Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

LASSE METSO

Effective requirements management in preparation for large software projects

DEGREE PROGRAMME IN INDUSTRIAL MANAGEMENT
2024

ABSTRACT

Metso, Lasse: Effective requirements management in preparation for large software projects
Bachelor's thesis
Degree programme in industrial management
June 2024
Number of pages: 74

Competition in the rapidly digitalizing world has increased the strategic importance and mission criticality of software in organizations' success. Implementing software successfully requires almost always an information technology project together with strong project execution capabilities. In 2012, advisory firm Gartner concluded that only 16,2% of large software projects were considered fully successful, whereas 31% have been declared complete failures. Even though information technology has advanced significantly since 2012, the success rates of large software projects have improved only minimally.

The purpose of this study is to identify and describe the most important factors in requirements elicitation and management process in project preparation phase for increasing the probability of a successful outcome of a large software project. The first research problem is to identify through existing theory the most important quality criteria for requirements expressions in large software project preparation. The second research problem is to empirically study the impact of the identified requirements quality criteria on the success of large software projects. The theoretical framework in this study is a synthesis of the existing scientific research and other literature on the effective requirements elicitation and management process and requirements quality criteria, and on their impact to projects' success.

The empirical part of this study was executed as qualitative multiple-case study. The research data was collected by interviewing eight experienced information technology professionals with semi-structured interview method. Individual case descriptions together with their key findings were first written from each research case, and the findings were then compared to each other and synthesized through a cross-case analysis.

The empirical research showed that the way requirements are elicited and managed when preparing for software projects has an impact on the project outcome. Organizations should invest enough time to identify and analyse the real business needs behind project initiatives to ensure the initial relevance of the subsequent requirements. Business requirements should be first derived to answer how the project itself helps to fulfil the business needs, and all the following user and software requirements should be based on and mapped to them. Understanding how the users currently work and why is very important, but users should not be tasked to write requirements bottom up. Instead, the business transformation goals should be embedded into all requirements.

Keywords: software project, requirements management, requirements elicitation, requirements quality, business needs, business requirements

CONTENTS

1 INTRODUCTION

1.1 Background and motivation

Software applications and services have become an integral part of everyday operations of almost all organizations. When in the past, software was mainly utilized to support internal processes, for example sales, operations, finance, and general administration, today software is ever-increasingly integrated to the actual offering of products and services of the organizations (Eigner & Stary, 2023, p. 15; Raddats et al., 2019, p. 213). This has increased the strategic importance and mission criticality of software in organizations' success (Dumitriu & Mirona, 2020, p. 933).

Implementing software successfully requires almost always some form of information technology (IT) project together with strong project execution capabilities. Even though there are no definitive global statistics on software project failure rates, research shows that IT-projects – especially large and complex – rarely go without any difficulties. IT consultancy and advisory firm Gartner concluded in 2012 that larger IT-projects with budgets exceeding 1 m$ have almost 50% higher risk of failure compared to ones with budget below 350 000$. The conclusion of the Gartner report was that only 16,2% of the projects were considered fully successful, whereas 31% have been declared complete failures (Mieritz, 2012). In 2015, Budzier & Flyvberg (2015, p. 22) estimated in their study that 50% of technology projects don't meet their cost targets.

According to Alami (2016, p. 70), despite the expectations, the advances in technology and technological support for project execution have not reduced the failure rates of large IT-projects. On the other hand, PMI's 9th global project management survey in 2017 (Project Management Institute, 2017) pointed out that organisations focusing not only to the traditional budget, scope, and time,

but on benefits realization management and agile project management approach instead were increasing the rate of meeting their IT-project goals for the first time since 2011. Despite this promising trend, the same research revealed that 14% of all IT-projects are still deemed failures.

Why are large IT-/software projects more likely to fail then? Only a limited quantity of in-depth research exists regarding the real reasons behind the troubled projects (Alami, 2016, p. 63). In Gartner's 2012 study, the three largest reasons to view IT-projects as failures were (Mieritz, 2012):

1. Substantial delay in schedule (28%)
2. High variance of costs (24%)
3. Issues in required functionality (22%)

Alami (2016, p. 68) states that in larger and more complex transformations, meeting project deadlines should not be the main criterion for success. Instead, large transformations should be delivered with roadmaps broken down into multiple smaller projects. For example, in one research case company, there was no strategy to deliver digital transformation. The company simply relied on vendors to provide the hardware and software which was then expected to automatically bring the business benefits. This did not happen. Although there was a solution concept of the mission critical functionality in the beginning, it was not accompanied with precise requirements which would have guided the project on the exact expectations of the business organization (Alami, 2016, p. 69).

Software project failures – especially in the public sector – have also been in the news regularly. The implementation of a new payroll system in the city of Helsinki in 2022 resulted in 4,5 m€ of unplanned costs, huge public image loss and personnel burn outs. According to an external audit, conducted after the crisis had been overcome, there were various problems contributing to the failure. These included poor change management, low resourcing, and technical difficulties. Insufficient project preparation led to unrealistic expectations which then cumulated into a massive bundle of problems in the actual project (Takala et al., 2023).

United Kingdom's second largest city Birmingham had similar failure in 2022. Birmingham's enterprise resource planning (ERP) system implementation project was three years behind schedule and the original cost estimate of 22 m€ had grown to 116m€ (Kailio, 2023). The city was apparently aware that there were problems with the system's functionality regarding the processing of financial and human resource transactions. Still the system went live in 2022, causing the city's accounting function to crash. Together with payroll issues, although not directly related to the ERP project, the chain of events eventually led the whole city going bankrupt (Ylä-Anttila & Kailio, 2023).

It seems that the preparation phase in these example projects didn't produce a clear picture of the gaps between the required and delivered functionality, otherwise the projects most likely would not have started with such unrealistic plans, budgets, and resources. The examples suggest the importance of careful project preparation and the documentation of clear requirements which then help to identify risks and gaps in the expected deliverables early enough.

1.2 Objectives and research questions

The purpose of this study is to identify and describe the most important factors in requirements elicitation and management process in project preparation phase for increasing the probability of a successful outcome of a large software project. There are two supplementing research problems which both are broken down into four research questions altogether.

1. Research problem: *To identify through existing theory the most important quality criteria for requirements expressions in large software project preparation.*
    a. 1st research question: *What are the key characteristics of a well described requirement?*
    b. 2nd research question: *What is the reasonable scope and level of detail for describing requirements as part of the software project preparation?*

2. Research problem: *To empirically study the impact of the identified requirements quality criteria on the success of large software projects.*
    a. 3rd research question: *Does the research data indicate that the identified requirements quality criteria affect the success of software projects?*
    b. 4th research question: *Can other factors in project preparation be identified from the research data that contribute to the success of software projects?*

The managerial objective of this study is to provide insight on where to set focus in requirements elicitation and management when preparing for large software projects, to maximize added value. This includes understanding both the key characteristics of a well described requirement and the reasonable scope and level of detail when describing different levels of requirements.

1.3 Scope and limitations

The perspective of this study is primarily business needs driven and how these needs are then effectively translated into different levels of high quality requirements. This helps the project team to understand the end business goals and to focus on the right priorities for the maximized business value of the project.

To narrow the scope down to large software projects, the following criteria is set for the research cases:

1. Project's planned duration needs to be over 6 months, corresponding to Gartner's 2012 definition of large IT project (Mieritz, 2012).
2. There needs to be cross-functional process integration between business functions which the software impacts. For example, an ERP software implementation meets this criterion.

The intention is to leave out such software projects which can be executed quickly within one functional organization silo. These types of projects lack the complexity of conflicting stakeholder interests and change management is-

sues, and thus may be successful even with both insufficient preparation and poor requirements elicitation and management. Another important scoping decision is to leave out such software projects that are driven entirely by software life cycle management. These are typically major software version upgrades that might be large projects by nature but are driven by technical application management instead of real business needs.

Instead of the governance of a project implementation or its execution or change management, this study focuses on the phase and actions before a software project execution, referred hereafter as project preparation. In this context, it means that there would still be time to change the project scope, schedule, budget, or to even cancel the project entirely. In addition, there would not be yet any commitments towards the development or implementation partners. However, the intention is not to investigate the underlying reasons behind or the validity of the business needs. Although many software projects are most likely initiated also based on erroneous assumptions or false business needs, the simplification is made here that the business needs are seemingly real for the parties driving the software initiative forward. The errors contributing to possible project failure then emerge in the preparation phase that follows and in the selection of the approach that is chosen to fulfil the business needs.

1.4 Structure of the study

In scientific research, four distinct parts of the study can be identified which together form a generally accepted and coherent framework for a research report. These four parts, which are also linked to the structure of this study in the following Table 1, are:

1. Theoretical framework
2. Presentation of the research methodology
3. Presentation of the findings of empirical research
4. Conclusions of the study

Table 1. The structure of the study

Chapter	Content	Purpose
1	Introduction	To orientate the reader to the topic: purpose, scope, and structure of the study
2	Literature review	To review the existing research on the topic, and to present the theoretical framework
3	Research methodology	To present and justify the research method: how the research was conducted
4	Empirical findings	To present the research cases and their most relevant findings
5	Conclusions	To combine the theory and the empirical findings: synthesis, theoretical and managerial implications, and limitations of the study

As presented in Table 1, Chapter 2 will form a theoretical framework for this study which will be summarised in Section 2.4. Here, the first research problem will be solved through the theoretical framework, and the first two research questions will be answered. The methodological choices and the research process are then described in Chapter 3, followed by the descriptions of the research cases and the analysis of their empirical findings in Chapter 4. By combining the empirical results and the theoretical framework, conclusions of the study are finally presented in Chapter 5. Here, the second research problem will also be solved, and the final two research questions will be answered.

2 LITERATURE REVIEW

2.1 Project preparation

During project preparation, the business problem or opportunity is identified, a conceptual solution is defined, and a project is formed to build and deliver the solution to the organization. Large software projects are typically transformations initiated after an opportunity identification either to improve efficiency/-

reduce costs or to generate more value to customers/increase sales. What is perceived as an opportunity is always a subjective view of a specific actor in the organization, a stakeholder, while in parallel the investment resources, both financial and personnel, are typically limited. Therefore, to ensure that the investment has a sufficient return, including trade-off cost, there is almost always an evaluation and decision-making process in place to prioritize the best initiatives to be moved forward to become actual investments and software implementations. (Chemuturi, 2013, pp. 170–171; Sima, 2022, pp. 18–19).

To be able to convert an initiative into a tangible software investment and implementation requires expressions of what the software should be able to produce as an output to meet the needs of the business. These expressions are called requirements (Altable, 2015, p. 64; Halbleib, 2004, p. 8). Robertson & Robertson (2013, p. 1) emphasize that the most important goal of requirements elicitation and specification is to understand the real business problem which then forms the foundation for potential solutions. Instead requirements are often only associated with written requirement documents which are important, but useless as such if the underlying problem is not truly discovered.

Requirements together with enterprise architecture form the foundation on how the business needs will be met. Gorkhali & Xu (2017, pp. 1–2) define enterprise architecture as principles, models and methods utilized to optimize and transform an enterprise's organizational structure, business processes, information systems and IT infrastructure to enable an organization to rapidly respond to changing customer behaviour. In other words, enterprise architecture is the process of aligning the business part of a company with information technology through the integration of processes, organizations, and people (Dumitriu & Mirona, 2020, p. 933). Enterprise architecture both sets boundaries for the proposed software solution and promotes the reusability of already existing requirements (Robertson & Robertson, 2013, pp. 338–339). For example, even though a business stakeholder might be persuaded to invest in a new software to perform a specific process, it might contradict with other existing developments, or the same result might be obtained by further developing a set of existing systems with less cost and complexity (Dumitriu & Mirona, 2020, p.

937). In one of Alami's (2016, p. 66) research cases, a software was designed in isolation from other systems and the vendor gave assurance that it works for certain implementation purposes. However, without the enterprise architecture context, the design failed to consider the critical dependencies as prerequisites for the software to be able to successfully provide the required outputs. Halbleib (2004, p. 10) states that in these types of cases, prior architectural/design constraints should be considered as real requirements of the new development project.

There are several categories and levels of requirements which are typically associated with different stages of the project as well as to the purposes of which the requirements will be utilized for (Heumann, 2003, p. 4). These categories are investigated in more detail in Section 2.2. All types of requirements should ultimately reflect and consider the scope, stakeholders, and goals of the project initiative (Robertson & Robertson, 2013, pp. 43–44). Unfortunately, it is common that a software project proceeds without a clear understanding of the problem it is trying to solve (Davis & Leffingwell, 1996, p. 1). Bjarnason et al. (2011, p. 41) identified in their study that unclear vision of an overall goal, logically broken down into clear requirements, leads to interpretation and power struggles between different units and technical areas rather than constructive communication on how to reach the common goal. In another case study company by Damian et al. (2005, pp. 259–260), requirements were communicated from the stakeholders only as short statements of required features. The features were poorly defined or documented and did not provide enough of the right information. As the features were then not fully understood by the project, it was difficult to plan for and execute the desired change. Ultimately, functional designs of the features can turn out as subjective interpretations made by individual technical designers with no actual linkage to the original business needs (Bjarnason et al., 2011, p. 45; Damian et al., 2005, p. 260).

Requirements also contain constraints. A constraint is a condition that cannot be bypassed. For example, preconditions are the circumstances and inputs the software requires to be able to produce the desired outcome (Jastram & Kara, 2016). This is important for expectation management. There must be a

specific and structured process how the software is utilized to obtain the business benefits. Therefore, a software project always contains an element of change management, and the larger the project is, the larger the change management element is. When the transformation is large and its requirements are complex, substantial amount of time should be invested in requirements elicitation and analysis phase (Atalbe, 2015, p. 64). The invested time can be later recovered with reduced risks since the preparation reduces subsequent complexity. During the period in which requirements are defined and evaluated, stakeholders can explore consequences and the unknowns, thus increasing the probability to reach an informed decision (Alami, 2016, p. 69).

The procurement of technical platforms, applications and professional services is also one of the areas that must be considered in project preparation. If the project requires procurement of technical platforms and/or applications, at least some software requirements need to exist to determine that the procured solution matches the expectations. If the procurement is for professional services, the requirements utilized in procurement should consider vendor delivery capability and service quality. In the end, the procuring organisation must fully understand what they are buying and what is the delivering vendors' role in ensuring the business benefits (Finkelstein et al., 1996, pp. 2–3; Heumann, 2003, pp. 2–4). For example, Alami (2016, p. 65) found in one research case that the requirement analysis was executed in detail only after the contract had been made with the software vendor. Thus, the functional designs were already in place before detailed requirements were known in detail. The vendor had made initial promises which were flawed by the lack of understanding of the precise project needs. Then, regarding the management and quality of the professional services procured, key personnel ended up leaving the project regularly due to constant time pressure and micromanagement from the steering stakeholders. This case example illustrates the need to ensure that both the customer and the vendor share a common goal and understanding of the upcoming project – of its underlying business needs and goals, scope, and the subsequent requirements.

2.2 Requirements management in project preparation

Davis & Leffingwell (1996, p. 2) and Lai & Ali (2013, p. 38) define requirements management as a systematic approach to identifying, organizing, documenting, communicating, managing, and prioritizing scattered and changing requirements. It is continuous work which requires input from and collaboration of all stakeholders (Khan et al., 2013, p. 21; Lai & Ali, 2013, p. 38). The process of managing requirements can determine project success and is especially important in larger outsourced projects and with mission critical software. A typical project will have hundreds if not thousands of requirements. Identifying and specifying those requirements take time and effort but the payback can be substantial because the requirements impact every aspect of a project. Thorough preparation and analysis, forming a solid foundation for the project, can therefore substantially reduce project's risks and increase its efficiency (Altalbe, 2015, p. 64; Halbleib, 2004, p. 8).

Many projects are impacted by either new requirements being added along the way and/or frequent modifications of existing requirements. This is called requirements volatility (Alami, 2016, p. 67). Large and complex software projects are exposed to more volatile conditions because there are more stakeholders involved, often with differing interests and priorities (Khan et al., 2013, p. 23). In addition, longer lead time of a larger project leaves more room for interference along the way. Lai & Ali (2013, pp. 38–39) emphasize that large and complex software projects are often organized in decentralized global teams with lingual and cultural differences which also increases volatility. In other words, the volatility is proportional to the amount of information that requires processing and integration as well as the geographical distribution of teams and stakeholders. In traditional requirements management processes, the working assumption is always that requirements have been fully elicited and documented before moving on to the execution stage and not modified anymore in the execution. However, it is found to be common that under-developed or at least insufficiently documented requirements are utilized as working assumptions in functional and technical designs. As the most complex projects always face unpredictability and uncertainty during execution, it is essential that there are

disciplined processes in place for analysing, approving, communicating, and documenting changes in requirements (Altalbe, 2015, p. 65).

Bjarnason et al. (2011, p. 37) have identified that solid requirements foundation for a project cannot be built on mere documentation since the way of communicating and interpretating requirements plays a vital role in successful project outcome. Studies have shown that many of the requirements management challenges facing large-scale software development are of an organizational and social character rather than technical, and that projects need to be organized in respect of ensuring coordination and communication of requirements from business stakeholders to both project management and to engineering (Claus et al., 1999, Section 4.3; Bjarnason et al., 2011, p. 37). Communication between stakeholders and engineers/architects around the design leads to identification and shared understanding of both requirements and the needed changes. In addition, functional domain knowledge is vital in expressing requirements and designing a solution that will meet stakeholders' needs. In change impact communication affecting multiple areas of the organization, there are always key people named information brokers who can enable effective requirements communication, but can also create noise, i.e. misunderstandings in the process. In addition, responsibility handovers to new roles introduce a risk of losing critical knowledge and awareness. This can lead to misunderstandings and incorrectly implemented features or to misinformed decisions without considering all relevant aspects (Claus et al., 1999, Section 4.1; Bjarnason et al., 2011, pp. 43–45).

Damian et al. (2005, pp. 260–261) and Lai & Ali (2013, pp. 42–46) both identify two specific actions which help in early-stage requirements management phase to improve the quality of project work in downstream software implementation. These are:

1. Requirements analysis sessions to refine the feature requests into more detailed requirements. This includes:
    - Cross-functional team participation.
    - Group analysis sessions.

- High-level design sessions to understand the impact of the requested features on the solution architecture.

2. Revised change management process
    - If a need to change requirements is identified, it is formally recognized in a change request document which describes the change, its extent, and an estimation of effort of its implementation.
    - These factors should be analysed by experts and then considered by decision makers who are expected to review and approve the requested change.
    - If approvals are given by all the mandatory approvers, the change is authorized by the project team and it then proceeds to implementation.

The conclusions of the case study by Damian et al. (2005, pp. 271–272) indicate that the above practices have significant positive impact in later stages of the project beyond merely improving the quality of requirements. In the case study, productivity increased, rework decreased, and the capacity for informed decision making improved. In addition, communication quality increased because the requirements analysis sessions involved extensive cross-functional interaction. Different departments were brought together to discuss specific requirements, providing an opportunity for engineers to understand the responsibilities and concerns of other functional roles in the larger development team to achieve a shared understanding of the feature. Claus et al. (1999, Section 4.1) state as one of their most important findings that the key people in the project team should be involved in requirements management from the very beginning. Even though this might mean more work outside their normal responsibilities, the motivation for the effort is likely to be achieved through understanding that considerable amount of later rework can be saved.

According to Heumann (2003, p. 4), there are different types and levels of requirements for different purposes. These need to be identified, linked, and utilized the right way and in the right sequence to both avoid unnecessary work being done too early, and to ensure that every requirement has a clear pur-

pose. Bigelow (2020) and Khan et al. (2013, p. 23) list three levels of requirements which are:

1. Business requirement
    a. Defines how the solution initiative will meet the business/stakeholder need in a measurable way, i.e. why the solution is being implemented.
2. User requirement
    a. Defines who are the users, what are their expectations and how they interact with the software to meet the business requirements.
3. Software requirement
    a. Defines what the software itself needs to do to meet the business and user requirements.
    b. Can be further broken down into functional, non-functional and domain requirements.

The extent to which requirements should be elicited in each category varies. Raczkowska-Gzowska & Walkowiak-Gall (2023, p. 76) state that the industry and domain where the project is executed is a critical factor here. To focus on the essential requirements, it is important to follow the hierarchical order, starting from the business requirements, complementing them with the most important user requirements and finally matching every business and user requirement with the necessary software requirements (Halbleib, 2004, p. 9). According to Davis & Leffingwell (1996, p. 10) and Heumann (2003, p. 5), all requirements regardless of the category should also contain one or more attributes. Attributes can for example help to classify or group requirements with regards to a specific business area, functional module, use case, or priority.

With regards to utilizing different types of requirements in the procurement most effectively, the correct way is again case and context specific. When forming and negotiating a contract with vendors, Bigelow (2020) suggests combining requirements from all three categories in the most applicable way, considering each vendor's role in the project. According to Chemuturi (2013, pp. 47–48) and Robertson & Robertson (2013, p. 242), when procuring com-

mercial off the shelf -software, requirements emphasis should be placed on identifying and documenting requirements from the functional areas which most probably require configuring or customizing the software to meet the organization's needs. This requires specific functional expertise from the procuring organization. In this scenario, thorough commercial negotiations and demonstrations are recommended to ensure that the product technology selected can in the end be configured or customized to meet the requirements and associated constraints with reasonable effort.

It is possible to certain extent compensate the mistakes and flaws in the early-stage requirements management process through careful design and selection of project management methodology. According to Kumar & Kumar (2011, p. 116), agile implementation approach provides more freedom over the traditional waterfall implementation approach in exploration around requirements during the project; to complement them, or to even scope out the vaguest requirements. However, execution of agile approach effectively – especially in large software projects – requires a competent and empowered project organization. Furthermore, it can only help to make the most out of the choices already made. Therefore, agile methodology is not a solution for obtaining the business benefits in situations where the actual business needs behind the software project are not fully clear (Chemuturi, 2013, p. 230).

2.3 Quality criteria for software project requirements

Requirement is a capability which a product or service should possess (Khan et al., 2013, p. 21). Requirement as a term is utilized to describe a set of desired characteristics and attributes within a particular product or a service. They can function as something that needs to be performed or act as an attribute that needs to be an integral part of an entity (Altable, 2015, p. 64). The goal of a requirement is to communicate the desired behaviour in a clear way to a diverse set of groups: users, customers, vendors, other stakeholders, and to the project and development teams (Heumann, 2003, p. 3). According to Halbleib (2004, p. 10) a requirement statement should generally answer ques-

tions “What must the project do?” or “What must the product do?” but almost never “How will the product do it?”. In other words, requirement statements should generally be unaffected by the analysis, design, and implementation methods and notations utilized in the implementation project.

High quality requirements specification should allow solution implementation and testing without returning to the analysis phase for additional clarifications. However, it is not simple to write a requirement so that it is both readable for people who work with it and that it contains all, and nothing but, the necessary information. It is typical that some requirement specifications are concise but lack the essential information and/or are inconsistent while others contain all the relevant information but are incomprehensible (Raczkowska-Gzowska & Walkowiak-Gall, 2023, p. 57–58). To overcome this challenge, Bigelow (2020) lists the below factors as most important quality criteria for requirements:

- Clear and understandable.
- Correct and complete.
- Consistent, not redundant.
- Unambiguous.
- Design-agnostic.
- Measurable and testable.
- Traceable.

Mongomery et al. (2022, p. 191) identify the same criteria in their survey except they also include *complexity*, *reusability,* and *relevancy*. However, these three additional criteria are scoped out from the theoretical framework of this study. *Complexity* is not an issue itself if the requirement is clear, understandable, and possible to implement in the given constraints. It may even be necessary in some domains. *Reusability* is not directly related to the project outcome and therefore not being relevant to the purpose of this study. *Relevancy*, although extremely important for the project outcome, is expected to be fulfilled inherently for the simplification reasons explained in Section 1.3.

Requirement being *clear and understandable* means that it is expressed concisely and with correct language. Such expressions and language should be utilized that nontechnical stakeholders can understand and verify the correctness of requirements (Robertson & Robertson, 2013, p. 21). Bigelow (2020) emphasizes plain language, free of jargon, which makes requirement documents easier to evaluate.

Correct and complete requirement does not miss any critical information. Criticality should be assessed from the point of view of the business needs (Bigelow, 2020). According to Robertson & Robertson (2013, p. 24) this does not mean that all possible requirements must be elicited before moving forward with the project since this can lead to initial scope creep. Lower-priority or new requirements can most probably be elicited and implemented later in further releases as part of continuous development of the implemented solution.

For requirement to be *consistent, not redundant*, it must be aligned with other requirements with regards to both the functionality and the terminology being utilized. If two requirements are in conflict, it is impossible to implement a solution that meets all requirements (Davis & Leffingwell, 1996, p. 8). Robertson & Robertson (2013, p. 387) suggest having a prioritization and conflict resolution processes in place in case conflicts in requirements between key stakeholders emerge.

Unambiguity means that there is no room for interpretation regarding a requirement. If a requirement has several interpretations, the probability of failing some expectations grows. Increasing the level of detail of a requirement statement generally reduces ambiguity (Davis & Leffingwell, 1996, p. 8). Chemuturi (2013, p. 212) emphasizes the importance of avoiding expressions like “easy to use” or “aesthetically appealing” since these types of expressions are difficult to interpret and implement.

Design-agnostic requirement is not dependent on how exactly a specific outcome is achieved, for example how a software technically works. According to Davis & Leffingwell (1996, p. 8), external behaviours from the perspective of

users or by other interfacing systems are still considered requirements regardless of their level of detail. On the other hand, when a requirement addresses subcomponents or their algorithms, it is not a requirement anymore – it is design information.

To be able to confirm that a specific requirement has been fulfilled, it must be *measurable and testable*. Halbleib (2006, p. 10) recommends avoidance of again such words as “most” and “some” and adjectives such as “quickly” since these will make the requirement inherently non-testable. If a finite set of tests cannot prove that a system has passed or failed to satisfy the requirement, the requirement should be rewritten or re-evaluated.

Finally, requirement should be *traceable* in a way that it can be linked all the way from the business needs to the actual implemented software code. Traceability helps in change management and in quality assurance since it ultimately proves that the software does what it is supposed to do. (Davis & Leffingwell, 1996, p. 11). When a project manager can verify that all approved business requirements are ultimately reflected in the software code, the project can be considered complete (Bigelow, 2020).

There is no universal answer on how to express different requirements in the exact right way. Robertson & Robertson (2013, p. 227) suggest deriving the notation from a business need scenario or business use case. This means answering to the question “What does the product have to do to complete this step in this scenario?" with a requirement expression. Bigelow (2020) formulates this method in the following format:

1. The subject is identified, i.e. [a project, user type, or software shall…]
2. The task or action is described, i.e. […do something to…]
3. The justification is provided for the action, i.e. [… reach an outcome.]

The above structure generally fits to all levels of requirements. However, it is critical to keep the expressions precise and concrete. Too vague expressions often leave too much room to interpret a sufficient outcome which might differ from the underlying intention of the requirement (Bigelow, 2020).

2.4 Results from the literature review

The literature review suggests there is strong positive correlation between effective requirements management and a successful outcome of a large software project. As Halbleib summarizes (2004, p. 8), effective requirements management helps to control quality, cost, organization, and schedule, thus substantially increasing the probability of a successful project. Correspondingly, Davis & Leffingwell (1996, p. 1) and Kumar & Kumar (2011, p. 110) conclude that software projects regularly fail because of problems with requirements management. Now, to link the concepts of this study together, a conceptual model is synthesized in the following Figure 1.

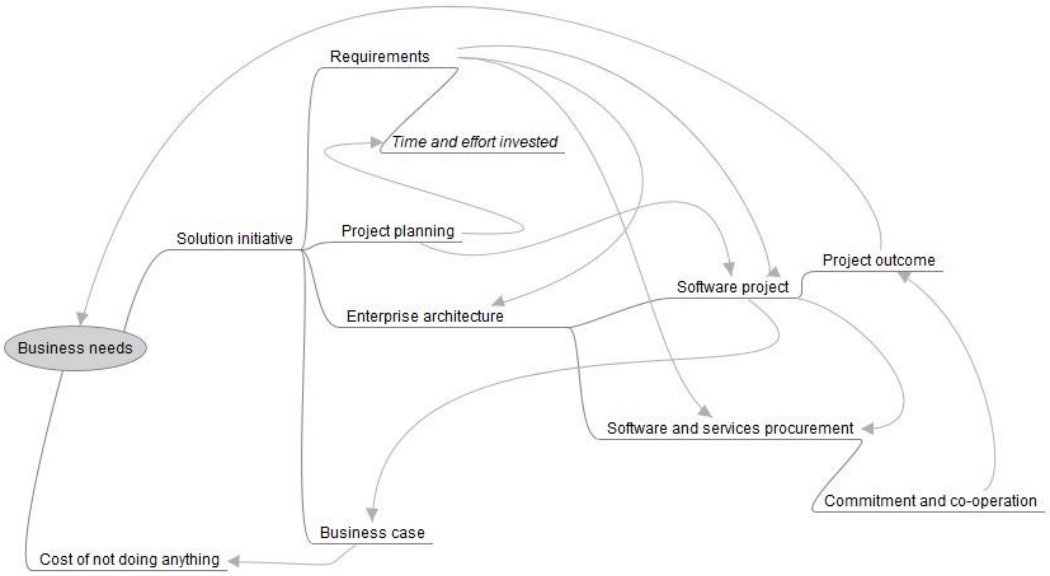

Figure 1. Conceptual linkages of the theoretical framework

Figure 1 illustrates how business needs drive new solution initiatives which then require careful planning, consideration, and preparation to increase chances of successful development and implementation. Requirements management, when executed as the theory suggests it should, is a time-consuming effort. Enterprise architecture should be considered carefully in the requirements management, in subsequent project planning, and in procurement. In parallel, requirements need to be utilized in procurement to reach common understanding and to formulate a shared vision and goals with vendors. When formulating a business case, organization also needs to consider the cost of

not doing anything, which is emerging from the ever-increasing competition and from a constantly changing business environment.

Based on the literature, requirements can be categorized into three levels, all derived from the underlying business needs. The requirements hierarchy is presented in the following figure 2.

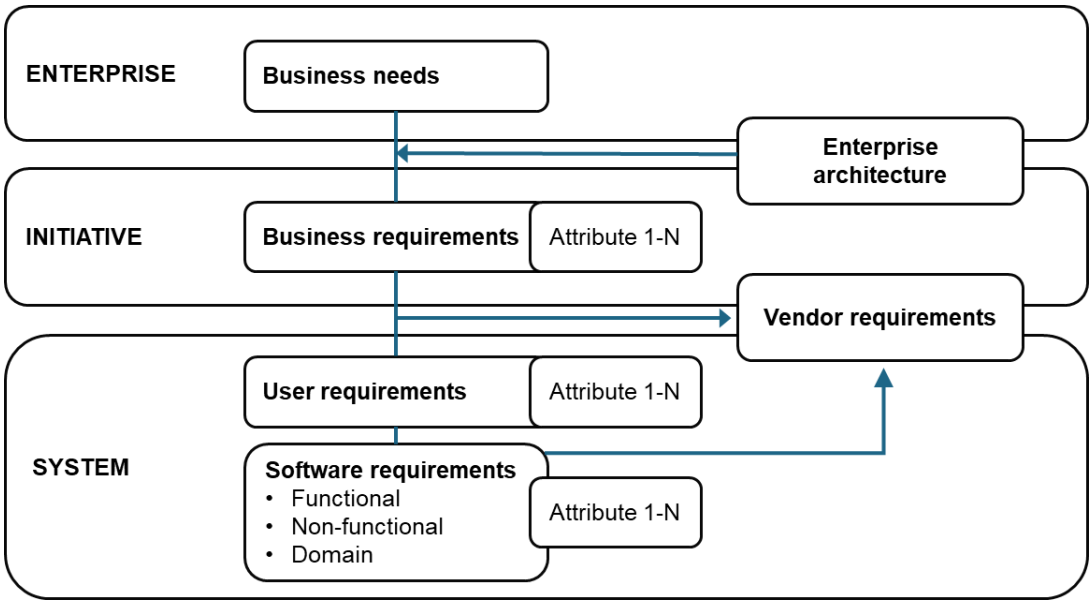

Figure 2. Requirements hierarchy

Business needs emerge from the ever-changing competitive environment and are not bound to any specific project or technology. *Business requirements* are the first level of actual requirement expressions, explaining why the project is necessary by describing what added value the solution brings to different stakeholders in a tangible and measurable way. *User requirements* are the second level of requirements which complement the business requirements by explaining who the users are, how they interact with the software, and what they expect and need. *Software requirements* are then the third level of requirements, explaining what the software does to fulfil the business and user requirements. However, software requirements should not express technical details on how the software does what it does. Software requirements can be further divided into functional, non-functional, and domain-specific requirements. Requirements can and should contain attributes and express constraints on what input and preconditions are necessary to reach a specific output.

A high-quality requirement meets the following criteria:

- Clear and understandable: utilization of concise language.
- Correct and complete: not missing any relevant information.
- Consistent, not redundant: no duplicates, conflicts, or differing terms.
- Unambiguous: no room for interpretation.
- Design-agnostic: focus on what and why, not how.
- Measurable and testable: have a clear “definition of done”.
- Traceable: linkage throughout the requirement hierarchy.

There is no universal answer on how to express different requirements in the exact right way, but literature provides an example structure through the following components of a story:

1. The subject is identified, i.e. [a project, user type, or software shall…]
2. The task or action is described, i.e. […do something to…]
3. The justification is provided for the action, i.e. [… reach an outcome.]

The above structure generally fits to all levels of requirements. However, it is critical to keep the expressions precise and concrete. Too vague expressions often leave too much room to interpret a sufficient outcome which might differ from the underlying intention of the requirement. In addition, constraints should always be carried along the requirements to manage stakeholder expectations. For example, a software might be perfectly capable of providing a specific task and outcome, but only if it has the proper master data management processes in place to ensure the required data foundation for operations.

There is extensive amount of research around effective requirements management in the execution of large software development projects. However, very little research seems to exist on how requirements should be elicited or managed effectively as part of project preparation to ensure the realization of business benefits. The empirical part of this study will investigate how different types of requirements elicitation and management approaches in the project preparation phase have affected project outcomes. Multiple-case study method will help to understand specific contexts and mechanisms of the relation-

ship between the early-stage requirements management and its consequent steps. With the selected approach, this study tries to contribute to the business/IT requirements management research in a practically relevant way.

3 RESEARCH METHODOLOGY

3.1 Research approach

When considering the most suitable research method, a holistic research strategy must be first formulated. Hirsjärvi et al. (2008, pp. 128–131) list three main categories of research strategies which are: 1) experimental study, 2) survey, and 3) case study. According to Saunders et al. (2009, p. 146), case study strategy is recommended when the researcher wants to gain a rich understanding of the context and processes of a specific phenomenon.

Case studies aim to either create new theory or alternatively validate existing theory through empirical research. Hypothesis can be first created based on the theory which the empirical results are then compared to. In the end, the theory itself can be modified through the results (Darke et al., 1998, pp. 275–276). According to Myers (1997, Section “Case Study Research”), case study as a research method suits very well to information systems research since information systems are usually studied in organizational/social rather than in technical context.

A case study is considered as a multiple-case study if it contains more than one research case. According to Yin (2014, pp. 56–57), analytical advantages can be accomplished this way compared to a single-case design, but attention must be given to selecting proper research cases and processing them in a uniform way. The following Figure 3 illustrates the systematic and replicated execution of a multiple-case study.

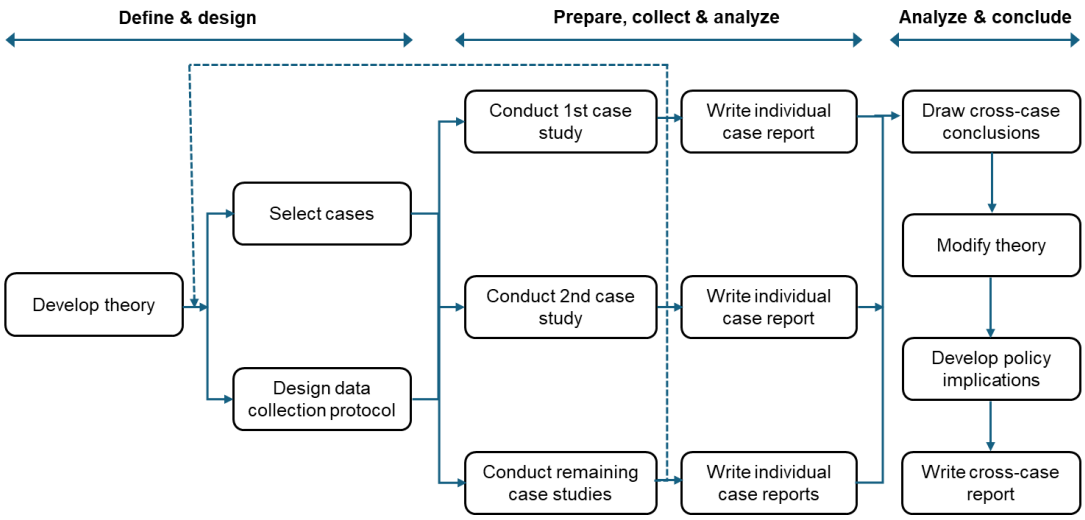

Figure 3. The process of conducting a multiple-case study (Yin, 2014, p. 60)

The strategy of this study is a multiple-case study with the intention of describing and understanding the phenomenon how requirements management impacts the outcome of a large software project. The study follows the structure presented in Figure 3. First, a hypothesis is created based on existing theory. The research data is then collected and the output from the data is compared to the theory to either prove or discard the hypothesis. This strategy can also be categorized to be an action-analytical research approach (Neilimo & Näsi, 1980, p. 67) since the goal is to draw a descriptive hypothesis from existing theory, test its validity empirically, and finally provide normative practical recommendations. According to Olkkonen (1994, pp. 72–74) this strategy is ideal for research problems which are difficult to structure, but where the researcher has a thorough understanding of the research problem. In action-analytical approach, the subjects of research are typically analysed through in-depth conversations, and very little exact methodological guidelines or norms are associated with the research method itself.

The research subjects selected for this study are eight professionals, each with over 10 years of experience in preparing and executing large IT-/software projects. Individual case reports of each research subject are first written and presented in Chapter 4 together with a cross-case analysis. Finally, conclusions and theoretical and managerial implications are presented in Chapter 5.

3.2 Collection and analysis of data

Method refers to a rule-based process which in science enables collection of specific data and solving a research problem (Hirsjärvi et al., 2008, p. 178). Regarding the selection of the right method for this study, both the availability of data and the purpose and scope of this study have been considered and evaluated. Eisenhardt (1989, pp. 534–535) states that interview is one valid method for understanding the dynamics and emerging patterns of a specific phenomenon. A semi-structured-/theme interview, which is a popular qualitative data collection method in business research, is the method selected for this study. In a semi-structured interview, the researcher formulates a structure and questions for the interviews, but interviewees answer the questions with their own words. The order of questions is also not fixed in advance. Effectiveness of this method is based on researcher's ability to guide an interview without controlling it entirely. It is also seen as a motivating method for interviewees (Koskinen et al., 2005, pp. 104–105).

According to Hirsjärvi & Hurme (2022, Chapter 7.1) and Saunders et al. (2009, pp. 489–491), although there are various ways to analyse the data from semi-structured interviews, only a few standardized techniques are recognized, and no technique is proven to be superior to another. Main characteristics of a qualitative analysis are:

- Observations made by the researcher of the phenomena emerging from the data, for example based on frequency, repetition, distribution, or characteristics.
- Classification of models from the observations.
- Maintaining the data in its original format.
- Inductively generalizing or creating new theory directly from the data, or deductively utilizing the data to validate the research hypotheses based on existing theory.

The following Figure 4 illustrates the spiral-like process of the analysis of qualitative interview data in an iterative way:

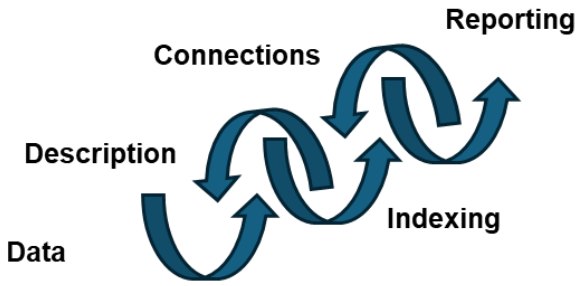

Figure 4. Iterative process of analysing interview data (Hirsjärvi & Hurme, 2022, Chapter 7.4)

The iterative process in Figure 4 was followed in this study. The semi-structured interviews were recorded with the consent of all interviewees. An applicable interview framework was first prepared (Appendix 1) after which the interviews were conducted. The most relevant parts of the interviews were then transcribed into textual format to help the analysis and documentation of findings. According to Hirsjärvi & Hurme (2022, Chapter 7.2.1), there is no exact guideline for the accuracy of the transcription. In this study, the transcription was done only for the parts relevant for this research, and all other conversations or expletives were not transcribed. The transcriptions were indexed to ease their interpretation and pinpoint the connections between the data and the theoretical framework. These connections and other findings were then compared between the research cases and the theoretical framework. Finally, a report of the findings was created in the form of final conclusions.

3.3 Reliability, validity, and operationalization

Reliability and validity are utilized as general metrics for assessing the credibility of scientific research. Reliability means that results of a study stay the same even when the study is repeated multiple times. In other words, reliability measures repetitiveness and accuracy of results. Validity then measures how well the results match to reality (Saunders et al., 2009, pp. 156–157). Validity can be further split into 1) construct validity, 2) internal validity, and 3) external validity. Construct validity shows whether the correct operational measures have been generated for the study. Internal validity then focuses on the internal

harmony and logic of the study. Finally, external validity implicates whether generalizations can be done from the study or not. (Yin, 2014, pp. 46–49).

According to Koskinen et al. (2005, pp. 254–256) the concepts of reliability and validity do not fit well into qualitative research. Reliability has a definite meaning only in quantitative analysis or in indexed polls. Validity, on the other hand, has a definite meaning mainly in experimental studies. Reliability and validity therefore represent conservative principles which, if utilized strictly, often lead to risk aversion. In the end, the goal of scientific research is not mere error elimination – it is the generation of new knowledge.

The reliability of this study is increased primarily through methodological choices. Multiple-case study increases the reliability because larger quantity of research subjects stabilizes the accuracy of the results, even if the quality of the interview content varies between interviews. In addition, all the research subjects are experienced professionals, motivated to participate in this research, with no conflicts of interest to lobby towards specific results. Correspondingly, the validity of this study is increased through the selection of semi-structured interview as the research method. Semi-structured interviews both provide in-depth knowledge of the research cases and address all relevant questions in every single interview. Finally, the accuracy of the analysis is improved through transcriptions and indexing of the interview recordings.

The internal validity of a study can also be improved through operationalization (Saunders et al., 2009, p.125) which in this study means that the research problem is logically broken down all the way into individual interview questions. This way, it is ensured that the study has really focused on the intended research problem. This operationalization process is illustrated in following Table 2.

Table 2. Operationalization of the study

Effective requirements management in preparation for large software projects				
Purpose	To identify and describe the most important factors in requirements elicitation and management process in project preparation phase for increasing the probability of a successful outcome of large software project.			
Research problems	To identify through existing theory the most important quality criteria for requirements expressions in large software project preparation.		To empirically study the impact of the identified requirements quality criteria on the success of large software projects.	
Research questions	What are the key characteristics of a well described requirement?	What is the reasonable scope and level of detail for describing requirements as part of the software project preparation?	Does the research data indicate that the identified requirements quality criteria affect the success of software projects?	Can other factors in project preparation be identified from the research data that contribute to the success of software projects?
Theory	Sections 2.2, 2.3, 2.4	Sections 2.1, 2.2, 2.3, 2.4	Sections 2.2, 2.3, 2.4	Sections 2.1, 2.2, 2.3
Empirical findings	Interview questions H1–H3, Q1–Q2, P1–P2	Interview questions M2–M3, Q1–Q2, P1–P2	Interview questions M1–M4, H1–H3, Q1–Q2, P1–P2	Interview questions M1–M4, H1–H3, Q1–Q2, P1–P2

The operationalization table 2 helps to understand the interlinkages between the sections of this study. The interview questions were derived from the theoretical framework in a way that possible correlations between the quality of requirements and the outcomes of the research cases could be shown. Opportunity was also provided for the interviewees to express other factors which they saw important for the context of this study. All the personal and privacy information, including names of persons and organizations, were replaced with only the unique identifier codes of the research cases and with the titles of the research subjects. In the end, this study can be objectively seen to pass the criteria of scientific research and to be compliant with the ethical ground rules of a good scientific practice.

4 EMPIRICAL FINDINGS

4.1 Presentation format of the interview data

To improve the readability and understandability of this study, a uniform presentation of the interview data must first be created. This also helps to ensure that all case studies are being processed in a uniform manner. The following figure 5 illustrates the presentation format of the data in this study.

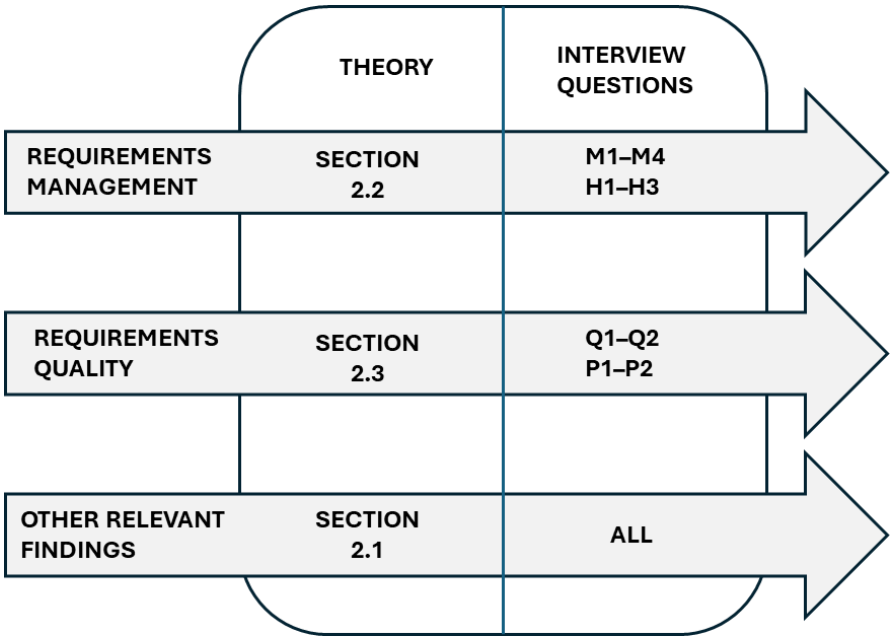

Figure 5. Presentation format of the interview data

The research questions provide a solid foundation for data analysis, but additional attention must be also paid to decoding the whole interview content in a coherent way. That is why three separate sub-sections presented as the arrows in figure 5 have been formulated. The sub-sections are linked in matrix to the theory and to the interview questions. Together they form a logical order of first describing the big picture – the project preparation and its related requirements elicitation and management activities – followed by the details of the quality and presentation format of the actual requirements. Finally, the separation of other relevant findings as an own sub-section allows answering the fourth research question through the same cross-case analysis as the other questions.

The potential interviewees were initially screened and contacted in December 2023 after the completion of the thesis implementation plan. The sample of final interviewees was then selected in January 2024 based on their experience, availability, and willingness to participate in this study. The interviews took place in April–May 2024 after the completion of the theoretical framework and the interview questions. In the end, the thesis report was finalized in June 2024 after the cross-case analysis of all the collected research data. The following Table 3 summarizes all the case studies and subjects.

Table 3. Summary of the case studies and subjects

CASE	INDUSTRY	SUBJECT
Alfa	Electrification/automation	Global Process Owner
Beta	Software and services	Senior Product Manager
Gamma	Machinery manufacturing	Manager, Data Platform & Analytics
Delta	Software and services	Director, Products and Technologies
Epsilon	Public sector	Program Manager
Zeta	Software and services	Chief Technology Officer
Eta	Automotive	Global Manager, Manufacturing IT
Theta	Software and services	Vice President, Operations

The interviewees in the above Table 3 currently represent altogether seven different organizations. The selected sample of case study subjects incorporates both customer and vendor as well as private and public sector viewpoints. The case descriptions are presented in the following Sections 4.2–4.9. Finally, a cross-case analysis is conducted in Section 4.10 by comparing the key findings of all the individual research cases.

4.2 Case Alfa

Subject Alfa is a Global Process Owner in an international electrification and automation corporation. He has over 10 years of experience in software imple-

mentations, from multiple roles, mainly in the supply chain management and execution functions.

Requirements management

Subject Alfa sees a positive correlation between investments in requirements management and a successful project outcome. The business needs behind software implementations have always been expressed to some degree but the needs don't seem to fully translate into how the subsequent projects have been initiated and scoped. According to Alfa, one of the main reasons for this is that the organizations responsible for business or mission execution want the project to solve their problems, but they seldom entirely understand the complexity of the reasons contributing to those problems. Another main reason is that solution initiatives are often driven by people who do not actually work in the business function in scope of the project and therefore do not fully understand the day-to-day reality. The result is that the business side is in the end unable or unwilling to change the way they work to enable the original business needs to be fulfilled by the new software. In other words, the constraints of the underlying requirements have not been fully appreciated.

> I find systematic and hierarchical requirements mapping especially important as means to estimate the probability of a project's success – what is really required from all levels to reach a successful outcome.

Alfa has never personally witnessed a thorough and systematic requirement mapping all the way from business needs to user and software requirements being executed as part of project preparation. The prerequisite for effective requirements management is a clear understanding and scoping of the problem that needs to be solved by the project, but this is typically done too vaguely or not done at all. In other words, although the business needs might be expressed in the beginning, the actual business requirements for the project are not elicited and documented. Instead, requirements are elicited bottom up from the users which leads to a pile of subjective opinions about the challenges in the "as is" -state but not to a holistic picture of the "to be" -state.

> Too often there are requirements elicited from the users which might be nicely categorized and prioritized but haven't been in any way mapped to the cornerstones of why the project has been initiated. For example, that these specific user requirements link to this specific business requirement and so on. And what more, further including the enterprise architecture aspect into the requirements – why the existing systems don't fulfil the requirements and what does it require to move from old to new.

What typically happens next is that management authorizes and funds the project even though they have not really seen any hard evidence how the business is in detail planning to take the new software into use. An implementation partner/vendor is then procured to deliver the solution. This creates an illusion that the responsibility has been moved from the customer to the vendor, but in real life it isn't because the customer always must manage the vendor through requirements of some form. As the project progresses further, and understanding of the reality increases, requirements are often changed and new requirements are added because the solid business requirements foundation is missing. Thus, the scope becomes a moving target, and complications will follow.

Requirements quality

Subject Alfa thinks that in the project preparation phase the most important requirements quality criterion is whether the requirement is fit for the project or not. In other words, that the requirement is 1) relevant to the project goals and 2) realistic in a way that the project has real means to fulfil it. The challenge is often the lack of consistency, meaning that requirements from top management are too high-level pipe dreams whereas user and software requirements are scattered, ambiguous and subjective opinions. Moreover, these two levels seldom meet or are being mapped and linked together hierarchically.

> High quality requirements should tell a clear story, not only about what we want to achieve, but also what does it require from us to achieve it. This should be then reviewed carefully with top management in the project preparation. For example, if we scope these specific components out to reach a cost target set by the top management, we will not reach the benefits they are expecting. So, are we ultimately ready to do what it takes, including changing the way we work, and invest the time and effort required for the software to work and produce the expected output? If not, we should not proceed with the initiative.

Alfa sees it almost impossible to have such a perfect requirements quality in the project preparation that any challenges wouldn't occur later. This is because there are always unknowns which are revealed only after more people are involved as the project progresses. But if the two quality criteria – relevancy and applicability – are met, other requirement details can most likely be iterated and clarified along the way without endangering the project outcome.

The exact format of the requirements is not so relevant if the proper understanding is translated into the execution. In one example project preparation, all relevant business processes were first visualized as value stream maps where all the challenges were shown and explained. But what was produced next was a list of Excel rows of user and software requirements without direct link to the original value stream maps. What would have served the project better would have been a storyline in a form of presentation where each identified challenge would have been analysed and broken down into how it will be solved by the project, what software components are required to achieve this, and what does it mean for the processes and for the ways of working. The required level of detail then depends on the context. A requirement can be expressed in very detailed level, but it must be expressed in a format that justifies the solution even with subjective criticism coming from the individual users.

Other relevant findings

The importance of data quality, constraints, and availability were considered essential for the software being able to produce the desired output. According to Alfa, the master data part and its complexity is often overlooked in projects.

> For example, a project delivers a new software platform to collaborate with supply chain partners and collect data for improved decision making. But sourcing function is not included in the project to negotiate with the partners of the requirements to provide the data through the new platform. Or you might buy a software based on its features, but you have not planned how the required master data is managed. It is not enough that you build a capability, but you must also push the capability into effective use. These type of partial project executions are very typical where the change management element is totally overlooked.

4.3 Case Beta

Subject Beta is a Senior Product Manager in a large software company. He has over 10 years of experience in software implementations in many industry domains, especially in retail, also in the role of a project manager.

Requirements management

Subject Beta sees a strong positive correlation between investments in requirements management and a successful project outcome, especially in large software projects. But despite some defects in requirements management process, it is still possible to reach a positive outcome in the end. It will take more time and thus cost more as additional work must be done in eliciting and documenting the requirements later during the execution. But if the underlying business needs are clear, probability of eventually reaching the goals is high. However, this absolutely requires that the product itself is good and fit to purpose.

Beta has never personally witnessed a thorough and systematic requirements mapping all the way from the business needs to software requirements done by the customer in advance. It is more common that high-level business needs – such as reducing product stock outs – are relatively clear, but the actual business requirements, including the target KPI's, are missing. Too often both customers and vendors use a lot of time brainstorming together the details of a specific functionality but don't discuss enough about *why* it is required: Who are the stakeholders and the end users, what are their specific business challenges, and what does the top management expect. Here, the industry domain expertise of the vendor plays a critical role in bridging the related gaps because a competent vendor with the aim to make the project successful can significantly help the customer in requirements elicitation and management.

> If we as a vendor don't fully understand the customer's goals and what we are expected to deliver, a red flag should immediately appear. Because if we as credible domain experts don't, who does?

According to Beta, another critical factor in requirements management, leading to a successful project outcome, is the quality of co-operation between all parties. Since requirements are never perfectly documented in advance, misunderstandings happen to some degree in every project. If both parties work together and take responsibility by admitting their mistakes openly, the damage and arguments can usually be repaired and settled effectively. However, if each party tends to blame the other side for placing or misinterpreting an imperfect requirement, the project is likely to become troubled very quickly.

Requirements quality

Beta emphasises mapping of all requirements – even loosely – to the business needs and to the desired outcomes of the project, in other words traceability, as the most important requirements quality factor. However, there are significant differences on which requirements should be expressed with high level of detail and which can be expressed in more general level. This is very domain specific and there is no simple way how to do this effectively. A vendor either knows or doesn't know by experience both to ask the right questions and to consider the various constraints of their provided solution. The customer on the other hand may or may not have the right professionals involved to identify and clarify those requirements that are not obvious to all vendors. Experience is the key but then again, all new things must be done for the first time by someone. In this type of experimentation project scenario, the mindset of partnership and fair risk and reward sharing is critical for success.

User stories are generally good tool to describe the most important requirements. In other words, 1) explaining the role of a subject, 2) what the subject wants to do, and 3) what is the desired outcome from that operation. But it is important to focus on cases which really contribute to a specific business goal and not to those emerging from old habits or from subjective opinions.

> Eventually it doesn't matter whether the requirements are in a written document, a visual presentation or whatever if it describes the problem to be solved. Because even though the goals might be clear as such, it

> might still be unknown why the customer hasn't reached those goals. But when you know the real problem, it is much easier to design the solution.

Other relevant findings

Beta has witnessed complex stakeholder interests and incentives affecting project outcomes. Chances of success are increased if 1) a vendor is rewarded for an effective and agile execution and 2) if the vendor genuinely cares about the success of their customer, even with the risk of losing some short-term financial gains. On the other hand, if a vendor seeks to execute non-value adding work and thus lengthen the project to make profit, it will have a negative impact. These types of contradicting interests might emerge in large software projects or programmes being delivered by multi-vendor consortiums.

The importance of data quality and data constraints always play a critical role in software being able to produce the desired outcome. It might be an unpleasant finding during the project that a critical functionality cannot be delivered or utilized because the required input data is missing entirely or is not on a right level of granularity. It is not enough to mention the data requirements once at some point in the project preparation but the data requirements and constraints need to be regularly reviewed. It must also be ensured that the right people responsible for the data are involved, available and committed to the project.

Finally, Beta emphasized including the right experts from key functions into the project preparation. It would be even more effective if a vendor with strong industry domain expertise could participate and contribute to the scoping of a planned project already in the sales phase by allocating their product experts.

> Unfortunately, it happens that only a handful of people have brainstormed together and decided to start a project without including the right people. Then it is found in the execution phase that in real life there are a lot of challenges and constraints that were not in any way known or were purely neglected in the project preparation. No need to tell that in these cases the starting point for a successful project is not very good.

4.4 Case Gamma

Subject Gamma is a Data Platform & Analytics Development Manager in a global machinery manufacturing company. He has over 15 years of experience in software implementations in supply chain management and business intelligence domains, also in the role of a project manager.

Requirements management

Subject Gamma sees a positive correlation between investments in requirements management and a successful project outcome. Early stage requirements management helps to identify how applicable the planned software solution really is and whether it is the right approach to the problem. Gamma has never witnessed a total failure in business needs fulfilment but there have been vast differences in how well projects have succeeded. If the right experts are present in preparing the requirements, the chances of success are increased. But it can also happen that requirements are prepared without really understanding the complex reality in which case risks of failure are increased.

Effective requirements management process should focus around identifying and prioritizing the most valuable requirements with regards to business benefits. Otherwise, there is a risk of extensive work without added value or an initiation of a project which cannot fully succeed in the end. The process should also contribute to understanding whether there exists real interest to transform the ways of working with the help of the new software. Especially in larger organizations, there can be different stakeholders pushing solution projects forward with the expectation that new technology will solve their challenges, but who don’t know well enough the reality of the day-to-day business. In worst case, a project can last for a long time with costs running even though there have been domain experts in the organization from the beginning who have clearly seen that the project will never reach a successful outcome.

> We can execute a major expensive technology project to make something transparent. But then after the effort, what changes does the organi-

> zation really do with this new transparency? And it may even be that the same transparency has been there already in a form of knowledge. So just with effective management and right incentives, the same result would have possibly been achievable with fraction of the time and cost.

Requirements quality

Gamma sees the single most important quality criterion being the requirement's real impact to the business, in other words relevance with regards to the business needs. Other factors mentioned are completeness, measurability and traceability. The requirements should also contribute to ensuring that the planned solution has a sufficient life cycle to cover the investment. High quality requirements are also grouped to enable specific sets of requirements being implemented in releasable phases. This is more related to agile implementation methodology where requirements can be further iterated along the way and feedback collected from the organization before going too far with a specific implementation, to ensure that the focus stays on the correct things.

The format of requirements doesn't need to be fixed, but typically requirements management software helps to keep the requirements and their mapping organized. According to Gamma, these type of software tools should be utilized in all large organizations. But in addition to the information managed in the requirements management system, there must also be a separate change management material package with key messages linked to the business requirements. In other words, answers to why the organization is pursuing what it is pursuing. This is to ensure the commitment of the business organization to the project. It is also important that the management of requirements is not outsourced to vendors, and that there is a continuous dialogue between the vendors and the implementing organization.

> What I see really helping in ensuring successful project outcome is a market dialogue between multiple vendors in the procurement phase of the project preparation. And by that, I mean really bringing the right domain experts from the buyer into that dialogue instead of just letting the procurement experts collect bids. Because, despite the constant time pressure, allocating for example one more month to do the homework can easily save a year in implementation which then contributes to major

cost savings. So, thorough preparation and market dialogue is definitely more effective than rushing into a project in the seeming time pressure.

Other relevant findings

Gamma has seen cases where there are business requirements to harmonize ways of working between business units even though the units are fundamentally different in how their business environment works. In other words, there is a thought process behind the business requirements that a new software helps to harmonize processes and thus brings benefits even though there is no real added value to the core business coming from the new software. So, it seems that business requirements are decided without really considering the underlying business needs. In these types of situations, systematic requirements management could bring transparency to what is the real problem the organization is trying to solve and why, or does the problem even exist.

4.5 Case Delta

Subject Delta is a Products and Technologies Director in a large software company. He has over 15 years of experience in software implementations in multiple expert and managerial roles, mainly in the finance sector.

Requirements management

Subject Delta sees limited correlation between investments in requirements management and a successful project outcome. Although requirements management is important, the real challenge in organizations is identifying the real business needs and executing projects based on them. Projects are often initiated on personal ambitions and too narrow understanding of the reality, main driver being typically a vague target of improving efficiency. But when all the complexities of the business environment become transparent during the project, there might not be real commitment to push the necessary change to really transform the way of working in the organization. One of the main reasons for this is that the right people knowing the details of the day-to-day reality are

often not present in the project preparation. Finding and including the right experts and questioning the initial assumptions, if necessary, is difficult and slow whilst there is nearly always time pressure to push the project forward to show that the organization is working hard to improve.

> No matter how well you prepare, there are always a lot of uncertainty and unknown factors in the project initiation. Therefore, I find it important that the organization is willing and able to iterate the requirements along the way as the project progresses. So, you have an idea in the beginning, but it will be further refined or might be even scoped out along the way.

According to Delta, especially in large software projects, it might be that the complexity of business reality on all levels of the organization is just so high that there is no single correct process to effectively manage requirements. Therefore, the emphasis should be on discovering the real business needs and converting them into such requirements that can be constantly reviewed and iterated. However, Delta has personally never seen that happening in a structured and effective way. Typically, software is procured through generic feature requirement lists without any real connection to the business process transformation. It seems that organizations are expecting benefits to emerge just by acquiring and implementing the technology without any need to really change anything in the day-to-day processes.

> I’ve seen cases where requirements management goes into way too detail way too early. Instead of business needs, you have 500 detailed requirements and an expert can immediately say that around 30 of those are relevant. But I’ve also seen cases where organizations just start doing something based on some vague idea without any requirements.

Requirements quality

Delta sees relevance being the single most important requirement quality criterion. If relevance is fulfilled, all the other criteria can be managed along the way through different stages of discovery and iteration. The relevance of the requirements in this case is directly related to understanding the business needs and subjecting all the following work to serve that purpose. Other important factors mentioned are correctness, completeness, measurability and

traceability. These are also tightly coupled with the relevance of the requirements. If the project has a relevant goal, it will be reached sooner or later.

> The quality of requirements and the business potential should come from really understanding the day-to-day work and processes. This could mean for example that the key people in the project are humble enough to disembark to learn what is really being done in the field. But too often the requirements emphasis is only on some irrelevant software features.

Delta doesn't see any specific requirements expression format being superior to another. A generic good model is first describing the business needs in some sort of concept document, having the right experts involved in the concept creation. Second, user stories that explain how the users contribute to fulfilling the needs should be created and linked to the business concept. If user requirements are elicited separately without any connection to the business needs, they most likely will turn out to resemble the features and logic of the legacy systems which hardly adds any value. Finally, software requirements should be written by the people who truly understand how the software really contributes to fulfilling the business needs and user stories.

> Involving the end users in requirements management work is important for change management purposes to make them feel appreciated. But the key input is really what they do day-to-day and why. There is generally no point in asking the end users to write any software requirements.

Other relevant findings

Delta brought up in the interview that organizations don't necessarily have a clear perspective to what they really want or need. Failed projects are sometimes required just to educate organizations to become more effective in their business needs discovery. Traditional project preparation models don't support this very well since they are built around planning, scope, schedule, and budget, with the assumption that all the answers already exist in plans.

> It seems that organizational culture is often based on acting busy, executing, and delivering something fast instead of stopping and thinking what is really needed. It is almost if our brains are wired to this, and projects then provide an easy way to show that we are doing something

> useful. And at the same time our brains try to avoid the really difficult thing which is thinking, innovating, and questioning the existing models.

Another important point was the need to have a clear business owner for the requirements and for the subsequent project. The owner needs to have the motivation and authority to drive the transformation forward, having the vision about the business problem and why it needs to be solved. If the owner is missing, requirements management will just be co-ordination work of a project manager in which case project failure is guaranteed.

> For example, management initiates a project with the target of maximizing automation. But then when they realize along the way that it actually means that some of their teams become redundant, they don't want the automation anymore. So, both the vision and the will are just not there.

4.6 Case Epsilon

Subject Epsilon is a Program Manager in a public sector organization. He has over 20 years of domestic and international experience in both large software project and infrastructure capability implementations.

Requirements management

Subject Epsilon sees a strong positive correlation between investments in requirements management, continuous risk management, and a successful project outcome. Early-stage requirements elicited top down in a logical and structured way create a solid foundation for the project and help steering it to a right direction. Requirements management should start from a capability concept which clearly identifies the business needs and business requirements: the "as is" vs. "to be" together with the justification of why the change is necessary. This phase shouldn't yet touch any specific technology. After the concept is approved by top management and key stakeholders, user and software requirements can be elicited. However, these should always be mapped to the higher-level concept and shouldn't contradict with it in any way.

Epsilon has witnessed a systematic requirements elicitation, mapping, and breakdown in a large infrastructure programme but never in a large software project. According to Epsilon, this is probably because infrastructure systems are typically solutions to more tangible and clear organizational needs which makes requirements management more intuitive compared to software. Software projects on the other hand can be initiated for reasons which do not necessarily relate to a direct business need. The driver could be for example a need to migrate multiple legacy software into one modern solution to decrease application management costs or the need of generating information for managerial decision support. Here, a project preparation can go wrong if the requirements elicitation is not instructed and conducted the right way.

> For example, in one specific case there was a clear business need to improve business process and application management efficiency, and to increase the information transparency for top management. But because it wasn't broken down into more specific business requirements by process domain, what followed was a project with isolated domains where all stakeholders and users ultimately were customizing a new software to work exactly like the so familiar legacy systems. The project grew too big, lost focus and eventually had to be fully restructured.

Requirements quality

Epsilon mentions unambiguity, consistency, and measurability as the most important quality criteria for individual requirements. Concept documents where multiple software requirements may be mapped to a single business requirement or use case then need to be especially clear and understandable. These concepts and use cases can be validated and further refined in software demos, provided by vendors, to increase common understanding and buy-in.

> In software projects, individual requirements don't create a comprehensive understanding. If a project manager receives a list of rows without any business concept or user stories and the rows express things like "better than old" or "has to be intuitive", it is impossible to start building or procuring a solution or to eventually validate the project's success.

Epsilon sees three key pillars for a high-quality requirements format. 1) The capability concept which is a comprehensive document explaining the priori-

tized business needs, business requirements, desired change and why the change is necessary. 2) Use cases and user stories which explain what the users need to do to fulfil the business requirements. 3) Software requirements, both functional and non-functional, which should be managed in a requirements management system to keep them in a right format and allow the traceability. Traceability is especially important tool in situations where an individual requirement cannot be fulfilled. The impact and criticality to the whole system can then be more easily analysed.

Finally, throughout all levels of requirement elicitation there should always be clear instructions for the participants why a specific piece of requirements documentation is being produced and how it will be used. This way it is ensured that the participants only focus on the relevant requirements with regards to the project goals and not waste any time in non-value-adding details.

Other relevant findings

Epsilon emphasized the mission critical importance of top management support and commitment for building new capabilities with software in an ever-digitalizing world. Traditionally, software has been something that's been the business of a separate ICT department, not the top management. But if the leadership and steering of building software capabilities is left directly to large quantity of middle management stakeholders, the chances of a successful outcome will be dramatically decreased.

> Ultimately success boils down to leadership throughout the organization. Because even if top management can squeeze the business needs and business requirements for the project, there is always the middle management layer below who are able to saw off the chair legs in a way that the chair will eventually crash.

This challenge is less likely to occur in infrastructure capability projects where the ownership of the business challenge and its potential solution is clear and there often exists more specific instructions and gate models for project preparation. It could also be that the ability to continuously modify and develop soft-

ware for multiple stakeholders makes it more difficult to commit to a specific design as where in infrastructure a waterfall type of design and implementation is often enough to fulfil a specific business need. Large software projects typically last a long time, which creates a lot of opportunities along the way to question their necessity unless the foundation is solid.

4.7 Case Zeta

Subject Zeta is a Chief Technology Officer in a medium-sized software company. He has over 20 years of experience in software implementations in many industry domains, for example in healthcare, retail, tax administration and HR.

Requirements management

Subject Zeta sees a positive correlation between systematic requirements management and the level of professionalism of the implementation preparation. The role of requirements management is to link the business needs and software requirements together and map them into measurable outcomes. This usually happens when the initiative is being prepared by a business owner who knows both the “as is” and “to be”. If there is only a vague idea of business needs without any mapped requirements and a project has been established on that foundation, it is a clear red flag for all the stakeholders. One typical example of this is when a software company has managed to sell a solution idea to a prospective customer without any hard business case or through a business case that is calculated only by the software vendor.

> I’m sure every software company tries to sell their value proposition as the most relevant solution to any customer’s challenges. But how often can you really link it to measurable outcomes and are those even defined? I’ve seen a lot of variation here. A company can for example be drifting in a crisis and then looking for solution to break out of it. Here, it can be very tempting to seek a technological medicine to reduce costs, but the risk is high that the project will just make things worse.

According to Zeta, the starting point of effective requirements management should be to spend enough time analysing the real business requirements: 1)

What is the real problem to be solved and 2) how it should be solved. Is the required solution only technical or does it include thousands of people changing the way they work? If a specific solution has worked somewhere, it doesn't necessarily mean that the same solution works in another organization because there can be significant differences in change management capabilities. In the end, if no one is using the solution, no benefits are being realized. But if the business requirements analysis is done well, then it can be straightforward to elicit and map the subsequent software requirements. However, Zeta has seldom seen the whole analysis and elicitation done very systematically.

> I've seen different types of feature requirements lists which can be nicely categorized in multiple levels. But their mapping to the real underlying business requirements… it's surprising how seldom this happens in real life. It is much more typical that there is a lot of energy and enthusiasm to just do something to move fast and thus seem effective. But then the complex reality sinks in at some point and the enthusiasm fades.

Requirements quality

Zeta sees the most important requirement quality criterion being that the value potential of the requirement is being clearly expressed. This means expressing both why the requirement is there and what is its priority in the big picture. This allows iteration and adding more detail later when things become clearer, without losing the original purpose. Understandability of the requirement to all the key stakeholders is also important. Other things mentioned are measurability and traceability. The format of requirements is not so decisive if they are expressed in a way that stakeholders on all levels understand them.

> In expressing requirements effectively, one can take advantage of the various formats which resonate best on each level, enabling the evaluation whether this is realistic or not. For example, for top management you could express requirements and impact through tools like balanced scorecard and such but then on implementation level you break it down into something which serves the project management or technical experts better.

The format and extensivity of user requirements depend on how large the transformation is. If there are only minor adjustments to current processes, it

makes sense to document how the user's life can be improved in the current reality. But if an organization is really transforming the way they work and there is a clear vision and ownership of how to do it, users simply must be trained to the new way – whatever it may be.

According to Zeta, the necessary extensivity and level of detail of all requirements in large software projects is very difficult to define exactly. To avoid listing thousands of requirements, one option is to seek a similar enough high performing reference organization which has implemented similar solutions, and simply go towards the same direction. That way the organization doesn't have to start from scratch. However, it is still necessary to elicit those domain requirements which are mission critical to the realization of the most important business benefits because very few organizations have the capability to procure and implement solutions in a pure agile manner, based only on abstract business goals.

Other relevant findings

According to Zeta, one fundamental problem of requirements management is that organizations' view of requirements is often based on current reality and therefore limited. For example, there can be a lot of requirements initially listed but then there could be a solution which entirely removes the need for those requirements. In this scenario, the requirements didn't really link to the business needs or alternatively the expressed needs were not real. This is why nowadays it is typical that organizations are looking for software as a service -solutions to also learn business best practices. Traditional customer driven requirements management simply doesn't fit very well to this type of setup where a strong business-technical combination is required to achieve real change.

> I think that some IT-organizations are born as support functions for business organizations which means that even though they are leading the preparation process, they simply don't have the necessary transformation mandate. But then, the business organizations are not professional procurement or change management organizations and might end up buying something totally unnecessary to solve a problem that they have created themselves. So, you really need both sides working together.

Another finding was that there can be surprisingly high amount of political theatre in organizations and solution initiatives can serve as one instrument to orchestrate this. Procuring new technology with the hope of fast benefits is an easy way to gamble for personal glory in the organization. And sometimes even the lack of business benefits doesn't stop this from happening.

> I know a case where in executive board, a presumably implemented solution was being presented as huge success. But in reality, the solution *didn't actually exist*. So, bubbles detached from reality can definitely be formed when there are many organizational levels involved.

4.8 Case Eta

Subject Eta is a Global IT and System Manager in a large automotive supplier. He has over 20 years of international and domestic experience in software implementations in many industry domains, such as pulp & paper, telecommunications, and automotive, also in the role of a project manager.

Requirements management

Subject Eta sees strong positive correlation between successful project outcome and thorough and systematic requirements management done in project preparation. This phase is so critical that the best way to carry it out would be through an independent analysis and specification project. This type of approach helps to ensure that sufficient time will be spent to understand the business problem and what is required to solve it before moving forward with the project. The time invested here will save a lot of trouble and rework later.

According to Eta, the role of requirements management is to link the business needs and software requirements together and map them into a justified business case. This requires a business owner who knows both the "as is" and "to be", and a competent IT-organization to identify and manage the technical constraints. The key really is to have all the right people involved.

> One major problem in organizations is that simply the wrong people are leading the preparation with unclear roles and responsibilities. The cornerstones of all successful software projects that I've seen have been that enough time is being invested in requirements elicitation and evaluation, and that contribution is coming from the right professionals.

After the business needs are identified and analysed, they can be broken down into prioritized business requirements expressing what the project is expected to deliver and in what priority order. Different subsets of software and user requirements can then be elicited and mapped to these business requirements. Here it is critical that, in addition to the original business owner, there is also a concept owner type of person leading the requirements management process and ensuring the continuous mutual understanding between the business owner, business process experts and technical software experts.

> Very often something that is considered as a software project is really a business transformation project. For example, a manufacturing execution system might fully change how the factory should work. Software are just tools – they are not silver bullets or something you just install and achieve the results that way. The transformation element is really critical for the benefit realization and it requires hard work from the business side.

Requirements quality

Eta mentions relevancy, unambiguity, consistency, completeness, traceability and measurability as the most important quality criteria for individual requirements. There should always be a punctual and descriptive header and then a quantifiable "definition of done" to enable agreement that the requirement has been fulfilled. They cannot be high-level concepts but must be hierarchically mapped to granular enough level instead. Business process experts should also let go of trying to design details of user interface and other software components but instead focus on expressing what needs to be achieved and why.

> Listening to users is important but it cannot be a single defining factor. Requirements need to be hierarchical, and the business case should drive them in the end. Because it might be that the user is requesting something which seems like a minor thing but – if customized – could change the whole data model of the software and corrupt it entirely. But it is still

> important to understand the reasons behind the user requirements because there might be something there that is not visible or evident at all.

According to Eta, high quality requirements should start from a clear and understandable business concept document which can seem almost like a manual. The focus in the concept should be to first explain the justified business case and goals and then the refined and unambiguous business requirements for the project. Process and information flows should be also described already at this stage. This should roughly be the end of business side's responsibility, shifting the lead to IT organization and technical experts to elicit the technical requirements and constraints. Finally, a mutual review should be done about how realistic it is to succeed in the project, especially considering the required business transformation.

> The right balance between requirement accuracy and elimination of non-value-added work can be found when the business requirement is unambiguous. After that the competent technical experts know what is required and how to describe it as the essential subsets of software requirements. But you really must understand where the business requirement turns into system requirements and then have both parties focused on their own side of the coin, not forcing the other party to do stupid things.

Other relevant findings

Eta emphasized that there are often unrealistic expectations towards software projects which are to some extent emerging from the difficulty to understand the principle of delivering software in releases throughout the life cycle. For example, different industrial professionals understand that investments and deliveries of material assets require the whole system to be delivered fully in one piece before it can be made operational. But with software it can be difficult to understand that the initial production release is not yet supposed to be anything else than a foundation. This misunderstanding might lead to scope creep.

> Delivery in releases seems to be very difficult to understand for many, especially with global software systems. That we are investing a lot of time and money to deliver a platform which is only a foundation on which to build the business processes and use cases. But that the foundation takes 80%-90% of the time and budget and it needs to be released as a

> base product first instead of just keep doing more within the project. So, the project is just an initial delivery and the real improvement happens then during production as part of continuous development roadmap.

One interesting finding was also that there can be fundamental flaws in lessons learned and retrospective analyses executed after a project which prevent an organization from really learning and improving. If the project preparation has been based on non-existing business case or the scope of the project has changed along the way, it means that the steering group of the project has failed to deliver the right guidance and decisions. But since the steering group of large software projects is most often staffed with senior management, it is too embarrassing to highlight the bad decisions as lessons learned. Therefore, the real reasons behind the failures are not written in retrospective analyses which then prevents these analyses from generating any new knowledge.

4.9 Case Theta

Subject Theta is an Operations Vice President in a large software company. He has over 15 years of experience in software sales, presales and implementations in many industry domains, also in the role of a project manager.

Requirements management

Subject Theta sees a positive correlation between systematic requirements management done in the project preparation and a successful project outcome. However, there is not necessarily one right approach for success. In large organizations or in highly regulated environments a thorough and systematic way helps in disciplined scope management but the trade-off is the time it requires. Then there might be other type of cases where it is generally accepted to explore and experiment solutions based only on more high-level business needs. Whether the time spent in early-stage requirements work is gained back later in more effective project execution depends on the relevance of the work. It is essential that whatever the approach is, it is mutually agreed and

tightly managed by both the customer and the vendor and that there is transparency and constant open communication towards all stakeholders.

> In large organizations and projects, the high-level requirements management and all the dependencies must be well disciplined so that the goals are set realistically, and the scope doesn't explode. But then you can bring some lean elements into it depending how well you break the goals down into packages where you then have the flexibility to decide how the requirements are precisely fulfilled. But this approach must be agreed.

Theta has seen both well executed and poorly executed requirements elicitation and management in project preparation. In well executed cases, there has been a clear top down -approach all the way from business needs to software requirements. But then there have been cases where it has been unclear even what the real goal is which typically mirrors the general management culture of these organizations. Effective requirements management process should focus around first understanding the strategy-driven business needs and then identifying and prioritizing the most essential business requirements that the project is expected to deliver. However, there will most likely be some business requirements that simply cannot be identified yet in the project preparation although they later prove to be critical for the project's success. In these situations, it is essential to have clear roles and responsibilities to be able to solve the unexpected situations as they emerge without paralyzing the whole project. The roles and responsibilities are also important from the change management perspective and the necessary transformation element should be embedded already in the business requirements.

> For example, if there is a generally accepted idea, at least on some level, that the project is building a global process template… and there is even a named owner responsible for that. But then it appears that this owner's change authority is valid only in the business unit this person represents. It is just impossible for the project to achieve its global goals in that setup.

Requirements quality

Theta didn't highlight any specific quality criteria to be specifically important. The general quality criteria appearing in the theory can be accepted as impor-

tant, but it is difficult to follow them meticulously without falling into an infinite loop of specifications. Universal formats such as user stories can help to ensure the quality of a specific expression. Still, the business reality is not simple.

> You can go wrong by being too nice and saying yes to a customer requirement in time pressure to show progress, underestimating the effort it requires by promising it for Friday… and then after four Fridays you sit in a meeting with a red face when the work is still not ready. The requirements are important and you must do your best but the reality always requires honest and open communication about the constraints too.

According to Theta, the format, extensivity and necessary level of detail of all requirements in large software projects is very difficult to define exactly. It is typical that there have been consultants involved in the preparation who have already generated extensive documentation. But what is essential is that there exists a high-level business concept describing the goals and to some degree also the most important KPI targets. Then there should be a list of the most important use cases related to the business concept which represent the business requirements that the project is expected to solve and deliver. Software requirements can then be mapped to these use cases, but it must be generally accepted that especially the software requirements together with some of the hidden use cases cannot be perfectly elicited yet in the preparation phase. The customer organization should focus first independently on creating at least a preliminary version of the use cases which can then be reviewed and iterated further together with software vendor's product experts before any contract is signed. When the right level of confidence has been achieved, for example with the help of demos and reference visits, the project can move forward and the final details of the requirements together with how they will be exactly met can be left to the project to decide.

> There might be situations where the focus is to describe how the software should work in some very specific way even though the near same result could be achieved in much simpler but different way. Then, a lot of time can be wasted on debating on this nitty gritty detail instead of just reviewing what the business requirement and goal really was. Involving the users in this discussion is a difficult question. Naturally trying to implement something where the users have not been involved in any way is not the most kosher idea. Maybe the best approach is to identify some of

> the more dynamic users and involve them as key people in the project. They can then also help as change agents and improve communication, for example, to help explaining to other users why something cannot be done exactly the way they are requesting.

Other relevant findings

One interesting finding was that, according to Theta, software project initiatives might serve as instruments to try to show that transformation is being strived in an organization even though the stakeholders are fully aware that there are unresolved power struggles and change resistance in the background which seriously decreases the probability of success. In other words, the political contradictions are not dealt first to help the project. Instead, the project is left in the ungrateful position of struggling to make the change happen without the necessary top management support.

> I think this is simply wrong. The project should be seen as an instrument to prove that the real desired “to be” works – not to be something where the unpleasant change management responsibilities are just outsourced. The political battle should be fought elsewhere, paving the way for the project to work as a showcase and motivation for the whole business that the improvement efforts are worth it and that results can be achieved.

Theta also mentioned that success requires naturally strong partnership attitude between customer and vendor but also that both parties are actively managing their own organizational stakeholders and their expectations, thus keeping the project focused on the essential. Generally, those experts – for example concept owners – who understand both the business process logic and the technical aspects are extremely important key people for the project.

> One clear characteristic of a successful project is that there is an effective duo of project managers from both customer and vendor who act as gatekeepers. Customer’s project manager must naturally be demanding towards the vendor but also towards his own organization to shoot down crazy ideas. And similarly, the vendor’s project manager must know when to do what the customer asks and when to fight relentlessly to prevent an execution of a requested work order which could lead to a slippery slope. I would describe it as mutual humility – to not just push one’s own agenda.

4.10 Cross-case analysis

In this section, a cross-case analysis is done by comparing the findings of the individual cases to each other. The same structural narrative of the individual case description is utilized. In the end of the cross-case analysis, a visualization of the key questions for organization to consider is presented. A final synthesis of the cross-case analysis, which ultimately provides the answers to the set research questions, is then presented in Chapter “Conclusions”, in Section 5.1.

Requirements management

All the research subjects see at least some correlation between the outcome of a large software project and how its early-stage requirements elicitation and management is conducted. Everyone fully agree that identifying the real business needs is the most important starting point which builds the foundation for the entire project preparation. However, there seems to be a lot of variation how well organizations succeed in this, and one research subject even pointed out that it is not even realistic to expect that organizations can proactively understand their fundamental business needs in the existing complex reality.

Even though the subjects agree on the importance of business needs identification, there is some variation how the requirements management should be conducted, particularly regarding the allocation of time, thoroughness and level of detail of the requirements elicitation. This seems to depend strongly on the domain, business environment and the management culture of the organization. In more regulated environments and in tightly managed organizations, the whole requirements management process seems to be more systematic, and the organizations often have experts who are capable of leading the work effectively. In these environments, the process is more thorough and takes more time, but the project scope is also managed better and the outcome is more predictable. Then there are other types of domains and organizations where it is generally more typical to initiate projects based only on the business needs or high-level requirements. In these environments, projects can also be suc-

cessful if there is a strong partnership and agreement with the customer and the vendor that the project is an explorative one and that the detailed requirements will be elicited together along the way. But trouble will emerge if the customer expects that a software vendor will solve all customer's problems by just doing everything the customer says and demand during the project.

All the research subjects see that the identified business needs should optimally be broken down into business requirements which would describe what the project is expected to deliver and why and what is the value potential and priority of each requirement. This then formulates the business case and the scope for the project and helps to identify whether the software project is the right approach or should the business requirements be fulfilled in some other way. If a software is decided to be sought, the customer should already have at least the most important business requirements identified before going into detailed negotiations with software vendors. In optimal situation, a competent software vendor can then help the customer to refine these requirements further before a contract is signed if the right experts from both organizations are included in the sales process. However, it is critical that a procuring organization is critical towards the generic promises of software vendors and does not accept their value propositions blindly. Instead, a thorough negotiation process must be conducted to ensure that all the constraints of the business requirements can be really managed together and that the customer is fully aware of the transformation the new software requires to obtain the business benefits.

Some variation emerges on how thoroughly the software requirements should be elicited in the project preparation phase. This is particularly domain specific and dependent on how well the customer understands what they really need. All subjects agree that all the software requirements – no matter when they are identified – must be mapped as subsets to the business requirements and that there must be a mutually agreed model on how to deal with new requirements emerging during the project execution. There seems to be no single correct answer on how to manage software requirements in overall but the following factors can be identified to clearly help: 1) business-technical expertise of both the customer and the vendor, 2) configurability and quality of the vendor's soft-

ware, 3) competent steering and project management to reject bad requirements, and 4) stakeholders, business process experts and users focusing on *what* needs to be achieved and *why* – not *how* the software should work. Regarding the elicitation of user requirements, all subjects agreed that its main purpose is change management – to make the users feel included and listened to. But the critical success factor is to really understand what they do and why.

The data seems to also provide some clear requirements management patterns which should be avoided to maximize the probability of a successful project outcome. One wrong way seems to be to start eliciting software and feature requirements bottom up in the very beginning without any business requirements because this most likely leads into large amount of non-value adding work as well as rework in later stages of the project. Another wrong way seems to be to conduct the procurement of a software by just going through plain software feature lists and comparing vendors based on the features. Finally, it also seems to be wrong to let users write user and software requirements themselves because this will most probably lead to unnecessary customization to match legacy features as well as to less than maximum utilization of the new software's business improvement potential.

Requirements quality

All the research subjects see *relevance* being by far the most important quality criterion for any need or requirement. The reason for this seems to be that every subject has at least some experiences on situations where requirements have been written without any direct link to real business needs or business requirements. From the quality criteria presented in the theoretical framework, measurability is mentioned directly by half of the subjects and can be seen as the second most important criterion. All the other criteria are mentioned too in some context. One subject mentions applicability in context with the relevance which indicates that some organizations might strive towards relevant solutions as such but for some reason they might still not be realistically applicable in that specific organization. Some of the subjects mention that requirements

quality criteria are generally good principles but the complex business reality makes it extremely difficult to follow them systematically.

The case studies don't provide a definite single answer on what is the correct scope and level of detail regarding how needs and requirements should be expressed when preparing for large software projects. Some of the subjects rely more on detailed analysis and thorough elicitation early on where others see professionally managed exploration through trusted partnership being more practical approach. The best approach seems to depend heavily on the business domain and circumstances. Still, the case studies indicate that one well working way seems to be to first spend time creating a concept document which clearly describes the strategic business needs together with the measurable and prioritized business requirements of what the project is expected to deliver and *why*. The business requirements can also be further illustrated with process and information flowcharts. Then, if an implementation project is decided to be initiated, the most important business requirements should be broken down into software requirements and into the most essential user stories by the experts who best know the "as is" and "to be". All research subjects emphasize the importance of having the right people with the right expertise involved in this work.

Other relevant findings

In addition to the direct perceptions on requirements elicitation and management, four distinctive yet interlinked themes can be identified from the data which clearly affect the software project preparation and have an impact on the project outcome. These are 1) business transformation, 2) business ownership, 3) business-technical expertise and 4) organizational politics.

Most subjects emphasize that large software projects must be viewed as business transformations where the new software often fundamentally changes the way the users should work and how the organization should operate. Based on the subjects, it seems that organizations claim they seek to change but when the change finally needs to be executed, they find various excuses

not to do so. Master data management and automation are some good examples to illustrate this. To obtain the benefits, the new software might require new level of process discipline and new resources but at the same time it might make some of the existing processes unnecessary. This seems to sometimes come as a surprise for many organizations which suggests that these aspects have not been considered carefully enough in the project preparation before rushing into the software procurement and implementation.

Multiple subjects mention the importance of clear ownership to push forward the required transformation and change management efforts. A strong business owner who is ready, willing, and credible enough throughout the organization to drive the necessary business transformation enabled by the new technology seems to be one of the most critical persons both in the preparation and in the execution phase. The business owner should have a clear vision of the required transformation, supported by the top management, but he/she should also understand how the “as is” works and why it works that way, to ensure that the transformation vision is ultimately valid and doable.

In addition to the business owner, subjects address that right key experts who know the day-to-day reality well enough must be involved already in the project preparation. In optimal situation, these experts should be able to talk directly to the software vendor’s product experts already in the negotiation phase to ensure that the product is fit to purpose. Also, the business and IT organizations both have important roles in ensuring the potential project’s success. Business knows their day-to-day processes and pain points, but they are not transformation, project, or procurement professionals. IT on the other hand often lacks the credibility within the organization and might be overrun by the business or might push for a hyped technology without any real business need. Thus, these two organizations should continuously work together, listen to each other and respect each other’s professional contributions. In addition, business-technical expert roles such as concept owners seem to be highly important for ensuring that business and IT ultimately understand each other.

Finally, despite all the above, the probability of a successful project outcome can be effectively decreased with organizational politics. Several subjects point out that software or technology projects might be used either as work-force for change management efforts or as instruments to strive for personal glory. When software projects are initiated on personal ambitions, too narrow understanding of reality and without careful change evaluation and communi-cation in advance, the probability is high that they will meet significant change resistance in multiple organizational levels. When resources are limited, there are often several competing initiatives fighting for the same resources and those who's projects have been rejected may not provide unconditional sup-port for the project that has been put forward. This political dimension can be seen to generate both static and motion friction to project initiatives and thus links directly to the related themes of business transformation and ownership.

Key questions for organizations to consider in project preparation

The individual research cases had clear similarities which can be ultimately summarized into key questions organizations should consider carefully as part of their software project preparation. These questions are presented in the following Figure 6. The questions are placed into the same hierarchical frame-work illustrated in the end of the literature review, in Section 2.4. This helps both to better connect the theory into the empirical findings and to improve the practical applicability of this study in organizations.

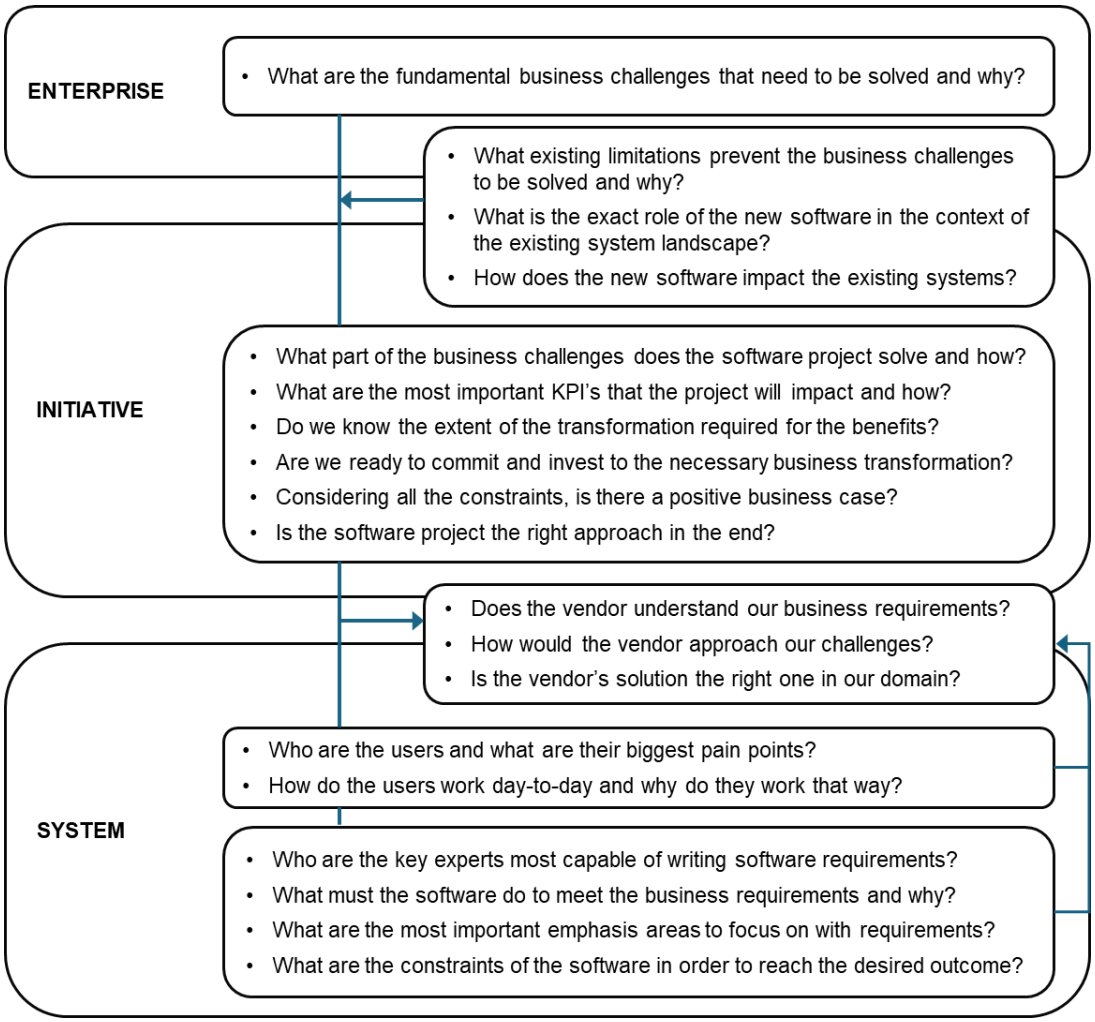

Figure 6. Key questions for organizations to consider in project preparation

Although the questions in the above figure 6 don't contain all the necessary factors to consider when preparing for a project, they may still work as a high-level checklist to ensure that at least the most fundamental elements of a successful project are in place before the project itself is initiated.

5 CONCLUSIONS

5.1 Synthesis of the cross-case analysis

The fundamental purpose of this study has been to identify how requirements elicitation and management should be effectively conducted in organizations

when preparing for large software projects and how it affects the project outcome. The study was structured around two research problems, each containing two research questions. These research questions can now be fully answered through a synthesis of the previous Chapter's cross-case -analysis.

The first research problem was to identify through existing theory the most important quality criteria for requirements expressions in large software project preparation. The problem was split into the following two research questions:

1. What are the key characteristics of a well described requirement?
2. What is the reasonable scope and level of detail for describing requirements as part of the software project preparation?

The literature review suggests a clear hierarchical division and categorization between a) business needs, b) business requirements, c) user requirements and d) software requirements. These should all be linked throughout the project life cycle although they don't have to be viewed together all the time. Each need or requirement should be 1) clear and understandable, 2) correct and complete, 3) consistent, not redundant, 4) unambiguous, 5) design-agnostic, 6) measurable, testable and 7) traceable. The empirical evidence didn't contradict with the theory in any way but clearly suggested that *relevance* should really be the most important quality criterion of any need or requirement. The literature review did also identify relevance as one quality criterion, but it was not highlighted because the simplified assumption in this study was that the reasons behind projects and requirements are generally valid. However, based on the empirical evidence it seems that this may not always be the case.

The literature review and the empirical evidence didn't provide a definite single answer on what is the correct scope and level of detail regarding how needs and requirements should be expressed when preparing for large software projects. This seems to depend heavily on the business domain and circumstances. However, one well working way seems to be to first create a concept document which clearly describes both the business needs and the measurable and prioritized business requirements of what the project is expected to

deliver and why. This should be the main emphasis in the preparation phase. Then, if an implementation project is decided to be initiated, the most important business requirements should be broken down into software requirements and from the most essential parts also into user stories by the experts who best know the “as is” and “to be”. The whole requirements library should ideally express a combination of managerial and transformation vision, business process understanding and technical knowledge. The wrong way seems to be to start listing software, user and feature requirements bottom up in the very beginning only based on vague business needs because this most likely leads into large amount of non-value adding work as well as rework in later stages of the project. A project can go even further wrong if these bottom up requirements are blindly considered as mandatory requirements for software vendors.

The second research problem was to empirically study the impact of the identified requirements quality criteria on the success of large software projects. The problem was spilt into the following two research questions:

3. Does the research data indicate that the identified requirements quality criteria affect the success of software projects?
4. Can other factors in project preparation be identified from the research data that contribute the success of software projects?

There seems to be positive correlation between successful project outcome and thorough and systematic requirements elicitation and management in the project preparation phase. Focusing on requirements early forces the organization to analyse and evaluate the real business needs behind the software initiative and review critically whether the software investment is the right approach for the organization. This then helps the project to focus on the essentials of value generation. However, the project can still go wrong in the execution due to scope creep resulting from poor project management or due to lack of credible ownership. On the other hand, projects can also succeed even if the early-stage requirements elicitation and management is executed poorly but they will most probably take longer and cost more money. It seems that due to complex reality, sometimes organizations just cannot identify the right

approach in the beginning but they might ultimately succeed in discovering it and committing to it during the project execution. However, this requires strong ownership, real management commitment and business-technical expertise.

The empirical evidence also suggests some other key factors that clearly contribute to the successful outcome of large software projects. As the theory already suggested, the challenges with software projects seem to be rather organizational and social than technical. Having a strong business owner who is ready, willing, and credible enough throughout the organization to drive the necessary business transformation enabled by the new technology is highly important. In addition, the project needs to be staffed with the right key experts who really know how the organization's day-to-day processes work, and they need to be involved already in the project preparation. The business and IT organizations both have a critical role in ensuring project's success, and they should continuously work together, listen to each other and respect each other's professional contributions. Finally, organizational politics and power struggles can be seen to generate both static and motion friction to project initiatives. These political contradictions should be dealt with before a software project is initiated to ensure that there exists maximal organizational support for the required business transformation.

5.2 Theoretical contribution and managerial implications

This study clearly affirms the existing theory regarding effective requirements elicitation and management in the preparation of large software projects. The awareness of requirements on all levels – enterprise, initiative and system – is important for project success, and their expressions need to be fit to purpose. The current theory seems to place a lot of emphasis on how requirements should be expressed for maximum utilization efficiency for the subsequent work phases. This includes a lot of tools and techniques to ensure the quality of requirement expressions. However, based on this study the real challenge often seems to be the real relevance of the requirements. Because even the

best tools and techniques don’t add much value if the requirements are focused on the wrong things. This should be highlighted more in current theory.

The managerial objective of this study has been to provide insight on where to set focus in requirements elicitation and management when preparing for large software projects, to maximize added value. Organizations should invest enough time to identify and analyse the real business needs behind project initiatives – even despite the constant feeling of hurry to be seemingly productive. This investment in thorough analysis both helps to ensure that the software project is really the right approach and if it is, also helps to keep the initiated project on track. Measurable business requirements should be derived from the business needs to answer how the project itself helps to fulfil them. These should be first processed to a level that they are fully unambiguous and then synthesized into a solution concept. All the subsequent user and software requirements should be based on and mapped to this concept by the right key experts who really know the day-to-day business and technical aspects in the organization. However, business process experts should let go of trying to design details of user interface and other software components but instead focus on expressing what needs to be achieved and why. In addition, instead of letting the users write requirements bottom up, it is much more value-adding to focus on just listening to them and understanding how they currently work and why and then just list the essential user requirements as user stories.

Technology can only bring benefits if it removes a current limitation. But if this removed limitation is not exploited in the way the organization operates, the benefits will not be obtained. Large software projects must be viewed as business transformations and the functional change element must be in the centre of the planning and execution. Based on this study, it seems that organizations claim they seek to change but when the change finally needs to be executed, they find excuses not to do so. Thus, it is essential that this transformation element is seriously considered in the project preparation. Questions such as “Are we willing to let go and replace a whole team with the new automation capability?” or “Are we willing to invest in new resources in master data management to ensure that the solution works with its full potential?” must be

evaluated and answered. All political contradictions should be dealt in advance to ensure that there exists maximal support for the required business transformation. In addition, a separate pre-project for the analysis and specification of the business case and for the requirements elicitation, management, and evaluation should be considered before deciding to implement the new software. And if there is a positive decision to proceed, a strong ownership is then required to drive the change and to mitigate the impact of hidden agendas, and to finally ensure that the organization truly learns from the project – whatever the outcome may be.

5.3 Limitations and potential future research areas

Even though the research subjects together form a data sample of eight experienced professionals, several industry domains and multiple different organizations and organizational levels, there are still fundamental limitations in this study. Firstly, qualitative research method only yields subjective opinions from the research subjects. Secondly, instead of analysing and concluding based on quantitative methods, the interpretation of the research data and its subsequent conclusions are only based on the researcher's own subjective view of the reality. However, in scientific research and especially in case studies, methodological choices and trade-offs must be made because it is very difficult and time consuming to create a research setup which would provide both extensive qualitative interview data and reliable quantitative test results.

There are some potential future research areas that can be derived from this study. One major finding in this study has been that technological investment initiatives such as software projects are often thought as silver bullets to challenges which could be solved with much simpler – although not necessarily easier – approaches. Furthermore, when striving towards technological solutions, organizations still tend to not utilize the technology up to its full potential. Instead, they often settle to customize the software to accommodate their old processes. It would be important to better understand why this happens.

This study has been a qualitative experience-based overview of both preparation of various software projects and of their subsequent outcomes. In the right circumstances a researcher could study the preparation of a single large software project and how the business needs there have been elicited and conceptualized, and how it all translates into project guiding requirements on different levels or does it? The project could be then observed throughout its life cycle to gain a deeper understanding of the organizational dynamics in the early-stage requirements elicitation and management and in the subsequent project outcome.

Based on this study, all the theory to avoid mistakes and to prepare for software projects effectively exists – yet so many organizations fail to do so. All the observations above would most likely provide good opportunities for future organizational research, especially in the field of industrial and technology management where humans are easily presumed to be rational actors even though the reality seems to be something very different.

5.4 Final reflections

The inspiration for this study and its theme has emerged from my personal experiences of working with business-to-business software for more than 15 years. During those years I have had surprisingly similar experiences as the research subjects which in the end was my personal hypothesis for this study too. However, both the theory and the interviews gave me fresh new insights as well, and especially the thorough conversations with the interviewees were extremely rewarding both personally and professionally.

In the end I believe the process of witing this thesis has enriched my thinking of software as means to improve the competitive advantage of organizations, as well as of organizational dynamics in general. I hope this thesis contributes to the work of those who strive to reach the full potential of business-technology. The world is steadily moving to a direction which makes the themes discussed in this thesis very current today, but even more current tomorrow.

REFERENCES

Alami, A. (2016). Why Do Information Technology Projects Fail? Procedia Computer Science, 100, 62–71. https://doi.org/10.1016/j.procs.2016.09.124

Altalbe, A. (2015). Software Requirements Management. International Journal of Advanced Research in Artificial Intelligence, 4(4), 64–68. http://dx.doi.org/10.14569/IJARAI.2015.040410

Bigelow, S. J. (December 9, 2020). What are the types of requirements in software engineering? TechTarget. https://www.techtarget.com/searchsoftwarequality/answer/What-are-requirements-types

Bjarnason, E., Wnuk, K. & Regnell, B. (2011). Requirements Are Slipping Through the Gaps - A Case Study on Causes & Effects of Communication Gaps in Large-Scale Software Development. IEEE 19th International Requirements Engineering Conference Research Paper, 37–46. http://dx.doi.org/10.1109/RE.2011.6051639

Budzier, A. & Flyvbjerg, B. (2015). Why do projects fail? Project Magazine. https://www.researchgate.net/publication/291945330_Why_Do_Projects_Fail

Chemuturi, M. (2013). Requirements Engineering and Management for Software Development Projects. Springer.

Claus, C., Freund, M. & Kneuper, R. (1999). Implementing Systematic Requirements Management in a Large Software Development Programme. Transport-, Informatik- und Logistik-Consulting (TLC) GmbH. https://www.semanticscholar.org/paper/Implementing-Systematic-Requirements-Management-in-Claus-Freund/6ea0d3d3e9ca14162ef423765bb57733e9fd5ee2

Damian, D., Chisan, J., Vaidyanathasamy, L. & Pal, Y. (2005). Requirements Engineering and Downstream Software Development: Findings from a Case Study. Empirical Software Engineering, 10(3), 255–283. http://dx.doi.org/10.1007/s10664-005-1288-4

Darke, P., Graeme, S. & Broadbent, M. (1998). Successfully completing case study research: combining rigour, relevance and pragmatism. Information systems journal, 8(4), 273–289. https://doi.org/10.1046/j.1365-2575.1998.00040.x

Davis, A. M. & Leffingwell, D. A. (1996). Using Requirements Management to Speed Delivery of Higher Quality Applications. Rational Software, 1–14. https://www.researchgate.net/publication/2385516_Using_Requirements_Management_to_Speed_Delivery_of_Higher_Quality_Applications

Dumitriu, D. & Mirona, A-M. P. (2020). Enterprise Architecture Framework Design in IT Management. Procedia Manufacturing, 46, 932–940. https://doi.org/10.1016/j.promfg.2020.05.011

Eigner, A. & Stary, C. (2023). The Role of Internet-of-Things for Service Transformation. SAGE Open, 13(1), 1–21. https://doi.org/10.1177/21582440231159281

Eisenhardt, K. M. (1989). Building theories from case-study research. The Academy of Management Review, 14(4), 532–550. https://doi.org/10.2307/258557

Finkelstein, A., Spanoudakis, G. & Ryan, M. (1996). Software Package Requirements & Procurement. Software Specification and Design 8th International Conference Research Paper, 1–7. http://dx.doi.org/10.1109/IWSSD.1996.501156

Gorkhali, A. & Xu, L. D. (2017). Enterprise Architecture: A Literature Review. Journal of Industrial Integration and Management, 2(2), 1–50. https://doi.org/10.1142/S2424862217500099

Halbleib, H. (2004). Requirements management. Information Systems Management, 21(1), 8–14. http://dx.doi.org/10.1201/1078/43877.21.1.20041201/78982.2

Heumann, J. (2003). The Five Levels of Requirements Management Maturity. Rational Software, 1–9. https://docplayer.net/19957100-The-five-levels-of-requirements-management-maturity.html

Hirsjärvi, S. & Hurme, H. (2022). Tutkimushaastattelu. Teemahaastattelun teoria ja käytäntö (2nd edition). Gaudeamus. https://www.ellibslibrary.com/

Hirsjärvi, S., Remes, P., Sajavaara, P. & Sinivuori, E. (2008). Tutki ja kirjoita (13th edition). Tammi.

Jastram, M. & Kara, A. (October 18, 2016). Modeling Requirements with Constraints. Requirements Engineering Magazine. https://re-magazine.ireb.org/articles/modeling-requirements-with-constraints

Kailio, A. (2023, October 10). ERP-uudistus ajoi kaupungin vararikkoon – järjestelmä pantiin tuotantoon vakavista varoituksista välittämättä. Tivi. https://www.tivi.fi/uutiset/erp-uudistus-ajoi-kaupungin-vararikkoon-jarjestelma-pantiin-tuotantoon-vakavasta-varoituksesta-valittamatta/5a04aadd-de6f-4f27-aac1-9499b0b573b9

Khan, M. N. A., Khalid, M. & Haq, S. (2013). Review of Requirements Management Issues in Software Development. International Journal of Modern Education and Computer Science, 1, 21–27. http://dx.doi.org/10.5815/ijmecs.2013.01.03

Koskinen, I., Alasuutari, P. & Pelkonen, T. (2005). Laadulliset menetelmät kauppatieteissä. Vastapaino.

Kumar, S. A. & Kumar, T. A. (2011). Study the impact of requirements management characteristics in global software development projects: an onthology based approach. International Journal of Software Engineering & Applications, 2(4), 107–125. https://doi.org/10.5121/ijsea.2011.2410

Lai, R. & Ali, N. (2013). A Requirements Management Method for Global Software Development. Advances in Information Sciences (AIS), 1(1), 38–58. http://dx.doi.org/10.4156/ais.vol1.issue1.3

Mieritz, L. (2012). Survey Shows Why Projects Fail. Gartner Inc. https://www.gartner.com/en/documents/2034616

Montgomery, L., Fucci, D., Bouraffa, A., Scholz, L & Maalej, W. (2022). Empirical research on requirements quality: a systematic mapping study. Requirements Engineering, 27(1), 183–209. https://doi.org/10.1007/s00766-021-00367-z

Myers, M. (1997). Qualitative Research in Information Systems. MIS Quarterly, 21(2). http://dx.doi.org/10.2307/249422

Neilimo, K. & Näsi, J. (1980). Nomoteettinen tutkimusote ja suomalaisen yrityksen taloustiede. University of Tampere.

Olkkonen, T. (1994). Johdatus teollisuustalouden tutkimustyöhön (2nd edition). Helsinki University of Technology.

Project Management Institute. (2017). Success rates rise: Transforming the high cost of low performance. Pulse of the Profession, 9th Global Project Management Survery. https://www.pmi.org/-/media/pmi/documents/public/pdf/learning/thought-leadership/pulse/pulse-of-the-profession-2017.pdf

Raczkowska-Gzowska, K. & Walkowiak-Gall, A. (2023). What Should a Good Software Requirements Specification Include? Results of a Survey. Foundations of Computing and Decision Sciences, 48(1), 57–81. http://dx.doi.org/10.2478/fcds-2023-0004

Raddats, C., Kowalkowski, C., Benedettini, O., Burton, J., & Gebauer, H. (2019). Servitization: A contemporary thematic review of four major research streams. Industrial Marketing Management, 83, 207–223. https://doi.org/10.1016/j.indmarman.2019.03.015

Robertson, S. & Robertson, J. (2013). Mastering the Requirements Process. Getting Requirements Right (3rd edition). Pearson.

Saunders, M., Lewis, P. & Thornhill, A. (2009). Research methods for business students (5th edition). FT Prentice Hall.

Sima, K. S. (2022). Initiating a Project, the right way. LIGS University Research Paper, 1–20. https://www.researchgate.net/publication/362176085_Initiating_a_Project_the_right_way

Takala S., Kuokkanen, K. & Moilanen, K. (2023, April 12). Selvitys: "Puutteellinen johtaminen" ajoi kaaokseen. Helsingin Sanomat. https://www.hs.fi/kaupunki/art-2000009513554.html

Tvete, B. (1999). Introducing efficient requirements management. Institute of Electrical and Electronics Engineers, Proceedings. Tenth International Workshop on Database and Expert Systems Applications, 370-375. https://doi.org/10.1109/DEXA.1999.795195

Yin, R. K. (2014). Case study research: Design and Methods (5th edition). Sage Publications.

Ylä-Anttila, A., Kailio, A. (2023, September 7). Miljoonakaupunki vararikossa – taustalla katastrofaalinen it-hanke. Tivi. https://www.tivi.fi/uutiset/miljoonakaupunki-vararikossa-taustalla-katastrofaalinen-it-hanke/477922a0-69a4-4923-b829-d95cd138357e

APPENDIX 1: INTERVIEW QUESTIONS

Context (C)

C1: Please briefly describe your background, current position and experience in the context of preparation and execution of large software projects.

Requirements management (M)

M1: When comparing successful and less successful software projects, what type of differences have you witnessed in project execution being based on real business needs?

M2: How do you see the role of requirements management in a software project preparation in the context of a successful project outcome?

M3: How would you describe an effective requirements management process in software project preparation?

M4: What do you think are the main reasons why large IT-/software projects fail to deliver according to the original expectations?

Requirements hierarchy (H)

H1: What type of breakdown of business needs into requirements have you witnessed in the context of a successful software project outcome?

H2: How do you see organizations performing in mapping their executed software project deliverables into the original business needs and thereby creating a proof of a successful project outcome?

H3: How do you see the mapping between original business needs, business and user requirements and software requirements affecting a software project outcome?

Requirements quality (Q)

Q1: What do you think are the most important requirements quality criteria in software projects to ensure a successful project outcome?

Q2: What type of correlation have you witnessed between high quality requirements in a software project preparation, a disciplined project execution and a successful project outcome?

Requirements presentation (P)

P1: What type of format do you think is the best in documenting and expressing requirements in a software project preparation to maximize the contribution to a successful project outcome?

P2: What level of requirements detail in a software project preparation do you think would be contributing most to the successful outcome of the subsequent project?