that financiers would start requiring XBLR data in order to be able to analyze

companies more efficiently, but the proportion of companies that are seeking the type of financing where this could be a competitive advantage is so small that it will not change the whole industry. Another party that would have something to gain from XBLR reporting are banks, but the CTO does not see it feasible that any bank alone will start requiring XBRL reporting in order to grant a loan since it would be intentionally limiting its business.

The CTO's evaluation of the implementation project as a whole is that it was a relatively simple task altogether, considering that it was completed within a few months by a software developer summer trainee with just a fairly modest amount of support. It required that the developer was well suited for the project in terms of domain knowledge but in addition to that, the project was not difficult. However, if the company would do the same proof-of-concept project again, they would not take the same kind of approach of pre-mapping the input data, but rather they would include the mapping between the different types of charts of accounts and the standard business reporting code set into the XBRL component itself. They would also have to build a tool that supports this mapping process. This way they could take real-world data as it exists in their production environment and use it to generate XBLR/SBR reports. Currently the proof-of-concept, as it was implemented, is not production capable.

Now that the proof-of-concept project is done, the company has no immediate plans on doing more research and development around XBRL/SBR reporting. They say that the first phases of XBRL reporting do not bring any added value or something that they don't already have for their users. Instead, they will wait until the standard is widely used enough that they have a business reason for adopting it and they can provide real customer benefits by doing so. The CTO hopes that the regulators and the public administration will push the use of XBRL/SBR forward in meaningful ways so that companies will have real reasons for starting to provide XBRL reports.

5.1.2 Case 2, Company B

Company B is a large accounting firm that operates in multiple cities around Finland. The interviewees are a systems development specialist from Company B and a student from a local college with which the company organized their XBRL implementation project. The company has approximately 2000 customers ranging from small businesses to publicly listed companies and it employs around 200 people. The company does not sell a software product directly to its customers, but instead it operates with a service model. They sell a wide range

of accounting related services and have an internally developed information system that the company uses to run these services.

The software that Company B uses and develops is a cloud service that can also be accessed by the company's service subscribers. The company employs an IT-team in-house that develops and maintains the software. The product has integrations to multiple external systems, for example a payroll system, debt collection, and online banking. The system also offers an access for the use of external auditors. Most of the software product has been developed in-house from scratch excluding a few commercial software components from third party vendors.

Company B has a cooperation contract with a local college that means that the college will offer "real world" software projects for the company and have their students implement them as a part of their software development project course. A teacher from the college approached the company and suggested that a group of students could do a black box implementation of XBRL/SBR report generating as part of this course. The company hadn't been actively following the development around XBRL/SBR taxonomy and the specialist wasn't familiar with the technology before the XBRL project actually started, but they still recognized the potential usefulness of such pilot project. After starting the project, the company joined the consortium that organizes activities around the new XBRL/SBR taxonomy.

The decision to start the XBRL implementation project was made in the summer of 2015 and the project started in the fall of the same year. The main motivation for the project was for the company to get familiar with the new standard and to obtain basic XBRL reporting capabilities that could be taken into use when the time is right for it. Since the project didn't take any of the company's IT department's time, it was easy to justify. The company also participates in a national initiative that aims to develop new digital financial administration services and to automate processes in financial administration. Both of these goals are such that the XBRL implementation project will also support them.

The initial idea to implement the XBRL reporting functionality as an external component came from the students and the company mainly just wanted to have the functionality done with as little effort as possible. Since the whole implementation work for the project would be done externally, a black box implementation was the preferred way to move forward because it has a clear-cut interface for data entry and it does not have any need to access the internals of the company's information systems. The XBRL reporting component takes in

data that has been pre-formatted to the standard business reporting code and produces an XBLR file instance. The company identified that using the standard business reporting code set as an intermediary will offer a data representation format that is fairly stable and that can harmonize the various different internally used charts of accounts. However, a mapping tool was not implemented as a part of the project due to schedule constraints, but instead the company did a conversion from internal data to standard business reporting code set manually with Excel.

The student group consisted of seven students from the same software development field of study. The students worked by using Scrum as their software development process, diving their work into two and three week sprints. One of the students was appointed as the project manager, three of the students focused on the actual software development, and three were responsible of the software specifications and documentation. One student in particular had previous experience in working as an accountant and thus operated mostly in an advisory role, bringing valuable domain knowledge into the implementation team. The progress of the project was reported to the product owners at Company B in sprint review meetings after each sprint. The whole project was completed within the timeframe of two teaching periods or roughly four months.

When the project started, the only goal for the company during the requirements planning phase was to take the internal data of the company's software and to produce a valid XBLR/SBR document for certain two specific report forms that could be then sent to the tax administration's systems. In this sense, the project's scope was rather narrow; the aim was to implement the conversion from a predefined standard business reporting code set format to XBRL/SBR and only support two types of reports with no additional mapping tools or complementary reports being included in the project scope. When the actual implementation project started, it did not have any formal planning phase where the company's representatives and the students would have worked together and defined the scope of the project, but rather the company and teachers from the college agreed on the project's scope and it was then handed over to the students.

In addition to receiving some guidance on the project scope from the company, TIEKE and representatives from the tax administration were involved with the students and helped them to understand the technical specifications of creating the XBRL reports. TIEKE arranged training on XBRL technology and the tax administration helped the students by validating their XBRL documents during the development and giving feedback on possible technical

errors. When the students first started to get familiar with the XBRL/SBR taxonomy, they felt that it was a big and challenging topic to understand and internalize. The student group did not have anyone who was familiar with the technology before the project, although general aspects of XML were familiar to the group. The students felt that having access to XBRL experts at TIEKE helped them significantly with getting up to speed with the project.

Event with the additional support from TIEKE and tax administration, the students felt starting right from the beginning of the project that they were missing guidance and input from the company. They had issues working with the input files that they had been given since the format changed throughout the project. The students and the company also felt that there were miscommunications between the two parties. The company also proposed that the students would expand their project scope midway through the project by including additional reports and functionality but the students felt that they cannot commit to anything outside of the initial project scope due to their tight schedule. The company also did not have time to attend to all of the review meetings which meant that the communication between the implementing student group and representatives from the company was sparse. The students felt that even though they would have otherwise had a good chance to complete the project within schedule and with the initially given scope, the lack of communication and commitment from the company's side made the project difficult.

At the beginning of the project, there was a brief two-day internal definition and planning phase after which the students started the actual implementation of the project that took most of the time in the project. The company was kept up to date on the progress with demo sessions between each sprint. Despite the lack of feedback, the students felt that they could accomplish most of what they had set out to do and felt that having the accounting domain knowledge inside the team helped them with the implementation project significantly. The students had to conclude their work at the end of their software development course and they had to leave out some minor work that they had initially planned to do in the beginning of the project. The final week of the project was dedicated for validation and making sure that every feature that had been implemented worked as expected. During the validation, the team used a third party XBRL validator tool, Arelle, to verify that their software produced a valid XBRL document since the tax administration did not provide any convenient means to validate the output against their own systems that would eventually be used receive the XBRL reports.

The finished XBRL/SBR report generating component is a Java application that can read CSV (comma separated values) files and produce two tax report forms as XBRL files. The application also has a graphical UI (user interface) component that can be used to input non-numeric data that is required for the reports, such as the reporting company's contact information. The application can also be run from the command line without the UI so that it can be programmatically triggered or scheduled to generate the reports.

After the project was finished, the students evaluate that the project was technically not too challenging. Although getting familiar with XBLR required a lot of work, the program logic itself was not complicated. Most of the challenges that the implementation team faced were due to miscommunication, requests to change the project scope while the project had already been started, and changes in the input format. Otherwise the students felt that they would not have done anything differently in the project. Having an accounting expert in the team and getting help from XBRL experts and the tax administration helped the team with their work significantly.

The company itself was satisfied with the project's output and they felt that they had gotten what they wanted from the project. The company intends to implement a broader set of XBRL functionality into their product later; they feel that having a working application that can produce XBRL/SBR reports will help them down the road so that they won't have to start from nothing after a year or two when they decide to implement XBLR properly. At this time, the company does not have any concrete plans on the future XBRL support in their software.

5.1.3 Case 3, Company C

Company C is a large multi-national corporation that, among many other things, develops software products, sells and maintains IT infrastructure, and offers consulting services. The interviewee is a software specialist at the company's sales and services organization's Finnish branch, in a team that sells, customizes and provides consulting services for a configurable software package that is aimed for producing financial reports and automating processes around these reports. In addition to creating financial reports, the product also offers a range of teamwork facilities that provide a secure collaboration environment for creating the documents. The company offers the product either as an installation on their customers' premises or deployed through the company's cloud service infrastructure.

The product is aimed for mid-sized and large corporations and it has a focus on supporting the complex reporting requirements of internationally operating publically listed companies. XBRL support that enables the use of multiple different XBRL taxonomies is built in to the product so that, for example, companies that operate in multiple jurisdictions can use the same report templates when generating reports for their various local administrations. The product can pull in data from various different input sources, such as Excel files, PDFs, and PowerPoint presentations, and create reports in various different output formats, XBRL being one of them. Other supported output formats are of the more human readable kind, such as Adobe InDesign and Microsoft Office formats.

The company develops their product in Canada and the Finnish team is mainly responsible for its local sales and support, although the initial deployment of the software package will require a significant amount of customization before it can create a report. Every new installation of the product is essentially a blank slate that is then crafted to fit the needs of each individual customer. In Finland and in the Nordic countries, the product is still relatively new and there are only a handful of customers using it. From the Finnish user base, no one has yet started using the product's XBRL reporting features.

The company joined the Finnish XBRL/SBR consortium when it was first started. As a major software vendor that caters for corporate clients it has an interest in following up with current developments in financial reporting in Finland. TIEKE approached the company in 2014 and asked if they would be willing to do a pilot in using the new XBRL/SBR taxonomy with their product. The company saw this as a possibility to introduce a new client to their product and also at the same time test their product with the new taxonomy and to prove that they can support it when it is taken into use by the tax administration and the patent and registration office. The main reason for supporting the new taxonomy is that the company expects it to be a prerequisite for selling their software when the Finnish officials start receiving financial reports in XBRL format. The specialist says that no one will buy their product because of its XBRL support, but it is rather something that just has to be there in order for them to be able to sell the product. They predict that they can use the XBRL functionality to possibly highlight other benefits that the product has, but as a standalone feature, it is something that every customer will most likely just expect to be there.

The product itself has been built in a way that different XBRL taxonomies can be imported into it as machine readable definitions. Since the input data that the software can be given to create reports is such diverse and not in any particular pre-determined format, there is always

manual work involved in mapping the various inputs to XBRL tags that can be then used to create XBRL reports. Because of this, the product has a built-in tagging functionality that is specifically designed to support binding inputs to XBRL elements. Despite the product can in theory import any type of XBRL taxonomies, the specialist says that there is always some degree of validation and testing required before they can be completely sure that a new taxonomy is handled as it is intended by product.

When Company C started to look for a suitable customer for their pilot, they were surprised how easily they managed to get a company to partner with them. The client company had not previously used the product but was familiar with it and they were interested to get to know XBRL more closely and to get to hear what is happening in the world of electronic financial reporting. This way Company C managed to get the client company even more familiar with their product and also do valuable testing at the same time, making the pilot a good value proposition for them.

The pilot installation and customization took only three days from which two were spent working with the XBRL functionalities and one with other features of the software. Company C had a developer come in from their R&D team in Canada to help with the pilot and to make sure that the product can support the new taxonomy and that the new taxonomy is well-defined in every aspect. In addition to this, TIEKE was heavily involved in the pilot because they also wanted to see that their newly developed taxonomy was working as intended. The two XBRL experts worked closely together, solving issues in both the taxonomy and the product. After the new taxonomy was correctly imported into the software product, the work for Company C's specialist in the pilot was more about getting familiar with all the features that the new taxonomy had to offer and what kind of data is needed to be mapped into XBRL elements.

After getting to know the taxonomy better, the specialist from Company C was able to work with two accounting specialists from the client company and to successfully create XBRL reports from the client's input data. Before the three-day pilot implementation started, none of these three specialists had had any previous experience in XBRL. The specialist from Company C was almost amazed how easily the whole pilot went considering that they were among the first companies to actually use the new XBRL/SBR taxonomy. The close attention from the XBRL expert from TIEKE, the support from their own R&D, and active participation and input from the client company made the pilot successful for Company C.

The specialist estimates that overall the project has been a positive experience for them. The company's local team was excited to see that they have the XBRL component, that has been developed already years ago far away from their Finnish customers, and when they now imported the new taxonomy into it, everything just worked effortlessly and they were able to tag Excel cells into XBRL elements. After the pilot, the company put their new XBRL/SBR taxonomy functionality into production and the reporting feature is now available for their customers to use when they choose so.

5.1.4 Case 4, Company D

Company D is a startup company that has been operating since January 2014. The company has developed a new accounting information cloud service from scratch with an emphasis on a product that is as digital as possible. The company aims to leverage their position where they can develop a completely new system from start to finish and offer a competitive digital service in a market that has a lot of inertia and where a lot of information is still being moved around and between systems manually, either on paper or with various intermediary file formats. At the moment, the company has individual users in the range of a few hundred. The interviewee is the CEO and co-founder of the company, whose main responsibilities are sales, product management, and customer support. The CEO has a long history in dealing with various accounting systems and also a good knowledge of competing systems. Company D believes that by focusing on the customer experience and digitalization, there is an opportunity to gain business form larger incumbent firms.

The company is focused on selling their cloud service to accounting firms that are billed on the basis of the number of individual user licenses. The company distinctively does not offer any type of accounting services or accounting consulting but rather limits itself to developing the software and providing technical support for it. The product has been recently launched into production but its development is still very much actively ongoing. Software development in the company operates in four-week Scrum sprints and new versions of the software are released to production after each sprint. The company's personnel – consisting of the CEO and three developers, two of which are consultants – does not work in a fixed location together, but instead the company is organized as a virtual team, communicating mostly by using Slack (an online group chat and collaboration application) and Skype and coordinating their software development processes by using various other online services. The CEO is in charge of creating software requirements and inputting them into the company's shared JIRA (a proprietary issue tracking product) system. After the developers

have created a new functionality, the CEO will then often test it and accept the new feature into production.

The CEO had been attending to various XBRL meetings and working groups starting from 2014 and the idea of implementing XBRL/SBR reporting functionality into their product had been growing gradually over time. The company wants to position themselves to be forward thinking and an early adopter of new technology, so being able to have one of the first products that supports XBLR/SBR reporting was seen as a key selling point in their favor. The actual decision to implement XBRL reporting was made in late 2014 and the company decided that the functionality will be deeply integrated into the product in such way that the product can internally handle all the accounting data by using the standard business reporting code set as its chart of accounts.

When a new company entry is created into the accounting system, the product offers the possibility to choose either a traditional chart of accounts or the new standard business reporting code set chart of accounts as the basis of the day-to-day accounting entries. The company plans to support the regular charts of accounts so that the non-XBRL accounts will be mapped into XBRL equivalents in the internal database, and in this way, everyone who is using their product can benefit from the XBRL functionality. The motivation to integrate XBLR and the standard business reporting code set so deeply into the product came from the realization that creating and filing financial statements is a central feature to an accounting information system. Since the company positions itself to be a forerunner in digitalization, it wants to make sure that they are well positioned to support the new electronic reporting when it becomes available. They believe that it will create real added value to their customers when they can, for instance, create and file their tax reports automatically. The CEO says that the XBRL/SBR reporting initiative is digitalization at its best and it is great to see that the tax administration is supporting it.

When the XBLR implementation started, the company was actively doing software development on their first release of their product. The CEO and one developer gathered the necessary requirements together with some help from TIEKE. At the beginning they had doubts whether the standard business reporting code set could really be used as the chart of accounts and whether their idea of enabling a one-to-one mapping between their regular accounts and XBRL elements would really be feasible. The CEO says that they worried that will the standard business reporting code set be maintained actively enough and with proper resources in the future. These doubts have been since cleared with the timely release of each

yearly update of the standard business reporting code set, but at the beginning they were real concerns. The company didn't seriously consider any other approach than integrating the XBLR functionality deeply into their application because the feature has such a central role in their product's feature lineup that they felt that an external XBRL report generator or a similar approach would not be robust and flexible enough. They also knew that since the whole XBRL/SBR domain was so new, there wouldn't exist an off-the-shelf solution that would help them with their implementation, so the only real option was to implement it by themselves.

The main XBLR implementation was done over the course of a year by a single developer based on the specifications that the CEO provided. The XBRL implementation was part of the company's other ongoing development activities and the CEO estimates that altogether one month was spend on the actual implementation work. The organization is small and lean and tends to avoid unnecessary processes. The whole requirements planning for the XBRL functionality was done over Slack and the CEO, operating as a product manager, wrote down the final requirements to JIRA. The developer, who wrote the functionality, is a subcontractor for Company D and has been working with them since the beginning of the startup. No one in the company had any prior XBRL experience before they decided to start the implementation, so at first a lot of time was spent of getting familiar with the technology.

After the initial specifications were done and the company had resolved their concerns about the use of the standard business reporting code set as the internal chart of accounts, the project proceeded without major issues. During the development, the developer used an XBRL validator, Arelle, to validate the XBRL documents that the software generated and also sent some of the documents to the tax administration for validation. The CEO says that their work would have been easier if the official interfaces for submitting and validating XBRL/SBR reports would have already been available, but they managed to make progress fairly well by using only the offline validator. At the time of the interview, Company D's XBRL implementation wasn't still in production but they had customers who were already using the standard business reporting code set chart of accounts and they had successfully sent a valid XBRL report to the tax administration. In addition to waiting for the receiving interfaces to open, some user interface changes are still pending before the whole XBRL functionality can be put into production.

The CEO says that the whole XBRL implementation project has been fairly challenging when compared to many other interfaces and file formats that are used in financial

administration and accounting applications. One difficulty with an XBRL report is that it has to be completely valid before it can be accepted, while many other formats accept messages and files that are not fully well-formed. The fully-validated approach of XBRL requires a certain type of rigor that has not been previously necessary. When asked if the CEO would do something differently if they would take on the XBLR implementation project again, the CEO says that they would possibly reconsider their approach of using the standard business reporting code set as the chart of accounts. The standard business reporting code set is something that is outside of the company's direct control and it affects the way in which the users of the product conduct their daily operations. The CEO is concerned that they will be facing unpleasant surprises down the road when the users have not been entering their data correctly according to the standard business reporting code set and their accounting data will not be properly formed because of this. There are also some aspects of the reporting code set that makes it unpractical to work with, for example, the account numbers are seven digits instead of the standard four. This can be inconvenient for accountants who are used to entering the account numbers by hand and remembering them by heart. The CEO wishes that future releases of the standard business reporting code set would drive the definition towards something that would be better suited to be used as a chart of accounts.

5.2 Case Comparison and Analysis

While all of the four projects that were observed were aiming to implement a similar set of capabilities, each of them had different backgrounds for the projects in terms of the companies' technical and organizational characteristics and their motivation for the projects. Interestingly, each of the companies took different types of approaches for their implementations even though the external characteristics of the projects were all similar. This following section presents a cross-comparison of the cases and reflects the findings against the theoretical framework described in section 3.3.

5.2.1 Comparison

Starting from the selected implementation strategies, all cases exhibited slightly different approaches to solving a similar problem. These approaches are summarized in Table 7.

Table 7, Implementation Strategy Overview

Company	Strategy	Input for XBRL component	Level of implementation
Company A	Bolt-on, external component for XBRL translation	SBR code set	Proof-of-concept
Company B	Bolt-on external component for XBRL translation	SBR code set	Limited proof-of-concept
Company C	Integrated XBRL component that imports new taxonomy	Mixed input	In production
Company D	Integrated part of the application	Internal database	In production

Company A and B chose to approach the XBRL implementation with an external bolt-on component that takes in data in the standard business reporting code set format. They both stated that their intention would be in the future to have a translation component that supports some form of tagging so that they could also input the data in regular account format and have it then be translated to respective XBRL tags. Both of these two companies also limited their implementation to proof-of-concept stage and didn't bring it into production yet. Company A stated that they don't see any immediate benefits from fully implementing the standard at this time and that they will wait and see how the initial wave of adoption will turn out before doing any further decisions. Company B was also fairly unenthusiastic about the current benefits of supporting XBRL/SBR reporting and they said that they will rather wait and see what will happens before doing any real decisions about it.

Company C had already previously implemented a deeply integrated XBRL functionality into their product and adding the new taxonomy turned out to be a fairly trivial exercise. Since Company C sells their product across multiple countries and legislations, they have seen it as a good investment to integrate the XBRL functionality deeply into their product so that they can support multiple taxonomies. Now they enjoy the benefit of being able to add support for a completely new taxonomy with just by importing a new definition file. Company D has also chosen to integrate XBRL functionality deeply into their product and they see that it is going to bring their customers added value in the near future by being able to file reports to the tax administration and other regulatory officials in a fully digital format and also by being able to partly automate the reporting process. The main motivations behind each implementation strategy decision are presented in Table 8.

Table 8, Motivation for Implementation Strategy Selection

Company	Motivation for strategy decision
Company A	Uncertainty of the future benefits of XBRL/SBR reporting
Company B	Deliberate wait-and-see decision
Company C	Need to support functionality in multiple countries
Company D	Embracing digitalization

Each of the four companies also had differences in what kind of teams they chose for doing the implementation work. These differences are illustrated in Table 9. Company A hired a new summer trainee to do the proof-of-concept implementation. Although they appointed the task for a trainee, they emphasized that the person doing the implementation will have to have enough domain knowledge in accounting in order to be able to properly perform in the task. The summer trainee worked in close collaboration with the CTO of the company and also had support from other people who were familiar with the main product of the company. Company B had their proof-of-concept done by a completely external student group who didn't have access to the internals of the company's product. The implementation had a strict and well-defined boundary and it was limited in functionality; the external "black box" application can read a file that is in a pre-determined format and produce two report forms into XBRL instance documents.

Company C's implementation was more of a verification exercise than an actual software implementation and it was done by an experienced in-house XBRL developer together with an XBRL expert from TIEKE. The implementation was to do as much with testing the new XBRL taxonomy as it was with validating that the company's product could support XBRL/SBR. Lastly, Company D developed their XBRL functionality as part of their ongoing software development efforts. They had a product manager and a developer who were handling the XBLR features as a team. For them, the XBRL functionality was a core feature that they needed to implement into their product.

Table 9, XBRL Implementation Teams

Company	Implementing team/individual
Company A	Summer trainee
Company B	External group of students
Company C	In-house and external XBRL experts
Company D	In-house software development team

On the topic of project success factors, each company was asked to list what difficulties they had had and what kind of factors had helped them most during the implementation. These factors are listed in Table 10. All but Company C stated that one of their main difficulties was the complexity of the XBRL format. Companies A, B and D felt that getting to know the standard and the SBR taxonomy in-depth proved to be more challenging than they had expected because of the feature-richness of it. Also the fact that XBRL/SBR is self-validating, meaning that the XBRL instance document has to be exactly right in order for it to work, caused some difficulties. For success factors, all of the four companies listed the help that they had gotten from external XBRL experts, mainly TIEKE. They felt that the active role of TIEKE as a facilitator helped them familiarize themselves with XBRL more quickly and also helped them with overcoming technical issues during the implementation. Companies A and B also mentioned that having accounting domain knowledge in their software development teams was important for their implementation projects.

Table 10, Difficulties and Success factors During Implementation

Company	Difficulties	Success factors
Company A	Complexity of XBRL format	Accounting domain knowledge
		External support with XBRL
Company B	Limited communication between stakeholders	Accounting domain knowledge in implementation team
	Miscommunication issues	External support with XBRL
	Complexity of XBRL format	
Company C	None mentioned	Being able to get contribution from both in-house and external XBRL experts
Company D	Complexity of XBRL format	External support with XBRL
	Challenges with fitting SBR code set as the chart of accounts	Being able to use an offline XBRL validator

When asked whether they would have done anything differently in the project in hindsight, companies A, B, and D had some aspects of the that they would have change. Companies A and B would have incorporated tagging functionality into their implementations instead of having the XBRL component take in standard business reporting code set values. Company D would have reconsidered their approach of using the standard business reporting code set as one of the optional chart of accounts and had instead had a tagging functionality that would enable the translation between account values into XBLR/SBR elements. Essentially each of these three companies would have wanted to choose a similar approach with the input format; having the XBRL component take in normal accounts and being then able to define the translation from these values to standard business reporting code set values. However, despite these comments, each of the four companies evaluated their project to have been an overall success. Only Company B lamented their limited commitment to the project and the communication difficulties that they had had, but otherwise none of the companies gave negative notes on their projects.

5.2.2 Analysis

The implementation strategies in the four cases can be categorized, as discussed in section 2.1, according to the depth of the implementation and the different XBRL implementation strategies that are identified by Garbellotto (Garbellotto, 2009a; Garbellotto, 2009b; Garbellotto, 2009c). The categorization of implementation strategies however is not completely straightforward; Company D's implementation for instance shows signs of

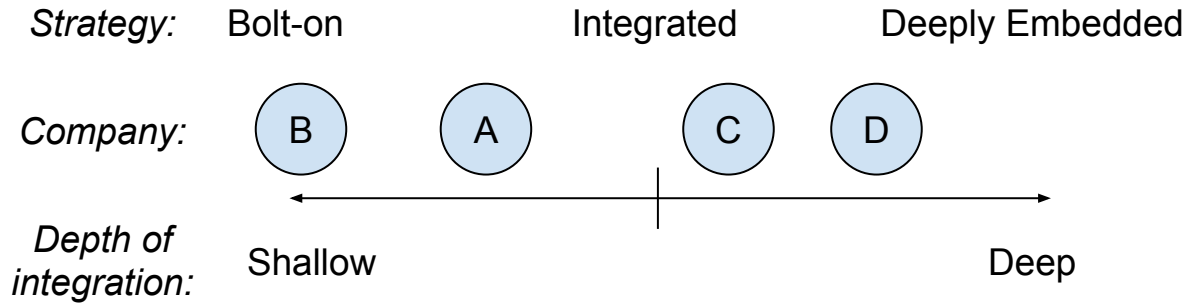
Garbellotto's "deeply embedded" strategy, such as using XBRL/SBR elements in the product's internal data formats, but without knowing more about the internal architecture of the product, it is not easy to make a definite statement about whether the implementation is more built-in than deeply embedded. Companies A and B had the clearest cases of bolt-on implementations but here again Company B's implementation is more strongly categorized as a bolt-on strategy than A's because the developers in Company B's case were completely external to the company and they didn't have access to the internal working of the company's product. The different implementation strategies used by the case companies are presented in the table below.

Table 11, Implementation Strategies in Case Companies

Company	Depth of Implementation	XBRL Implementation Strategy
Company A	Shallow	Bolt-on
Company B	Shallow	Bolt-on
Company C	Deep	Built-in
Company D	Deep	Deeply embedded

An alternative approach to categorizing the implementation strategies is to divide them into various degrees of depth of integration as illustrated in Figure 6. The deeper the integration in the implementation is, the more embedded the functionality is into the product and also, presumably, the more central the functionality is for the product. An interesting aspect of the two observed shallow implementations is that both companies A and B, that did a shallow implementation, didn't intend it to be their final version of the functionality but rather the starting point. This implies that the companies are taking an iterative approach to developing the functionality as discussed by Fichman & Moses, 1999 and Petersen & Wohlin, 2010. Company D on the other hand intended its deeply embedded implementation to be the final production ready version.

Figure 6, Depth of XBRL Implementation in Case Companies



When comparing the reasons behind the shallow versus deep decision, it is telling that Company D stated that one reason for choosing the more embedded approach was that they were doing a new product and decided to use the opportunity to properly support XBRL. This means that Company D didn't have existing software in production and historical technical decisions that would generate excess inertia for them. This brings in the concept of path dependence that is introduces in section 2.3; companies that have longer history of developing their products are less likely to do dramatic changes or make big commitments to new technologies due to reasons in their existing technical environments and organizations than are companies that are starting essentially from scratch. Company D was more unconstrained in making a decision about the implementation strategy than were companies A and B who chose to start small and iterate over time to find an approach to XBRL that would fit their products the best.

In addition to the path dependence, reasons directly affecting the implementation strategy decision were mentioned to have been expectations on the current and future benefits of implementing XBLR/SBR functionality. Due to the inter-organizational nature of the technology, the actions of the tax administration and other users of the technology affect these factors. The CTO of Company A implicitly said that they were expecting a sign from the legislator that the standard business reporting code set would become a new uniform chart of accounts in Finland or at least the new accounting act would somehow acknowledge XBRL. When this didn't happen, the company felt that they would be better off by taking a less active approach to the new technology and wait until they see where it is headed and what kind of benefits it will offer in the future. The CTO also mentioned that they are expecting that the tax administration will introduce some compelling reason for companies to adopt the new XBRL/SBR standard because they see that it is not going to happen in a marked driven way by itself. Company D also justified their implementation strategy decision with future expectations on the XBRL/SBR technology; their take on the topic was however somewhat different. The company believes that by supporting XBRL reporting, they will be able to bring their customers new and attractive features in the near future that wouldn't otherwise be possible.

The expected benefit regarding implementing XBRL/SBR originate from the network effects of the technology, the future expectations for the technology, and also the marginal cost and value of implementing it. At the moment of the implementations, the network effects were more or less the same for every company but the future expectations regarding the

technology were different in each case. Companies A and B didn't have a clear take on the topic, whereas C and D felt that it is something that needs to be supported in the near future. The marginal cost of the implementation plays into the expected benefits through the cost to payoff ratio of the project; if a company is expecting that the project will e.g. bring additional business, they will be more likely to justify the cost of a more expensive integrated implementation strategy. Company D felt that it is able to sell its product by highlighting some of its XBRL functionalities, whereas Company A stated that no one will buy their product because of XBRL by itself.

Each of the four projects exhibited various degrees of the main software project success factors that are presented in Table 4. Every project, except Company C's limited localization effort, used a proper software development process. In terms of the skill-level of the implementation team, it can be only speculated whether Company A's summer trainee or Company B's student team had significantly different skill levels than the full-time development teams at Company C and Company D. Table 12 summarizes the differences that each project showed in terms of project planning, management support, and stakeholder communication. Company D was the only one that had all three success factors on its side: a proper documented planning phase, active management involvement, and active communication channels between the project's stakeholders. Out of the four, Company B reported to have had the most difficulties during their project, most of which were related to insufficient communication. Company A implicitly acknowledged that had their planning phase been more thorough, some unexpected issues could have been handled earlier in the project.

Table 12, Software Project Success Factors in Implementation Cases

Company	**Project planning**	**Active contribution from management**	**Communication between stakeholders, including users**
Company A	Brief planning phase	Yes	Active
Company B	No formal planning phase	No	Limited
Company C	No formal planning phase	No	Active
Company D	Documented planning phase	Yes	Active

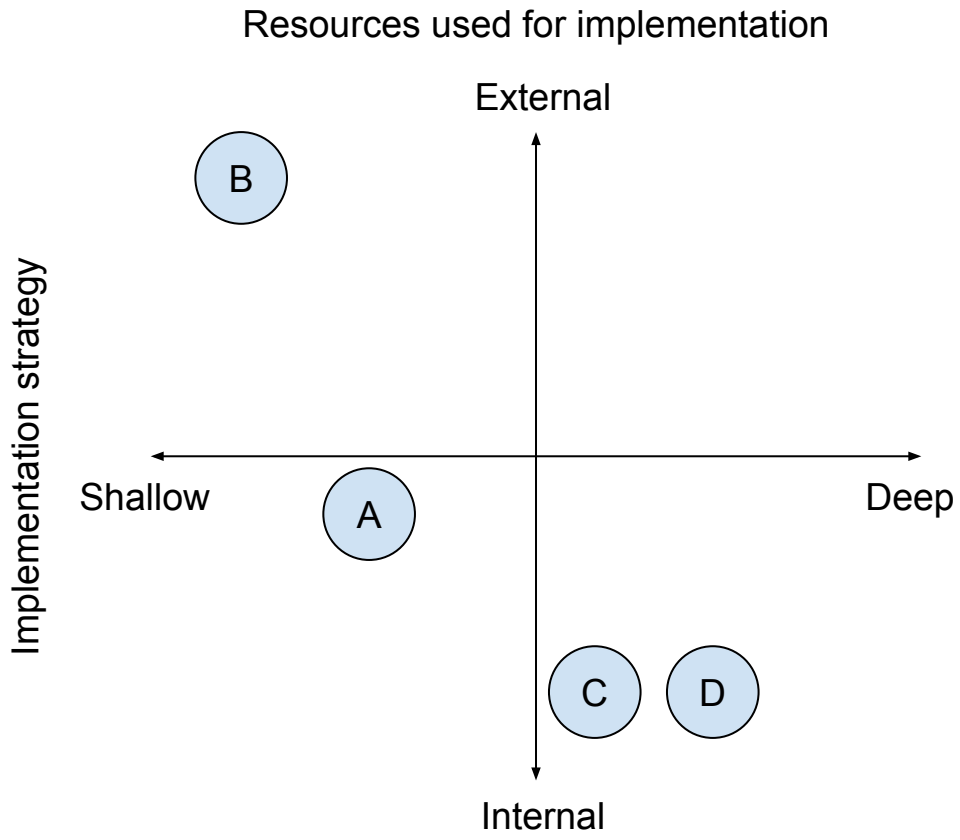
Having all the central software project success factors didn't however guarantee Company D a project without setbacks. The company reported that they had issues with using the standard business reporting code set as the chart of accounts and they were anxious how the standard business reporting code set would be updated and maintained in the future. They said that if

they would have the opportunity, they would likely reconsider their approach. This shows that even though project success factors are important, the central software design and selected software implementation strategy can define aspects of the project that cannot be changed by simply following the best practices for software development projects.

5.2.3 Observations on the Implementation Strategies

The strategies that the observed companies chose when approaching their software implementation projects follow some aspects of the implementation strategies that are identified in previous literature, but rather than being discrete categories, the observed approaches were more of a continuum between different extremes of completely bolt-on and deeply embedded. Based on the four cases, some comparisons can be made between the shallow and the deep implementation strategies. Figure 7 shows the observed relationship between the different implementations and the use of external and internal resources.

Figure 7, Implementation Strategy and Resources Used for the Implementataion



Company B used a completely external team for their proof-of-concept and Company A had a summer trainee, whereas Company C and Company D had their implementations done by their full-time developers. The implementation team selection with the deeply integrated strategies can be considered to be affected by the implementation strategy itself; a deeply

integrated solution cannot easily be developed without full access to the internals of the software product and the implementation is going to require support from the main development team. However, with the shallow, bolt-on implementation, it is not clear whether the implementation strategy affects the team selection or the team selection affects the strategy decision. Company B chose to do a bolt-on solution because they had the external student group doing a project for them and the company saw that the only way they could do an XBRL/SBR reporting component was to make it as an external component. Company A could have chosen to have their implementation done by their regular development teams, but they rather hired a summer trainee for the job. Table 13 illustrates these different approaches to the implementation team selection.

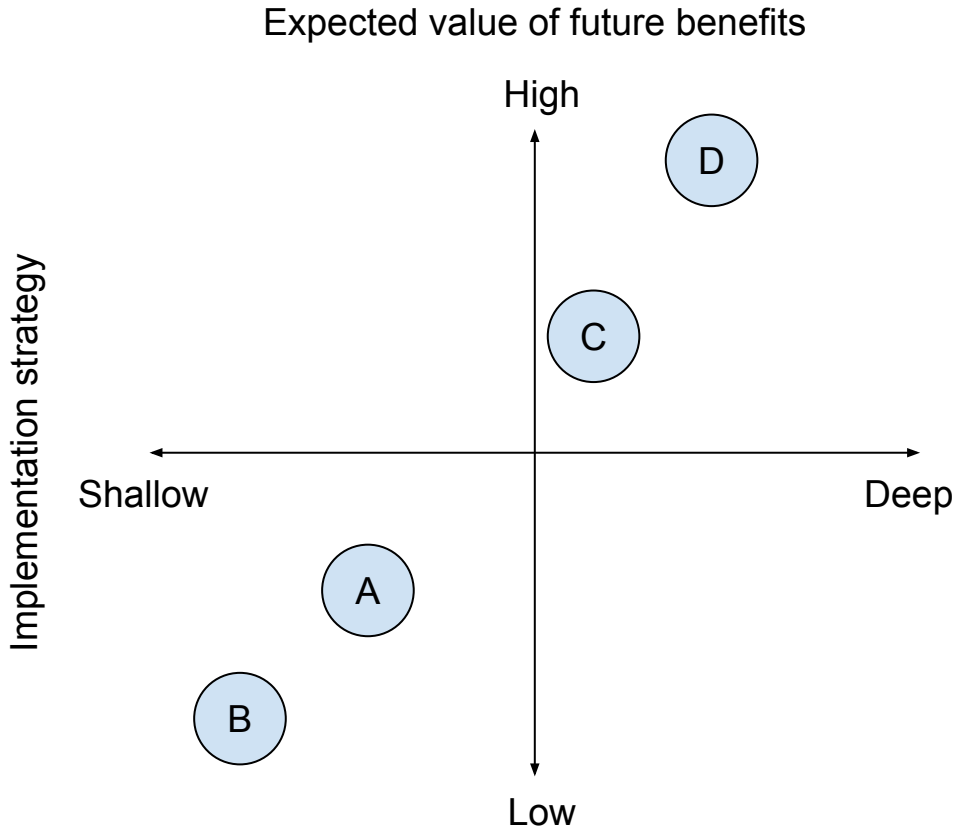
Table 13, Implementation Strategies and Selection of Development Team

		Development team	
		Internal	External
Level of integration	Shallow	Internally developed bolt-on	Externally developed bolt-on
	Deep	Internally developed deeply embedded	Not feasible

The reasons for selecting different implementation strategies followed the theoretical framework to some extent, but in the observed cases the uncertainty about the future of the technology was one of the most prominent guiding reason for the implementation strategy selection. In an uncertain environment, two out of four of the companies chose to wait and play it safe, postponing larger commitments and selecting a fast bolt-on proof-of-concept project instead of investing into a more integrated approach. One motivation for this kind of proof-of-concept approach can be though to be to learn about the technology without committing too much resources into it. After the proof-of-concept project, when the time is right for the real production implementation, the company is in a good position to make educated decisions on how to approach it. Figure 8 illustrates how each company approached the XBRL/SRB implementation in terms of their perceived certainty of the future benefits and the level of expected benefits for the technology and the depth of the selected implementation approach. This measure of the certainty and the level of expected benefits can be considered as the expected value of the future benefits for the technology. Even if the

potential of a technology would be seen as high, there might be too much uncertainty regarding it that it weighs down the overall expected value of future benefits. Or similarly, if the perceived benefit for a technology is relatively low but it is certain to realize, then the overall expected value of future benefits might still be high.

Figure 8, Implementation Strategy and Expected Value of Future Benefits



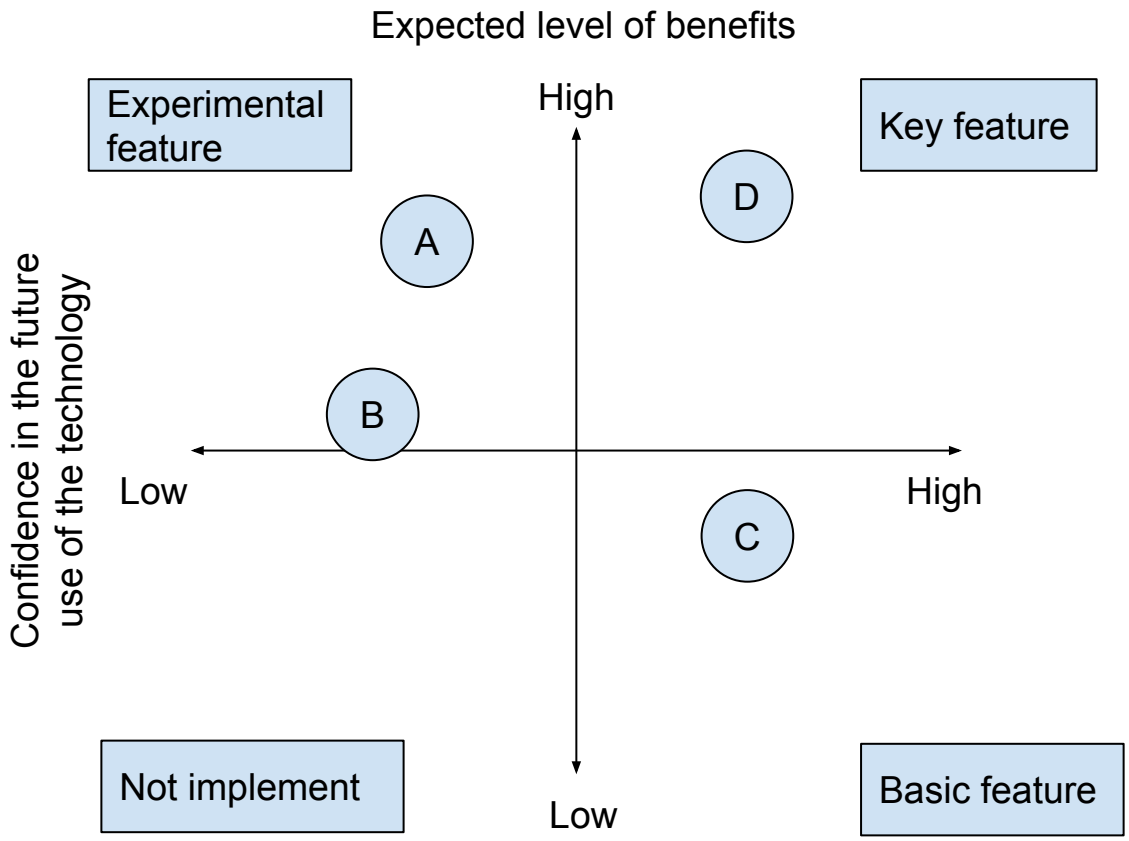
Company C and D both had strong feelings about the future expectations of the XBRL/SBR. However, where Company D saw that XBRL is clearly something that they can use in their advantage while marketing their product, Company C felt that XBRL support is something that just needs to be there in order for them to be able to sell their product but it is not something that alone would be a highlighted feature. Companies A and B had both relatively high uncertainty about the future of XBRL/SBR reporting. Company A was more invested into XBRL than Company B since it had been following the technology over the course of many years and understood the benefits of it very well, but still Company A lacked the certainty that the technology will be widely used and useful in the near future.

If a company evaluates a technology to have a low expected value of future benefits, it can still choose a shallow bolt-on implementation strategy instead of deciding not to implement at all. However, in the same situation that company is unlikely to choose a deeply embedded

implementation that would mean spending more resources to pursue the same level of uncertain benefits. On the other hand, if a company feels strongly that a technology is going be important for them and if they have the right resources for it, they can choose a more deeply integrated strategy if they feel that it is fitting for the application. This does not however mean that it is always necessary to do a deep integration. There might be valid scenarios where a shallow implementation is justified if it is enough to capture the value from a technology that has a high expected value of future benefits.

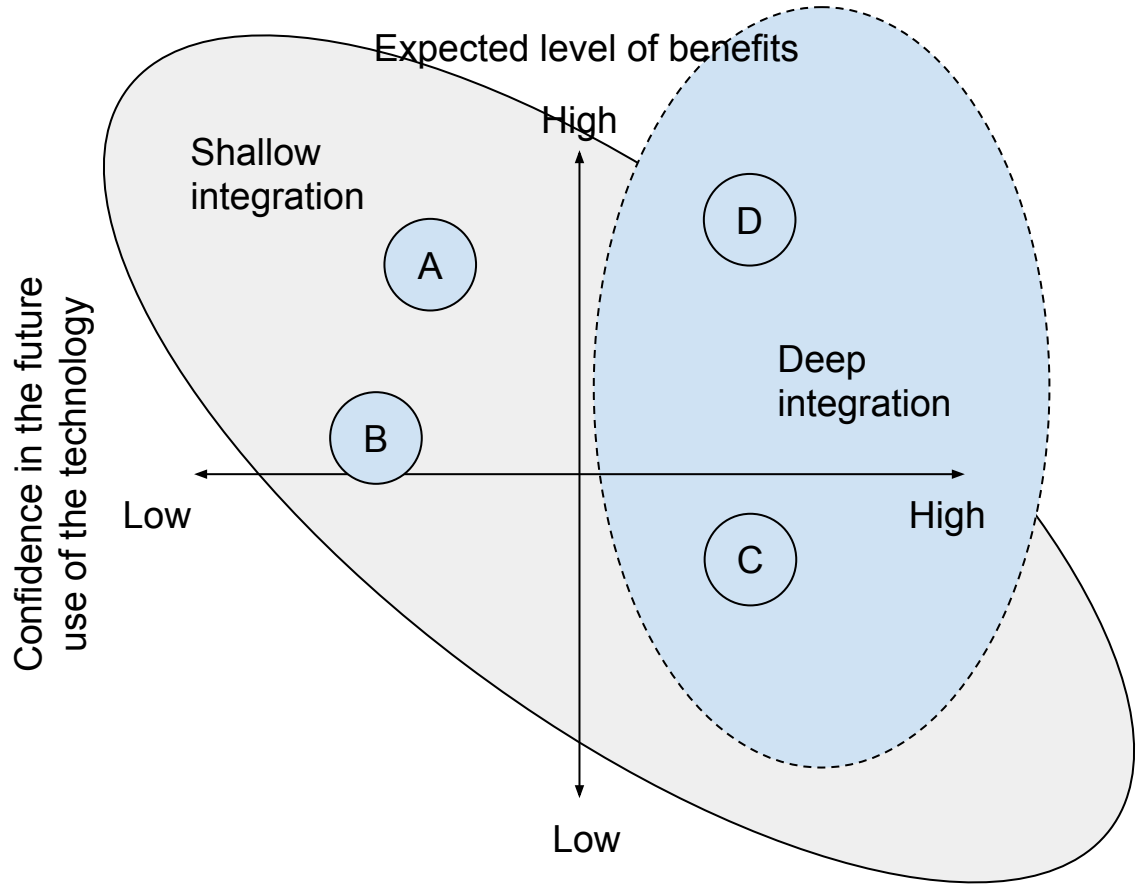
Since the concept of expected outcome of future benefits in Figure 8 combines two different factors, it is interesting to break it down to its components and examine the case results in respect to each company's expected level of benefits from XBRL/SBR and their respective confidence in the future use of the technology. This comparison is presented in Figure 9. In this illustration, each corner of the graph can be given a descriptive name that characterizes the nature of a functionality that falls into that particular quadrant. Company A and B are in the "experimental feature" quadrant, Company C is in the "basic feature" quadrant, and Company D is in the "key feature" quadrant. Naturally, none of the companies in the case study does not fall under the "not implement" quadrant where the expected level of benefits is low and the confidence in the future of the technology is also low. These feature names are consistent with each of the case company's own view of the nature of their XBRL implementation; Company A and B experimented with their proof-of-concepts, Company C saw XBRL as something that has to be there but does not in of itself sell any products, and Company D saw XBRL as a highlight feature that can be used to gain more customers.

Figure 9, Expected Level of Benefits from XBRL implementation and the Confidence in the Future of the Technology



Now, when the comparison graph of expected level of benefits and confidence in the future use of the technology is augmented with the level of integration in the software implementation, the illustration in Figure 10 is formed. Here it can be seen that the shallow and deep integration levels overlap where the confidence in the future use of the technology is high but the expected level of benefits from the technology is lower.

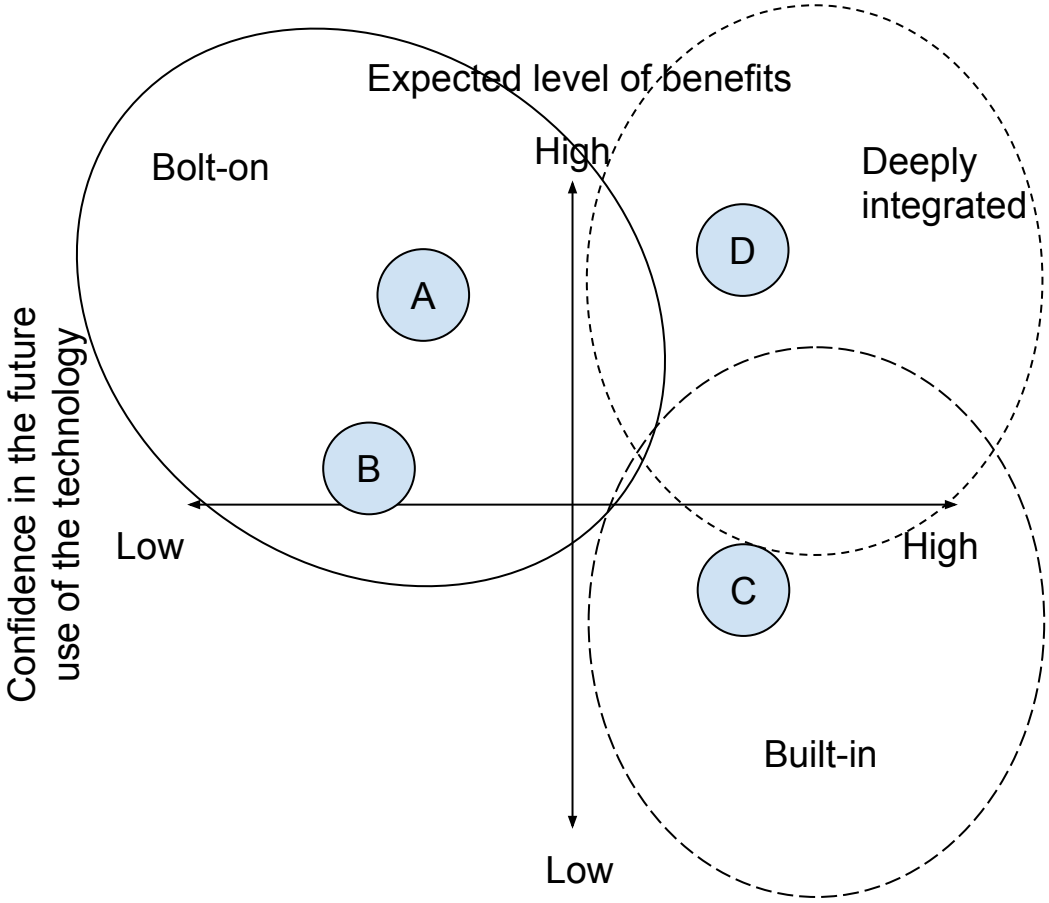
Figure 10, Software Implementation Integration Levels



Similarly as in Figure 10, Garbellotto's software implementation strategy categories (Garbellotto, 2009a; Garbellotto, 2009b; Garbellotto, 2009c) can be overlaid against the comparison graph of expected level of benefits and confidence in the future use of the technology as shown in Figure 11. In this illustration, each category occupies roughly one of the three viable quadrants. Thus Garbellotto's categories match with each of the software feature description that are shown in Figure 9:

- Bolt-on – Experimental Feature
- Built-in – Basic Feature
- Deeply Embedded – Key Feature

Figure 11, Software Implementation Integration Categories



6 Conclusions

The objective of this study was to find out the ways in which companies approach implementing inter-organizational functionalities into their software products and services and to identify the factors that affect their choice of these strategies. The study was descriptive, focusing on multiple cases of inter-organizational software implementation projects. The first wave of XBRL/SBR adoption in Finland among companies that develop accounting information system software offered a unique opportunity to study multiple software projects that were happening around the same time, all of which were aiming to implement similar functionalities and were using a common technology standard. The results of this study are relevant for any organization that wishes to implement XBRL/SBR functionality, but also, due to the general nature of the developed theoretical framework, the results are also applicable to various other types of inter-organizational software projects.

The theoretical framework that was developed for this study included external and internal factors that were expected to affect the implementation strategy decision in inter-organizational software project. Theories included in the internal factors were path dependence and excess inertia. The external factors included the theory of network externalities. The study also considered general software project success factors when evaluating the results of the studied cases. The main body of the empirical data for the case study was collected through five interviews that focused on four different companies.

The results of the case study show significant differences in the ways in which companies approach similar software implementation projects. Implementation strategies in inter-organizational functionalities, as well as other software, can be categorized in term of the depth of the implementation, between shallow and deep integration. The aspect of inter-organizationality brings in questions about network externalities, expectations of the future adoption of the technology, and expectations on the future development of the technology. All of these factors are outside of the direct influence of a software implementation project but they play a key part in many of its decisions. These external factors can make the implementation strategy decision more complex when the organization needs to consider how much they are expecting to get benefits out of the technology in the face of uncertainty of its future development and adoption.

Uncertainty of the future adoption, development, and use of an inter-organizational technology is a major determining factor in the implementation strategy decision, stronger

than was initially expected in the theoretical framework. Uncertainty can cause unwillingness to invest heavily into a project and in the face of unknown future expectations, companies can choose a shallow implementation strategy in order to postpone the real implementation decision and to gather more information about the technology in question.

6.1 Limitations and Suggestion for Future Research

A major limitation in this study is the lack of robust triangulation and to some degree, the lack of transparency. These limitations were difficult to overcome due to ethical considerations of maintaining anonymity of the case subjects and also because the events that were studied happened inside of companies and were not part of public records. In order to build a stronger case from an inter-organizational software implementation, more resources should be focused on the case building phase of the study.

This study focused on software implementation projects that all were dealing with the same technology, under similar external circumstances regarding the adoption of the technology and the future outlook of it. There is potential for future studies that compare inter-organizational software projects that have different external factors. Another interesting topic for future research is the role of a facilitating party in such software projects. Each of the observed cases in this study reported that having an active external facilitator helped them greatly in their projects and an active facilitator might also be able to influence the level of uncertainty about the future of the technology that the companies are experiencing.

References

Books and reports

Rogers, E. M. (2010). *Diffusion of innovations* Simon and Schuster.

Yin, R. K. (2011). *Applications of case study research* Sage.

Yin, R. K. (2003). *Case study research: Design and methods* Sage.

Articles

Berntsson-Svensson, R., & Aurum, A. (2006). Successful software project and products: An empirical investigation. *Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering,* Rio de Janeiro, Brazil. 144-153.

Arthur, W. B. (1989). Competing technologies, increasing returns, and lock-in by historical events. *The Economic Journal, 99*(394), 116-131.

Benbasat, I., Goldstein, D. K., & Mead, M. (1987). The case research strategy in studies of information systems. *MIS Quarterly,* , 369-386.

Chow, T., & Cao, D. (2008). A survey study of critical success factors in agile software projects. *Journal of Systems and Software, 81*(6), 961-971.

Cohen, W. M., & Levinthal, D. A. (1990). Absorptive capacity: A new perspective on learning and innovation. *Administrative Science Quarterly,* , 128-152.

Dosi, G. (1982). Technological paradigms and technological trajectories: A suggested interpretation of the determinants and directions of technical change. *Research Policy, 11*(3), 147-162.

Dvir, D., Raz, T., & Shenhar, A. J. (2003). An empirical analysis of the relationship between project planning and project success. *International Journal of Project Management, 21*(2), 89-95.

Farrell, J., & Saloner, G. (1986). Installed base and compatibility: Innovation, product preannouncements, and predation. *The American Economic Review,* , 940-955.

Farrell, J., & Saloner, G. (1985). Standardization, compatibility, and innovation. *The Rand Journal of Economics,* , 70-83.

Fichman, R. G., & Moses, S. A. (1999). An incremental process for software implementation. *Sloan Management Review, 40*(2), 39-52.

Fortune, J., & White, D. (2006). Framing of project critical success factors by a systems model. *International Journal of Project Management, 24*(1), 53-65.

Garbellotto, G. (2009a). XBRL implementation strategies: The bolt-on approach. *Strategic Finance, 90*(11), 56-57.

Garbellotto, G. (2009b). XBRL implementation strategies: The built-in approach. *Strategic Finance, 91*(2), 56-57.

Garbellotto, G. (2009c). XBRL implementation strategies: The deeply embedded approach. *Strategic Finance, 91*(5), 56-61.

Garner, D., Henderson, D., Sheetz, S. D., & Trinkle, B. S. (2013). The different levels of XBRL adoption. *Management Accounting Quarterly, 14*(2), 1-10.

Hofmann, H. F., & Lehner, F. (2001). Requirements engineering as a success factor in software projects. *Software, IEEE, 18*(4), 58-66.

Katz, M. L., & Shapiro, C. (1994). Systems competition and network effects. *The Journal of Economic Perspectives, 8*(2), 93-115.

Katz, M. L., & Shapiro, C. (1986). Technology adoption in the presence of network externalities. *Journal of Political Economy, 94*(4), 822-841.

Katz, M. L., & Shapiro, C. (1985). Network externalities, competition, and compatibility. *The American Economic Review, 75*(3), 424-440.

Keil, M., Cule, P. E., Lyytinen, K., & Schmidt, R. C. (1998). A framework for identifying software project risks. *Commun.ACM, 41*(11), 76-83.

Kernan, K. (2008). XBRL around the world. *Journal of Accountancy, 206*(4), 62-66.

Malhotra, R., & Temponi, C. (2010). Critical decisions for ERP integration: Small business issues. *International Journal of Information Management, 30*(1), 28-37.

Petersen, K., & Wohlin, C. (2010). The effect of moving from a plan-driven to an incremental software development approach with agile practices. *Empirical Software Engineering, 15*(6), 654-693.

Reel, J. S. (1999). Critical success factors in software projects. *Software, IEEE, 16*(3), 18-23.

Remer, D. S., & Nieto, A. P. (1995a). A compendium and comparison of 25 project evaluation techniques. part 1: Net present value and rate of return methods. *International Journal of Production Economics, 42*(1), 79-96.

Remer, D. S., & Nieto, A. P. (1995b). A compendium and comparison of 25 project evaluation techniques. part 2: Ratio, payback, and accounting methods. *International Journal of Production Economics, 42*(2), 101-129.

Srivastava, R. P., & Liu, Q. (2012). Special issue of JIS on XBRL. *Journal of Information Systems, 26*(1), 97-101.

Stantial, J. (2007). ROI on XBRL. *Journal of Accountancy, 203*(6), 32-35.

Varadaraj, M., & Goud, N. (2012). Successful software adoption - A study of software implementation methodologies. *International Journal of Computer Applications, 41*(16), n/a.

Wateridge, J. (1995). IT projects: A basis for success. *International Journal of Project Management, 13*(3), 169-172.

Zhu, K., Kraemer, K. L., Gurbaxani, V., & Xin Xu, S. (2006). Migration to open-standard interorganizational systems: Network effects, switching costs, and path dependency. *MIS Quarterly, 30*, 515-539.

Internet-references

Finnish Tax Administration. (2015). Standardoitu talousraportointi automatisoi yritysten ilmoittamista. Retrieved from https://www.vero.fi/fi-FI/Tietoa_Verohallinnosta/Uutiset/Standardoitu_talousraportointi_automatis(38671)

Moody's Analytics. (2011). Common reporting framework (COREP) - FAQs. Retrieved from http://www.moodysanalytics.com/~/media/Brochures/Enterprise-Risk-Solutions/UK-COREP/2011-30-11-MA-COREP-FAQ.pdf

Rintala, M. (2015a). XBRL tilannekatsaus ja jatkovaihesuunnitelmat. Retrieved from https://www.vero.fi/download/Ohjelmistotalopaiva_XBRL_2152015/%7BAB125D54-DD95-4AE0-AC7C-8DEB49514BE6%7D/10824

Rintala, M. (2015b). XBRL-ilmoittaminen, tilannekatsaus . Retrieved from https://www.vero.fi/download/XBRLilmoittaminen__tilannekatsaus/%7BBD7BF63D-C108-4B76-A074-E93F70E73850%7D/11322

XBRL Finland. (2015). XBRL suomi. Retrieved from http://xbrl.fi

XBRL International. (2016). An introduction to XBRL. Retrieved from https://www.xbrl.org/the-standard/what/an-introduction-to-xbrl/

Appendix A: Interview Questions

Before the interview

- The interview is being done for a thesis for Aalto University's School of Business.
- The names of the interviewed people and companies will not be published.
- The interview is semi-structured. The interviewee can add in relevant information at any point of the interview, even if the information is not directly asked for.

Background of the implementation project

- Company
  - In brief, describe your company and the organization that you work in.
  - How many users does the company's products serve?
- Person
  - What is you background in the organization?
- Product
  - In brief, describe your product or service.
  - What are the central features of your product/service?
  - In broad terms, describe the architecture of your product.
  - Has the product been developed completely in-house?
- Starting the XBRL/SBR project
  - How familiar XBRL and XML was for your organization before you started the XBRL/SBR implementation project?
  - When did you decide to take on the XBRL implementation project?
  - What were the main motivators for the project?
  - Did you use any external advisors or experts when making the decision or when the project was being planned? If so, did you find it useful? How did you select the advisor?
  - How did your company plan for the project and how was the decision to start the project made? Who were involved in the decision making?
  - What were the expected benefits for the XBRL implementation?

- What kind of implementation strategy was chosen and why?
- Was there any unexpected surprises and challenges involved in the project decision?

Implementation strategy details

- How did you decide to incorporate XBRL reporting functionality into your system?
- What were the most important factors that supported your choice?
- What did you think would be the greatest challenges with the strategy that you chose?
- How was the final decision about the implementation strategy made?

Timeline of the project

- What type of software development methods were used in the project (agile, waterfall, etc.)?
- How was the requirement planning phase done? Who were involved in it and when was it done? Did you use any external resources for the requirement work?
- How did you form the team that did the implementation?
  - What kind of roles were in the team?
  - Who were the team members?
  - What kind of backgrounds and positions did the team members have?
  - What kind of experience did the team members have about XBRL before the project?
- Timeline
  - Describe freely the timeline of the project from phase to phase. The phases can include for example requirement planning, implementation, validation, and production.
    - For each of the identified project phase, describe the following:
    - Who were involved and in what ways?
    - What kind of surprises or challenged the team faced?
    - What were the key success factors in each phase?
    - How was the progress of the project reported in each phase? How was the management involved in each of the phases?
    - How did you conclude each phase as completed?

Evaluation of the project

- How challenging would you describe the XBRL implementation project?
- What would you do differently if you would start the project over again?
- What were the biggest challenges in the project?
- What were the key success factors in the project?