Vol. 13, No. 2, 71-89
doi:10.17708/DRMJ.2024.v13n02a05

Dynamic Relationships Management Journal

REQUIREMENTS CHANGE MANAGEMENT: A CASE STUDY OF AN ENTERPRISE SYSTEM IMPLEMENTATION PROJECT

Laura Fink
GEA College, Ljubljana, Slovenia
laura.fink@gea-college.si

Ajda Fošner
GEA College, Ljubljana, Slovenia
ajda.fosner@gea-college.si

Andrej Dobrovoljc
GEA College, Ljubljana, Slovenia
andrej.dobrovoljc@gmail.com

Tomaž Poznič
GEA College, Ljubljana, Slovenia
tomaz.poznic1@gmail.com

Abstract

The main theme of this article is the collection, preparation, and improvement of user requirements. The research relies on a case study of a complex ERP project from an international organisation. We proceed from the fact that the development of an individual ERP solution often goes beyond just a single project. Often, the development of the ERP solution continues even after the completion of the individual project. In particular, the article addresses the user requirements and their quality with regard to the requirements engineering process and change management. We discuss the characteristics of adaptive (e.g., agile) and predictive approaches, as well as how they affect the quality of user requirements. We emphasise that it is important for team members to be aware of the substantial impact that well-defined user requirements have on the success of the project. Although we argue that the number of change requests should not be taken as an indicator of project success, we discuss the factors that influence the number of changes during the project and the additional work after the project is completed. We identified two strategies that address the excessive number of requests for changes and additional work. The project team should give more attention to the preparation of high-quality requirements and, most of all, to enhancing their formalised testing and verification processes. Managing changes, especially changes in user requirements, is challenging and critical to project success. Companies should constantly optimise requirements analysis based on lessons learned from previous projects and ensure that past lessons are applied company-wide in future projects.

Keywords: *Change Management, User Requirements, Requirements Analysis, Requirements Engineering, ERP Project Success, Enterprise System*

1 INTRODUCTION

Agile methodologies have spread in software development with the aim of improving the capabilities of software development teams since 2001. Agile methodologies, along with other adaptive methodologies (PMI, 2021), when compared to predictive methodologies, are supposed to enable both quick responses to changes and new demands and ensure that the implemented system does not become rapidly outdated or fail to be implemented in reality (Gonzalez-Barahona et al., 2017). Agile stands for the readiness to embrace change throughout a project. In an agile setting, not only requirements but also the end product, service, or what is considered acceptable release may change (PMI, 2021; 2017).

The predictive approach, which is diametrically opposed to adaptive practices, obviously fueled the expansion of adaptive practices. The predictive, also known as traditional, approach emphasises the importance of thorough planning in early project phases, with nearly any changes in requirements and functionalities in later stages, while adaptive approaches, along with the agile approach, foresee changes in requirements throughout the project and rely on flexible plans and an almost immediate response to changes (IPMA, 2018; PMI, 2021, 2017). Successful projects, particularly in software development, have popularised agile approaches to the point where they are applied without much consideration, even in projects that are less suitable for such approaches than software development projects. Therefore, it is vital to emphasize that the optimal approach needs to be selected with careful consideration of factors such as the characteristics of the product, service, or result; the project; and the organization (PMI, 2021; Fink, 2017). Since it is rare to find a project that is perfectly suited for either a fully predictive or fully adaptive approach, many projects necessitate the application of hybrid approaches. Hybrid approaches provide the opportunity to apply the unique best-fitting combination (PMI, 2021; Wysocki, 2019) of predictive and adaptive approaches to the project at hand.

Businesses invest heavily in the renovation of business processes to adhere not only to increased efficiency but also to the quality of their procedures and final solutions. The context of this research includes two fundamentally different, or even opposing, business cultures. The pharmaceutical industry (Scherer, 2000), characterised by carefully planned long-term drug development, strict control, and rigorous regulatory requirements for the approval of products, differs substantially from the software development industry (Tukel and Rom, 1998; Damian and Zowghi, 2003), awash with new technology, pressured by rapid development, short product life cycles, continuous improvements, and constant updates. Fundamentally different business environments (Iivari and Huisman, 2007) inevitably constitute differences in operations, procedures, and working styles.

The context mentioned is the source of fresh views and perspectives on the central theme of this research, which focuses on change management in developing enterprise systems and enterprise resource planning (ERP) implementation projects. According to Copola Azenha et al. (2021, p. 90), the literature lacks practically oriented evidence that could enrich the discourse regarding hybrid project management approaches fitted to "distinct organisational cultures, specific processes, customer contractual requirements, and project specificities." Further, Theunissen et al. (2022) explain that minimal requirements' documentation, which might arise from implementing agile methodology, does not always contribute to the success of software development. They suggest that the quantity and quality of requirements' documentation should be increased.

In our study, we are interested in what activities could contribute to improving the processes of user requirements analysis and, in particular, the user requirements' change management process in ERP initiatives. The purpose of this study is to investigate how changes in requirements affect project success, as well as to identify the appropriate design of business processes and actions that facilitate rather than restrict the development of a software solution in the best way possible. More concretely, we address not only the questions regarding the choice of project methodology in the context of partnerships, which are based on fundamentally different business cultures, but we also focus on the quality of user requirements and question how constant modifications of user requirements compared to more fixed requirements influence the course of action. Finally, we also discuss the ever-present issue of factors that

influence the number of additional improvements required after implementation, especially in connection with the precise definition of user requirements at the beginning of implementation.

The challenge of achieving the best balance (Ramesh et al., 2010; Rasheed et al., 2021; Mockus et al., 2003) between thorough planning early on in the project and additions to initial planning, for example, at the start of each iteration, is a pivotal point that requires attention in each project and context at hand. Specifically, thorough planning in the early project phases strives to address requirements holistically, prevent major unnecessary changes later in the project, and prevent successive requests for changes after the project's completion. On the other hand, incorporating later additions to the initial planning allows for increased flexibility and prompt response to stakeholder feedback, requirement changes, and emerging market trends. This balancing, along with team coordination (Bick et al., 2017), can bring long-term benefits for ERP implementation success.

2 THEORETICAL BACKGROUND

Implementation of ERP solutions is inevitably linked to challenges in the requirements engineering process. One could hardly find a software development project where requirements were not an issue. According to Sawyer et al. (1997), some of the ever-present major challenges in requirements engineering include inconsistent and incomplete requirements, requirements not reflecting the actual user need, the expensive introduction of any changes after requirements have been agreed upon, and misunderstandings of requirements between team members. The challenges mentioned mostly relate to a traditional point of view, which emphasises the importance of solidly defining the foundations in the early stages of the project, to the extent that the project results allow.

The scientific discourse in which flexibility becomes the cornerstone is based on the well-known adage that "the only constant is change," or, in other words, "changes are constant." Thereafter, scientific discourse on the efficiency of software development projects emphasised the need for the coexistence of both predictive and adaptive (e.g., agile) approaches, which assume flexibility in the foundations of the project throughout the entire course of the project. The challenges in requirements engineering, in particular the trade-off between precisely defining requirements early on in the software development process and using agile methods that pre-assume requirements to change in later project phases, tackle not only the scientific community but also communities of practice such as RESG (n.d.), IREB (n.d.), and INCOSE (n.d.), as well as project management associations such as IPMA and PMI. The present study is the continuation of our previous research on the importance of user requirements for the success of ERP implementation (Fink et al., 2024), but apart from the previous study, it addresses entirely other research questions focusing on changes in requirements.

We further divide the review of relevant studies into several parts. We begin with some basic characteristics and concepts in ERP initiative management and continue to discuss the differences between adaptive (e.g., agile) and predictive project management approaches. Finally, we include a review of previous studies that looked at the requirements engineering process and how to handle changes to user requirements.

2.1 ERP system implementation projects

Organisations strive to increase their efficiency by organising their work into projects that have a clear scope, deadline, and budget. ERP projects, in comparison to other types of projects, require high investments and are usually quite risky. The common goal of ERP systems is to optimise and modernise business processes. ERP projects can include existing comprehensive software solutions on the market, open-source programme solutions, the development of a new comprehensive software solution, or a combination of these. It is not uncommon that the development of a solution goes beyond just one project since ERP system development is usually not limited to a single initial investment in the ERP system but is an ongoing endeavour aimed at improving business processes. Often, the outcomes of the previous ERP projects include the new requirements that form the foundations for the next ERP development project.

As with other types of projects, ERP projects should meet their goals within a limited duration and with limited resources. The success of the project can be judged according to various factors, which do not only include the achievement of the quality of the implementation, the deadline, and the forecast, but rather the success is evaluated on the basis of factors such as customer satisfaction, user satisfaction, frequency of use of the final solution, and last but not least, the benefit of the project is also evaluated from the impact it has on operations, efficiency, and, last but not least, on the company's strategy and long-term development (Fink, 2017). Among numerous success factors on software development projects (Kronbichler et al., 2009; Sudhakar, 2012), proper planning, appropriate change management mechanisms, and efficient coordination importantly contribute to the successful delivery of software solutions.

The optimal project life-cycle, organisation, and project management are often tied to the characteristics and the level of complexity (San Cristóbal et al., 2018) of the final solution, the type of individual project tasks (Baccarini, 1996; Dasović et al., 2020), and the uncertainty regarding requirements (PMI, 2017). The project's complexity is primarily determined by the project's value, the number of stakeholders and team members, the project duration, and the project's technological innovativeness and uniqueness. ERP projects are usually complex and risky. The development of a single solution is usually ongoing and surpasses a single project. For that reason, it is of utmost importance that the processes are designed in such a way that the lessons learned can be applied to follow-up projects (Shanks, 2000). In that sense, knowledge management (Mirić et al., 2020) plays an important role.

ERP projects, whether performed based on predictive, adaptive, or hybrid methodologies, include phases such as requirement analysis, conceptual design, code development, verification and testing, and installation (Kronbichler et al., 2009; Falkowski et al., 1998). Nevertheless, there are several ways that ERP projects are implemented. Matende and Ogao (2013, p. 522), for example, suggest that requirement analysis comes after "system configuration, customisation, data capture, conversion, and rollout." Aloini et al. (2007), in comparison, suggest that strategic planning, requirements analysis, and selection of software are performed during the conceptual phase; software deployment, integration, testing, and stabilisation are performed during the implementation phase; and maintenance, upgrading, new release management, and evolution are performed during the post-implementation phase.

2.2 Adaptive, predictive and hybrid project management approaches

Takeuchi and Nonaka (1986), who first named Scrum, nowadays one of the most popular hybrid approaches (PMI, 2021; 2017), described decades ago that successful companies manage their development processes differently. From that point on, the discourse on how project and process methodologies contribute to efficiency flourished among the scientific community. Many authors, among them PMI (2021; 2017), IMPA (2018), Royce (2009, 2011), and Wysocki, R. K. (2011), contributed to these discussions. It is not a coincidence that agile methodologies quickly expanded and further developed in none other industry but in the software development and product development industries.

Agile approaches, which belong to a group of adaptive approaches that use iterative and incremental approaches (PMI, 2021), are characterized by frequent stakeholder feedback. The adaptive approaches, which fundamentally differ from the predictive, also known as traditional approaches, merely focus on responding to needs as they arise and reacting rapidly to changes in requirements as they occur. They are preferred when requirements are highly uncertain (PMI, 2021) and when it is necessary to embrace and integrate changes due to feedback, new market developments, or new technology developments.

These approaches are characterised by their shorter life-cycle phases (Royce, 1987; Al-Saqqa et al., 2020), iterative planning, and evolving requirements compared to traditional comprehensive upfront planning (PMI, 2017).

In an agile environment, many activities, for example, development and testing, are performed concurrently, whereas in a traditional setting, testing is usually a separate project phase. As we indi-

cated in the introduction, agile is all about the readiness to embrace change throughout a project. Not only requirements may change, but also the end product, service, or what is considered acceptable release (PMI, 2021; 2017).

Whether at the organisational level (Hernaus et al., 2020) or at the project level, agility encompasses somewhat similar guiding principles, such as facilitating responses to change and viewing changes as opportunities (Zhang and Sharifi, 2000). Similarly, the identification of risks is performed throughout a project (Lunesu et al., 2021) compared to the predictive approach, where it is mostly performed during the planning phase. Further, agile and adaptive software development is based on customer involvement and feedback throughout the project, compared to the predictive approach, where customers' input is concentrated at the start and at the end of the project. Moreover, documentation in an agile setting is focused on the essentials (Behutiye et al., 2022) compared to a traditional setting where documentation includes detailed plans and extensive requirements' documentation. The adaptive (e.g., agile) approach is also based on regular review, while the predictive approach is based on timeline tracking. Finally, adaptive and agile teams (Zainal et al., 2020) are often self-organising, cross-functional teams with a high degree of autonomy, while teams in traditional settings have a hierarchical team structure with specialised roles.

Predictive approaches have a long tradition, whereas adaptive approaches emerged later. The agile manifesto, agile values, and agile principles (Agile manifesto, 2001) have triggered further development and expansion of numerous agile approaches, as well as hybrid approaches, which are known to integrate the characteristics of both adaptive and predictive approaches (PMI, 2021; Wysocki, 2019). The adoption of specific hybrid project lifecycles and specific hybrid approaches such as Scrum (Schwaber, 1997), Kanban, extreme programming, Scrumban, Crystal methods, and others (Abrahamsson et al., 2017; Edison et al., 2022; Alqudah and Razali, 2016), has become widespread (PMI, 2017).

The popularity of agile approaches has contributed to a desire to highlight the benefits of agile approaches, such as their flexibility in responding to changes, while shortcomings, such as risk anticipation and a lack of fixed planning that could prevent major unnecessary changes, unexpected costs due to late changes, or additional work in later stages, are frequently overlooked.

PMI (2017) and many other previous studies, among which Wysocki (2020) and Fink (2017) previously recognised the need to tailor specific approaches to particular project attributes. Actually, many software development projects require hybrid approaches that include a unique blend of adaptive and predictive characteristics. The optimal set of predictive and adaptive components (Karlström and Runeson, 2005, 2006; Davis, 2012; Fink, 2017; PMI, 2021; Wysocki, 2023) needs to be determined for the project and context at hand. Factors such as degree of innovation, requirements certainty, scope stability, ease of change, delivery options, risk, safety requirements, regulations (product, service, or result), stakeholders, schedule constraints, funding availability (project), organizational structure, culture, organizational capability, project team size, and location (organization) should be considered (PMI, 2021). The optimisation and fine-tuning of the combination of approaches resulted in many hybrid approaches (e.g., Žužek et al., 2020) that, by nature, combine the characteristics of predictive and adaptive (e.g., agile) approaches (PMI, 2017).

The expectations about the change in requirements (PMI, 2017) are an important factor to consider when developing the best-suited project life cycle. While iteration- or incremental-based agile approaches coincide with hidden and misunderstood requirements, merely iterative or incremental approaches assume that feedback in between enables better further planning of the project (PMI, 2017). The agile approaches are therefore more suitable for projects where there is high uncertainty regarding requirements and it is expected that requirements "will change based on customer feedback" (PMI, 2017). The predictive approaches, on the other hand, are suitable for low-risk serial projects and projects requiring substantial investment. They focus on the fact that the cost of change (PMI, 2021) increases exponentially throughout the project duration. Simply put, the cost of introducing a change increases the later it occurs. Predictive methods are more appropriate in cases where fail-

ure to integrate key parameters, assumptions, and premises related to requirements beforehand or in the early phases would lead to substantial drawbacks. Failure to do so could result in severe irreversible consequences, such as unexpected repair costs, the need to remove previously completed work, a chain reaction of additional changes, adjustments, rework, and additional work, or even jeopardize the system's long-term operation.

2.3 The requirements engineering process

Since the user requirements represent the foundation for an ERP project, they are important not only for the choice of the type of solution but in general for the development of the solution itself. In particular, in an agile setting, the requirements are often underspecified, un-clear, and missing and are the subject of emerging refinement. On the contrary, in a stable setting where requirements are perfectly clear, there is no need to introduce agile components. Therefore, the characteristics of the requirements themselves are also important for determining the set of agile components that should be applied (PMI, 2017) to a particular project.

Requirements engineering is a challenging process that, according to ISO (2018) standards, results in "requirements for system and software products throughout the life-cycle."

This can include their initial definition, analysis, specification, validation (Atoum et al., 2021), prioritisation (Regnell et al., 2001), requirements management, and system modelling (Sommerville, 2009). A widely used taxonomy (Laplante and Kassab, 2022) categorises requirements into user requirements, system requirements, and design specifications (Sommerville, 2005), based on their level. Essential for acceptance testing, user requirements often include conceptual papers and user stories. They outline the specific functionality that the system should offer to users. They describe desired system behaviour in a manner that is comprehensible to the business user, from their perspective.

System requirements, often referred to as functional specifications or technical annexes, are essential for conducting integration testing (Laplante and Kassab, 2022). These requirements outline the behaviour of a system, typically in relation to functions that have been previously defined in user requirements. Design specifications, derived from system requirements, are essential for conducting unit testing (Laplante and Kassab, 2022).

A frequently used taxonomy categorises requirements into functional, nonfunctional, and domain requirements, depending on their specification types. Functional requirements delineate the specific services that the system is expected to offer and its corresponding responses to the inputs it receives (Laplante and Kassab, 2022, p. 6). High-level functional requirements align with the user's needs, whereas detailed functional requirements align with the system's requirements (Laplante and Kassab, 2022).

The system is defined not only by its functionalities but also by its nonfunctional behaviour. The NFRs (Non-Functional Requirements) (Laplante and Kassab, 2022; Rahy and Bass, 2022; Behutiye et al., 2020, 2022; Jarzebowicz and Weichbroth, 2021; and Karhapää, 2021) encompass a range of issues including security, reliability, dependability, reusability, maintainability, performance, usability, testability, interoperability, and constraints. Domain requirements encompass several aspects, such as introducing new functional requirements (FR), imposing limitations on existing FR, or determining the specific functions inside a given application domain (Laplante and Kassab, 2022).

Requirements are multi-layered and are reliant on the characteristics of the product and the source of requirements (Chemuturi, 2012; Regnell et. al., 2001; ReqView, n.d.). The identification of the higher-level requirements leads to the approval of a project in the first place (Matende and Ogao, 2013). More detailed requirements' documentation and lower-level requirements can be gathered and analysed throughout the entire project duration and even after the project's completion. Different techniques for conducting the requirements analysis (Köse, 2019; Nuseibeh et al., 1994; McGraw and Harbison, 2020) are more in line with either adaptive (e.g., agile) or predictive methodologies, or a combination of those. As the project management processes shall be suited to the project characteristics (PMI, 2017), so should the appropriate tech-

nique for requirements analysis (Guillemette et al., 2021). The complexity of the project itself is important for the complexity of the requirements engineering process. Nevertheless, it is meaningful that Theunissen et al. (2022), who investigate practical applications of agile methodologies, suggest there is a need for increasing the quantity and quality of requirements' documentation.

In the process of requirements engineering, a team should certainly consider that requirements need to be aligned, communicated, and prioritised (Rengell et al., 2001). Any miscommunication or misalignment can seriously jeopardise the achievement of the goals, so it is necessary to pay a lot of attention to the fact that the requirements are communicated in a way that everyone understands them equally (Robertson & Robertson, 2012).

2.4 Change management

The ever-changing nature of work and the rapid technological changes made practitioners and researchers realise that projects in which project goals are not clearly defined and the project activities are uncertain require approaches that, in their nature, pre-assume greater flexibility and embrace changes. The matrix of Wysocki (2011, 2014) and the Stacey complexity model (Stacey et al., 2000; PMI, 2017) support this notion by showing that the level of uncertainty greatly affects how the project should be approached. Despite the differences between project methodologies, any project and requirements engineering process must include a change control process, which ensures that any additions are "identified, documented, approved, or rejected" (PMI, 2021), as well as change management, "a comprehensive, cyclic, and structured approach" (PMI, 2017, p. 164) that embraces practices, skills, and techniques for transforming people, groups, activities, projects, or organisations to another state. Whether in a predictive or agile setting (L'ecuyer and Ahmed, 2016), where change is the only constant, it is important to evaluate the added value and anticipated investment associated with changes to be implemented. An efficient change control system (PMI, 2021), which sifts through meaningful changes through a formal approval process, ensures that the acquired value outweighs the resources, effort, and costs invested. Besides, it also ensures that "scope creep" (PMI, 2021) is avoided, especially in predictive settings, meaning that additions to scope or requirements should result in adjustments to the schedule, budget, and resources. Adequate planning and adequate risk anticipation can prevent overload of changes and hectic schedules, as well as planning of the activities that are likely to change during the course of the project. Efficient planning, change management, and coordination are considered important success factors.

In particular, agile approaches often closely intertwine with concepts of change management and are fundamentally based on requirements' uncertainty and the technical degree of uncertainty (PMI, 2017). Agile methodologies begin with less-defined requirements and include more iterations of shorter duration, each of which begins with a requirements analysis. In agile projects, even the project scope may be subject to change. During subsequent iterations, changes and details are added.

Predictive approaches to managing projects are based on quite the opposite: a clear definition of project goals, project activities, and, to the greatest degree possible, the elimination of all unknowns. Predictive approaches devote more attention to requirements analysis, detailed planning, and anticipation of risks and changes during the initial phases. This does not mean, however, that the predictive methodology does not foresee the possibility of introducing changes during the project, including "frequent reviews, change control mechanisms, and replanning between development phases" (PMI, 2021), but it does so in a clearly defined setting with a relatively stable scope.

The changes, especially those introduced in later project phases, can drastically increase duration, waste resources, and compromise the scope of any project. Though in some projects, particularly those led predominately based on predictive approaches, it might be impossible or extremely costly to introduce changes after the work has already been performed. However, requirements' documentation can be subject to change even in predominately traditionally led projects. This is particularly true for complex and high-risk projects such as ERP.

Project teams must decide how much time to devote to the requirements engineering process during the early stages of the project. In this regard, agile and iterative practices rely on shorter increments through which partial but continuous improvement of user requirements gradually evolves. The changing nature of work and the rapid pace of technological change stimulate a high demand to change and upgrade more detailed user requirements on a regular basis (Al-Ghamdi and Saleem, 2018). Predictive methodologies, on the other hand, are founded on a clear and precise description of user requirements during the project's early stages. In predictive methodologies, user requirement analysis is typically developed in greater detail and finalised earlier in the initial project phases. More attention is devoted to immutably determining key requirements, coordinating the understanding of key requirements among team members, and defining the specialisation of team roles.

However, even in predominantly traditionally led projects, the user requirements can be subject to change (IPMA, 2016), and on the other hand, even in predominantly agile-led projects, the need for clearer definition, documentation (Theunissen et al., 2022), and communication of user requirements exists.

3 RESEARCH QUESTIONS

We divide research questions into two parts: 1.) the requirements engineering process and 2.) change management.

3.1 The requirements engineering process

User requirements' documentation can be created using a variety of methodologies, best practices, and industry recommendations. We're curious about how the project management methodology influences user requirements' analysis. What are the characteristics of the requirements' gathering and analysis process? The company's internal regulations frequently determine the choice of a particular methodology and modifications to it. We want to inquire if the methodology used to gather and analyse user requirements affects the quality of the requirements' documentation and the final project outcome.

First research question (RQ1): *How does the appropriate combination of adaptive (e.g., agile) and predictive waterfall methodologies affect the quality of requirements' documentation and the final outcome of an ERP system implementation project?*

Second research question (RQ2): *How does the level of awareness in the organisation and project management about the importance of quality user requirements contribute to project success?*

3.2 Change management

We investigate whether the fact that changes to user requirements' documentation are introduced only during the first phases of the project rather than throughout the entire duration of the project affects the project's final outcome. We are particularly interested in whether change requests contribute to project success.

Even experienced teams find it difficult to accurately state all of the requirements prior to the start of the project and to capture all user experiences and user stories in advance. This would necessitate the participation of a large number of users, which is not only difficult to coordinate but mostly time-consuming. As a result, understanding that changes during projects can improve project outcomes, that changes should be seen as enhancements and additions rather than corrections of past errors, and that incorporating change management processes benefits projects is critical for any project, regardless if it is predominately applying predictive or adaptive (e.g., agile) methodologies.

Third research question (RQ3): *Do constant modifications and improvements to user requirements' documentation throughout the project, as opposed to fixed or static requirements, increase the likelihood of meeting project goals?*

When the project's scope, goals, and high-level requirements are lightly defined, it not only impacts the volume of changes to tasks within the project but also the volume of tasks that must be performed after the project's completion or during the post-implementation phase. Especially in an agile environ-

ment, this can lead to an excessive volume of change requests during the project and additional improvements requested after implementation.

Fourth research question (RQ4): *What strategies may a project team employ to address an excessive volume of change requests during the project and additional improvements requested after the completion of an ERP system implementation project?*

4 RESEARCH METHODOLOGY

As is common in IS and software engineering research (Runeson and Höst, 2009), an exploratory case study methodology is applied. The ERP system implementation project was meticulously chosen via judgmental sampling. The main reason for applying judgmental sampling is that the unique insights can be rightfully presented by important project stakeholders with specialised, unique expertise. Since ERP implementation projects are often unique, complex, and risky, information asymmetry is often prevalent, meaning that only decision-makers with direct involvement may be able to provide nuanced insights relevant to the research. Judgmental sampling enables a deeper and more comprehensive understanding of the complexity, the associated risk, and the dynamics of changing requirements in an ERP project. To avoid the possible shortcomings of judgmental sampling, we carefully selected the interviewee, a project manager who has a wide perspective, is a decision-maker, and has an overview of the entire project. A project manager who works for a large, international company and has rich experience in leading projects for the development and improvement of enterprise system solutions was the best suited person to provide us with the insights.

The research questions were derived from the research gaps that were found during the literature review. The measurement instrument was specified in advance. We applied three different types of data collection methods: collecting information through a semi-structured interview based on open questions, collecting further details about the key project characteristics through a short survey, and forming the study itself through an extensive literature review. The primary data are qualitative in nature and were obtained at a particular time. The interviewee and interviewers made an oral agreement outlining the confidentiality requirements. Two researchers conducted a one-hour interview with the project manager. Open-ended questions were used to elicit specific data. To minimise the impact of prior experiences or assumptions of individual researchers, all research participants carefully chose and prepared the questions in advance. After the interview, we prepared and reviewed the transcripts to identify and select the findings, and we organised a discussion about the theoretical and practical ramifications of the research questions. The recommendations (Runeson and Höst, 2009; O'Brien et al., 2014) and examples (Karlström and Runeson, 2006; Andersson and Runeson, 2007) for reporting qualitative research, as well as the recommendations for interview question preparation and qualitative interview strategies (Harvard, n.d.), are followed as closely as possible in the preparation of the qualitative research. Throughout the paper, we support the discussion and conclusions with quotes. With that, we enhance our comprehension of prevalent practices and methodologies applied in ERP projects that facilitate the achievement of higher-quality UR.

When assessing the validity of the study, it is reasonable to consider that this research primarily represents the viewpoints of a customer of an ERP system. We did not interview other project stakeholders, such as the software company, internal and external team members, supervisors, or users. We did not collect archival data like process models or specifications. However, by carefully selecting the interviewee, we partially addressed reactivity, researcher bias, and respondent bias, which, according to the Lincoln and Guba model (Robson, 2002; Karlström and Runeson, 2006), represent possible threats to validity. Since it is challenging to ensure the validity of findings based on judgmental sampling, our research also includes limitations such as limited generalizability, challenging assessment of representativeness, and challenging validation of results.

5 CASE STUDY

In our particular case study, the estimated cost of the upgrade ERP system implementation project ranged from 5 to 10 million euros. After 18 months, the initial solutions went live. Although

the schedule and the use of other resources diverged from the original plan, the system's successful implementation was not in any way jeopardised. Both the project's deadline and budget were met. Different modules, or subsystems, make up ERP systems. As shown in our case study, the modernization, renovation, and reengineering of important business processes in an enterprise system is a complex project requiring a sizable investment and the involvement of numerous stakeholders. The final solution had an impact on more than 1000 business users, and the internal core project team consisted of 20 to 25 individuals. The project involved a large number of external parties, including about five different businesses and additional independent contractors.

The primary objectives of the project were to streamline, modernise, standardise, and improve the intricate process of product release. The decision to release a product must be made after gathering a lot of data and checking a lot of control points. The objective of the project was to make it possible for data to be automatically gathered from various systems and sources and displayed on a dashboard. Decision-making is made easier and more quickly when information is accessible at a glance in one place.

The project was successful in terms of user adoption and satisfaction. The internal staff is very pleased with the software solution. While acknowledging that there is still room for improvement, the project manager is generally pleased with the solution, the collaboration with external contractors, and the collaboration within the internal team, as well as with the professional competence, technical expertise, and process knowledge of the internal staff. The successful implementation of an ERP system allowed us to identify good practices and criteria leading to success, as well as strategies to overcome challenges ERP projects might face. These are the reasons that led us to choose this project as a case study. In the continuation of the study, we provide further insight about the particular project methodology and requirements change management by supporting the discussion around the research questions with quotes.

5.1 The requirements engineering process

The project team, under the leadership of the customer of the ERP system, adjusted project management methodology to the specific character of the project (RQ1). Although it followed the company's general guidelines for project management methodology, it developed specific project methodology for the project. The validation plan specified the specific methodology. The customer, on the one hand, put an emphasis on executing a lot of the planning activities before the project started, which is typical of predictive approaches. The project, on the other hand, was organised in iterations, or rolling waves, which means that activities within work packages were defined in greater detail as the project progressed. The project team combined predictive and adaptive methodologies but did not use ordinary sprints, scrums, or strictly predictive approaches. A clear scope and thorough specification of business requirements defined in the early project phases contributed greatly to the overall successful implementation of the solution.

> *"The three main project packages, design, build, and testing, were performed more or less in parallel,"* says the project manager. "*The activities were intertwined. This flexible approach was defined in the validation plan from the very beginning."*

The planning process and requirements analysis included several rolling waves. Additional plans were developed after the first wave. A roadmap of a product release process with steps was an important input.

Some approaches in the very early stages of custom development, for example, included requirements that were then reviewed and signed off on before being handed over to the developers to develop the software. The requirements for the portion of the project that included standard solutions were not signed off on. Instead, they reviewed and signed off on the solution after it had been tested and approved informally. Furthermore, while the project was based on some high-level documents, the team did not wait for all tasks to be completed and compared the API documentation to proceed. Instead, the activities were carried out simultaneously.

Another aspect that we investigated was the level of awareness about the importance of quality user requirements (RQ2). The project manager stressed several times during an interview that the business requirements are a starting point and that it is crucial that IT and other experts understand the needs of the users. It is very important that the IT team members understand the requirements of the business team members, which are the primary drivers. Regardless of the context or type of project, understanding business requirements is a good place to start.

5.2 Change management

Our research supports the notion that managing changes, particularly changes in requirements, is the biggest challenge of ERP projects that strive for a high degree of user adoption and, on average, satisfied users (RQ3).

> In our specific case study, the project manager emphasises that the customer's *"expectation was never that the implemented system would represent the final solution."*

The high number of change requests to further improve the process during the project implementation and after the system implementation was officially completed is a reality that the customer acknowledged from the elaboration of the project onwards. In the future, they want to further improve and develop the product release process. The team gained important experience, which they made good use of during the post-implementation improvement of the processes. The high number of change requests in no way threatened the successful implementation of the system. Nevertheless, the manager acknowledges that the number of change requests could be decreased to some degree sooner. They consider the process to be a living organism that can always be improved.

The general scope of the project and high-level requirements were clearly determined early on, but after the conceptual design phase, an updated version of the user requirements' documentation was prepared. Throughout the entire project, they strived to optimise the requirements, and they continue to do so even during the post-implementation phase. After the detailed design documentation and development, the system demos were performed. During system demos, they sometimes realised that *"the requirements should be rephrased or that some important thing needs to be added."* The collaboration between IT and business team members was very close, and discussions about the updates took place almost throughout the entire project.

"The change requests were quite intense. In certain areas, the solution and the timeline have changed substantially," said the manager. The project changes made during the course of the project had an impact on the scope, the timeline, and the use of resources. However, each change was carefully approved and introduced in a controlled manner. For example, the system integration issue that occurred later in the project automatically triggered the need to add changes to the high-level requirements' documentation. The changes to the documentation were added up until the testing phase. Even after implementation, there are still requests for changes. As a consequence of the changed timeline, the system integration was late, and some projects that were performed in parallel on other systems had to be temporarily stopped. Nevertheless, the successful implementation of the system was in no way compromised.

According to the manager, the IT team may also be the one to initiate changing user requirements, not just the business part of the team. Especially when IT professionals recognise that some solutions may have a negative impact on end users or work better in another way. In that case, the team comes together to discuss how user requirements can be adjusted to improve the impact on the end user. In such a case, the team needs to work together to discuss these changes. Depending on the size of the impact of a change, the business change manager then coordinated the discussion and gathered feedback about the impact of user requirement changes. In the event of a change, users are notified and asked to provide feedback within a week. Communication with end users was also intense during requirements analysis and pilots in partner companies. The business process owners gathered user stories and feedback from the end users. During the pilots, one or two members from each pilot site joined the core team.

A change request was usually first coordinated between the person who started the change and the company's project management. After internal alignment, the project manager started to negotiate the change with the software company. With each request for a change, the agreed-upon project scope could be delayed and/or cost more than originally planned. So, compromises had to be made so that extra requests for changes didn't cause big changes to the agreed-upon project scope but represented small improvements to the quality of the processes. Some change requests were put on hold and added to the backlog for implementation within the next releases of the software since delaying the project was not an option. It was crucial to develop a suitable business solution by the agreed-upon deadline so that businesses could gain benefits as soon as possible and the project would not get cancelled. Because of this, everyone agreed that further software releases would include more process optimisation.

The project manager adds that, of course, the process *"has to be further improved and enhanced based on feedback"* obtained during the post-implementation phase. Furthermore, each product release represents an opportunity to find new ways to improve the process. The fact that the requirements are constantly optimised and gathered is important for future projects as well.

On the other hand, the project managers clearly stated that some change requests during the project as well as additional works requested after project completion could be avoided and addressed earlier on during the project (RQ4). This proves that it is challenging to find the right balance between predictive and adaptive approaches to planning, even in otherwise successful ERP system implementation projects. According to the assessment of the project manager, the project faced too many change requests. The users who are invited to provide feedback on partial solutions were reluctant to speak out or to provide comments on the solutions that were developed. Therefore, the team readily implemented some improvements to avoid as many change requests during the project and the post-implementation phase of this or other projects. It became clear that the testing procedure contributed to the change management project and that the manager realised that improving testing and, in particular, the validation process could yield even more favourable results.

> *"We plan to perform the rollouts at other sites. We have improved and will continue to improve the validation process. We formalised the process of validating the functionalities by introducing checkpoints. They must accept the solution from the start. That way, we can be certain that they are completely satisfied with the solution provided for each functionality. During the post-implementation phase, we began collecting and evaluating their major pain points. That is something we are still working on. We intend to improve the system and add improvements so that users can truly benefit from the new system."*

6 DISCUSSION

The paper addresses the requirements engineering process and change management, which we further discuss considering both the theoretical and practical ramifications. We round off the discussion with limitations and future research.

6.1 Theoretical contributions

The presented case study is an excellent example of high customer involvement in the development of an ERP system. The customer was not only fully involved throughout the project but also took on the leadership and coordinated project partners and activities. In our case study, there was no issue with lack of customer involvement, their inability, or non-alignment that was observed by Ramesh et al. (2010) on some other agile-based software development projects. Having a competent customer who is able to monitor the course of the project proved to be an important success factor. In large part, the customer determined not just the team culture but also the project methodology. The project team established its own unique approach to project methodology that was derived partly from the customer company's project management standards and involved a blend of predictive and adaptive methodologies. This had an effect on the quality of the requirements' documentation and the final

outcome. A unique combination of predictive and adaptive elements was applied to address the needs of the project, the characteristics of the final system, and the characteristics of all involved organisations to the best degree possible (RQ1). This has worked well since it addressed the requirements of the company, an international organisation working on large-scale projects, the characteristics of the particular process and the ERP system itself, as well as the requirements of the pharmaceutical industry and its standards. It is also in line with the challenges that often arise when applying agile methodologies to large software development projects (Alqudah and Razali, 2016). Further, the case study presented is also an excellent example of how two fundamentally different business cultures, the pharmaceutical and IT sectors, can effectively work together, given that a customer takes the lead in how the project should be approached.

The literature shows both theoretical ERP implementation models and standardised ERP implementation steps applied in practice (Lutovac and Manjolov, 2012), as well as examples of how agile methodologies were incorporated into ERP implementations (Nagpal, 2015; Kraljić & Kraljić, 2020; Kraljić et al., 2014) and large software development (Alqudah and Razali, 2016). However, the majority of prior research predominantly focuses on the perspective of software companies, such as SAP, Oracle, or others, with limited consideration given to the viewpoints of customers of ERP systems.

The case study confirms the existing findings regarding the importance of selecting the right blend of predictive and adaptive methodologies for managing projects that is best suited for a particular project. The project teams are constantly choosing between the thoroughness of an initial plan and the additions to the initial plan that they are willing to make later in the project (Rasheed et al., 2021). On the one hand, it is known that it is more difficult, challenging, and time-consuming to introduce changes later during the project. On the other hand, teams under time pressure want to begin developing solutions as soon as possible. Moreover, to further improve and develop the plan, the team would require more information, which can only be obtained once the development phase has already started. As a result, project phases frequently run concurrently. Though the lack of time should not compromise the quality, perfecting and fine-tuning the initial plan in the initial project phases or before the start of the project takes valuable time. Plan additions and details can be added later during projects. The compromise between thoroughly preparing the plan as early as possible and the number of additions to be added later is frequently context-specific. The characteristics of the project and of the project activities can be crucial for determining the best combination of predictive and adaptive approaches in a concrete project. Predictive and adaptive approaches have certain advantages and disadvantages. Even if adaptive approaches regard changes as opportunities, it might be hard, if not impossible, to introduce some major changes to the scope of the project later in the project due to many reasons, such as financial ones. Finally, we can draw the conclusion that both the internal company's project management regulation and the project specifics have an impact on project methodology.

Additionally, we can conclude that it is, regardless of the methodology applied, very important that the high-level requirements are agreed upon in advance, before the start of the project. The presented case study showcases this. A clearly defined scope and high-level business requirements guaranteed the success of the project, which was enhanced by the active involvement and assertiveness of users in activities such as requirements' collection, analysis, testing, and verification. Similarly, even when changes to requirements are allowed in later project phases, as is typical for agile approaches, it is crucial for the project team to be aware of the importance of high-quality business requirements (RQ2), and any miscommunication between members of the IT and business teams should be resolved as soon as possible to ensure that requirements are properly documented.

The rationale behind this may lie in the fact that, in any case, there is a high risk of requirements changing in ERP projects (Sudhakar, 2012). This is due to several reasons, such as the fact that the implementation of enterprise systems typically results in a substantial shift and transformation in the execution of processes, and it in general entails the modernization of processes. Furthermore, since information technology supports processes that behave as living organisms and adhere to the tenet that you lose it if you don't use it, they must constantly adapt and advance.

Change management is critical in ERP projects not only during the conceptualization and implementation phases but also after the project has been completed, during the post-implementation phase. Change management on an ERP project is inextricably linked to ongoing requirements analysis. The efficient change management process associated with requirements is critical for improving the system and information quality, especially in complex projects such as ERP projects with a large number of high- and low-level requirements and a large number of stakeholders.

Based on the case study presented, we can conclude that the companies should constantly optimise their requirements analysis based on lessons learned from previous projects and ensure that past lessons are applied company-wide in future projects (Al-Ghamdi and Saleem, 2018). The willingness of a customer to invest in further upgrades can speak about the success of the project, since customers who do not see benefits in the new system would be unwilling to undertake further upgrades (Markus et al., 2000). As a result, we find that modifications and improvements to user requirements' documentation throughout the project can increase the likelihood of meeting project goals (RQ3), prevent possible unnecessary changes, and incorporate some work that would otherwise be performed outside the project either as additional work or within the new project. The testing, and specifically the validation process, was shown to have an important impact on the change management process. Enhancing the testing and its early introduction may further improve the already positive results.

The volume of change requests during the project and post-implementation is not always an indicator of project success, but sometimes quite the contrary. Nonetheless, by actively involving all project participants throughout the project, many unnecessary post-implementation changes can be avoided. Project members, particularly business participants, shall actively participate in the process of defining and reviewing business requirements. Overall, we identified two strategies, which the case study exemplifies and a project team employed to address an excessive volume of change requests during the project and additional improvements requested after the completion of an ERP system implementation project (RQ4): high-quality requirements and, most of all, a formalised testing and verification process. We further expand on these very practically-oriented strategies in terms of their practical implications.

6.2 Practical implications

Before we further elaborate on the aforementioned strategies, let us summarise several further key findings that are of importance with respect to the practical implementation of future ERP systems.

First, customer involvement, monitoring, and possibly leadership of ERP system implementation are highly advised. Then, the specific project methodology that includes a blend of predictive and adaptive methodologies should be determined by the project team, based on the characteristics mainly of the customer's organisation where the ERP system is to be implemented, but also on the characteristics of other project stakeholders, the specific processes, and the ERP system itself.

Frequently, important factors such as project attributes, project tasks, customer and stakeholder organisation, and key team members are insufficiently considered in practice when determining the most suitable project life cycle. However, these factors are crucial in determining the appropriate approach to project management. Not all sizes fit all. As the leadership style should adapt not only to the situation but also to the personal characteristics of the follower and leader (Griffith et al., 2018) and the research methodology to its purpose (Runeson and Höst, 2009), so should the project methodology to its project. Customers of an ERP system should not get caught in a trap by expecting that the software company will take the lead in proposing the methodology best suited to their software solutions.

As observed in our case study, the customer took charge of the implementation process, led the main development initiatives, and controlled the course of events. Unless the customer is active and alerts the software company of the necessary adjustments, the course of events can go in the wrong direction. The role that customers play in the ERP implementation project can be crucial for its success. When the customer is actively resolving the

problems that arise on the project together with the software company, has the majority of the thread in his hands, is the one who has constant control and preview of the implementation, holds a decisive role, and makes key decisions in agreement with the software company, the ERP implementation project is much more likely to be successful.

A well-defined scope and comprehensive high-level requirements, as well as the active participation and assertiveness of users in activities such as requirements' gathering, analysis, testing, and verification, can further improve the project's outcome. In practice, however, the initial requirements and final result may deviate. It may happen that high-level requirements approved initially by the customer do not include the whole scope they should, or that the customer confirms a too-complex solution or an add-on that brings little benefit compared to the additional work it requires and the substantial increase in complexity of the overall model. In such cases, the early inclusion of testing might be particularly beneficial. Early testing, based on a clear list of testing activities derived from the requirements list, might assure prompt alerts about the things that should be included in the requirements but were not or about things that are just redundant since they bring too little benefit.

To address an excessive volume of change requests during the project and additional improvements requested after the completion of an ERP system implementation project (RQ4), project teams should give more attention to the preparation of high-quality requirements and, most of all, to enhancing their formalised testing and verification processes. *High-quality requirements:* Even when agile approaches are applied, it might be challenging to implement changes after the key building blocks of the system have already been determined and the architecture of the enterprise system is set. Therefore, it might be risky to start the development before all key building blocks are determined and included in the initial plan. Further, due to time constraints, the testing of partial solutions against the requirements might be limited. Similarly, the opportunities for fine-tuning and preparing a more user-friendly solution might not be fully realised due to time pressure. Therefore, we advise that at least high-level requirements are clear to all stakeholders in advance and that awareness about high-quality requirements is raised. *Formalised testing and verification processes:* In the absence of formalised testing and verification processes introduced early on, the development might continue too long in the wrong direction, and many opportunities for improvement and fine-tuning during the development phase might be missed. Feedback on partial solutions is crucial since it ensures an appropriate course of development. Therefore, formalising the testing and verification processes could contribute to reducing the number of changes in later stages, after the implementation of the solutions.

6.3 Limitations and future research

ERP implementation teams should carefully examine what is best suited for the project at hand rather than applying a methodology similar to those adapted by organisations that have previously successfully implemented ERP systems. It needs to be mentioned that, besides the selected project and the project manager, additional relevant projects or project team members were not included in our sample. Even if the exploratory study enables qualitative, in-depth insight into the dynamics of a particular case study, it is possible that parts of the conclusions are only valid for the particular project. As mentioned previously, it is challenging to ensure validity, generalizability, or representativeness based on judgmental sampling.

7 CONCLUSION

This case study-based qualitative research investigates the requirements engineering process and the requirements' change management process in an ERP system implementation project. In relation to the requirements engineering process, we investigate the use of adaptive (e.g., agile) methodologies versus traditional, predictive waterfall methodologies and the importance of user requirements in an ERP project. The project team constantly chooses between the thoroughness of an initial plan and the additions they are willing to make later in the project. Agile methodologies offer both benefits and drawbacks, but even in these projects, it may be challenging to implement substantial changes to the initial project's scope. The characteristics of the project and project activities

can determine the best combination of adaptive (e.g., agile) and predictive approaches for a concrete project. Further, we assert that the level of awareness in the organisation and project management about the importance of quality user requirements contributes to project success.

ERP projects that aim to optimise and modernise business processes are complex and require constant change management to improve system and information quality. We compare constant modifications versus fixed requirements and address the strategies that influence the volume of change requests during and after the system's implementation. Managing changes, particularly those in user requirements, is the biggest challenge for ERP projects that strive for a high degree of user adoption and satisfaction. In the case study, the project manager acknowledges that the customer's expectation was never that the implemented system would represent the final solution. Moreover, a clearly defined scope and well-articulated business requirements ensured the project's success. User participation and outspokenness during requirements gathering, analysis, testing, and verification are important factors that contribute to the effectiveness of the process. The factors that influence the number of change requests during the project and the post-implementation phase include awareness about the importance of high-quality requirements and formalised testing and verification processes. Without formalised testing and verification processes, development may continue in the wrong direction, which causes additional change requests in later stages. Though the number of change requests during post-implementation is not always an indicator of project success, by actively involving all project participants throughout the project, many unnecessary post-implementation changes can be avoided.

EXTENDED SUMMARY/IZVLEČEK

Članek obravnava zbiranje, pripravo in razvoj uporabniških zahtev. Raziskava sloni na študiji primera kompleksnega ERP projekta v mednarodni korporaciji. Izhajamo iz dejstva, da razvoj posamezne ERP rešitve pogosto presega en sam projekt. Pogosto se razvoj ERP rešitev nadaljuje tudi po zaključku posameznega projekta. Osredotočamo se na zahteve uporabnikov in njihovo kakovost predvsem iz vidika procesa inženiringa zahtev in iz vidika managementa sprememb. Razpravljamo o značilnostih adaptivnih (e.g., agilnih) in predikativnih pristopov ter o vplivu teh na kakovost uporabniških zahtev. Poudarjamo, da je pomembno, da se člani tima zavedajo pomembnega vpliva, ki ga imajo dobro opredeljene uporabniške zahteve za uspeh projekta. Čeprav trdimo, da število zahtevkov za spremembe ne bi smeli pojmovati kot kazalnik uspeha projekta, razpravljamo o dejavnikih, ki vplivajo na število sprememb tekom projekta in na dodatno delo po zaključku projekta. Prepoznali smo dve strategiji s katerimi lahko vplivamo na preveliko število zahtevkov za spremembe in dodatna dela. Projektni tim bi moral več pozornosti posvetiti pripravi visokokakovostnih uporabniških zahtev in predvsem izboljšanju formaliziranih procesov testiranja in verifikacije rešitev. Obvladovanje sprememb, zlasti sprememb uporabniških zahtev, je bistveno za uspeh projekta. Podjetja bi morala nenehno optimizirati analize zahtev na podlagi izkušenj, pridobljenih iz prejšnjih projektov, in zagotoviti, da se pretekla spoznanja uporabljajo v celotnem podjetju v nadaljnjih projektih.

REFERENCES:

Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2017). Agile software development methods: Review and analysis. *VTT Publication, 478*, Espoo, Finland, 107p.

Agile Manifesto. (2001). Principles behind the Agile Manifesto. http://agilemanifesto.org/principles.html

Al-Ghamdi, A. S. A. M., & Saleem, F. (2018). General characteristics and common practices for ICT projects: Evaluation perspective. *International Journal of Advanced Computer Science and Applications, 9*(1).

Aloini, D., Dulmin, R., & Mininno, V. (2007). Risk management in ERP project introduction: Review of the literature. *Information & Management, 44*(6), 547-567.

Alqudah, M., & Razali, R. (2016). A review of scaling agile methods in large software development. *International Journal on Advanced Science, Engineering and Information Technology, 6*(6), 828-837.

Al-Saqqa, S., Sawalha, S., & AbdelNabi, H. (2020). Agile software development: Methodologies and trends. *International Journal of Interactive Mobile Technologies, 14*(11).

Andersson, C., & Runeson, P. (2007). A spiral process model for case studies on software quality monitoring—method and metrics. *Software Process: Improvement and Practice, 12*(2), 125-140.

Atoum, I., Baklizi, M. K., Alsmadi, I., Otoom, A. A., Alhersh, T., Ababneh, J., Almalki, J., & Alshahrani, S. M. (2021). Challenges of software requirements quality assurance and validation: A systematic literature review. *IEEE Access, 9*, 137613-137634.

Baccarini, D. (1996). The concept of project complexity—a review. *International Journal of Project Management, 14*(4), 201-204.

Behutiye, W., Rodríguez, P., Oivo, M., Aaramaa, S., Partanen, J., & Abhervé, A. (2022). Towards optimal quality requirement documentation in agile software development: A multiple case study. *Journal of Systems and Software, 183*, 111112.

Behutiye, W., Seppänen, P., Rodríguez, P., & Oivo, M. (2020). Documentation of quality requirements in agile software development. In *Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering*, 250-259.

Bick, S., Spohrer, K., Hoda, R., Scheerer, A., & Heinzl, A. (2017). Coordination challenges in large-scale software development: A case study of planning misalignment in hybrid settings. *IEEE Transactions on Software Engineering, 44*(10), 932-950.

Chemuturi, M. (2012). *Requirements engineering and management for software development projects*. Springer Science & Business Media.

Copola Azenha, F., Aparecida Reis, D., & Leme Fleury, A. (2021). The role and characteristics of hybrid approaches to project management in the development of technology-based products and services. *Project Management Journal, 52*(1), 90-110.

Dasović, B., Galić, M., & Klanšek, U. (2020). A survey on integration of optimization and project management tools for sustainable construction scheduling. *Sustainability, 12*(8), 3405.

Damian, D., & Zowghi, D. (2003). Requirements engineering challenges in multi-site software development organizations. *Requirements Engineering Journal, 8*(1), 149-160.

Davis, B. (2012). *Agile practices for waterfall projects: Shifting processes for competitive advantage*. J. Ross Publishing.

Edison, H., Wang, X., & Conboy, K. (2022). Comparing methods for large-scale agile software development: A systematic literature review. *IEEE Transactions on Software Engineering, 48*(8), 2709-2731.

Falkowski, G., Pedigo, P., Smith, B., & Swanson, D. (1998). A recipe for ERP success. *International Journal of Human-Computer Interaction, 16*(1), 5-22.

Fink, L., Fošner, A., Dobrovoljc, A., & Poznič, T. (2024). How user requirements affect the success of an enterprise system implementation project. *European Project Management Journal, 14*(1).

Fink, L. (2017). Competence heterogeneity in teams and success of projects: The case of construction teams (Doctoral dissertation). Ljubljana: Faculty of Economics.

Gonzalez-Barahona, J. M., Sherwood, P., Robles, G., & Izquierdo, D. (2017). Technical lag in software compilations: Measuring how outdated a software deployment is. In *Open Source Systems: Towards Robust Practices: 13th IFIP WG 2.13 International Conference, OSS 2017 Proceedings 13*, Springer International Publishing, 182-192.

Griffith, J. A., Gibson, C., Medeiros, K., MacDougall, A., Hardy III, J., & Mumford, M. D. (2018). Are you thinking what I'm thinking? The influence of leader style, distance, and leader–follower mental model congruence on creative performance. *Journal of Leadership & Organizational Studies, 25*(2), 153-170.

Guillemette, M. G., Frechette, S., & Moïse, A. (2021). Comparing requirements analysis techniques in business intelligence and transactional contexts: A qualitative exploratory study. *International Journal of Business Intelligence Research (IJBIR), 12*(2), 1-25.

Harvard. (n.d.). Strategies for qualitative interviews. *Harvard University*. https://sociology.fas.harvard.edu/files/sociology/files/interview_strategies.pdf

Hernaus, T., Konforta, M., & Sitar, A. S. (2020). A multi-informant assessment of organizational agility maturity: An exploratory case analysis. *Dynamic Relationships Management Journal, 9*(2), 85-104.

INCOSE. (n.d.). *International Council on Systems Engineering (INCOSE)*. http://www.incose.org/

Iivari, J., & Huisman, M. (2007). The relationship between organizational culture and the deployment of systems development methodologies. *MIS Quarterly, 31*(1), 35-58.

IPMA. (2018). *IPMA Reference Guide ICB4 in an Agile World*. International Project Management Association.

IPMA. (2016). *Project Excellence Baseline for Achieving Excellence in Projects and Programmes*. International Project Management Association.

IREB. (n.d.). *International Requirements Engineering Board (IREB)*. http://ireb.org/

ISO Standard. (2018). *ISO/IEC/IEEE 29148:2018 Systems and Software Engineering—Life Cycle Processes—Requirements Engineering*. https://www.iso.org/standard/72089.html

Jarzębowicz, A., & Weichbroth, P. (2021). A qualitative study on non-functional requirements in agile software development. *IEEE Access, 9*, 40458-40475.

Karhapää, P., Behutiye, W., Rodríguez, P., Oivo, M., Costal, D., Franch, X., Aaramaa, S., Choras, M., Partanen, J., & Abherve, A. (2021). Strategies to manage quality requirements in agile software development: A multiple case study. *Empirical Software Engineering, 26*(2), 28.

Karlström, D., & Runeson, P. (2006). Integrating agile software development into stage-gate product development. *Empirical Software Engineering, 11*, 203-225.

Karlstrom, D., & Runeson, P. (2005). Combining agile methods with stage-gate project management. *IEEE Software, 22*(3), 43-49.

Köse, B. Ö. (2019). Recent agile requirement engineering practices in IT projects: A case analysis. *Business & Management Studies: An International Journal, 7*(4), 1776-1805.

Kraljić, A., & Kraljić, T. (2020). Agile software engineering practices in ERP implementation. In *Information Systems: Proceedings of the 16th European, Mediterranean, and Middle Eastern Conference, EMCIS 2019*, Springer International Publishing, 279-290.

Kraljić, A., Kraljić, T., Poels, G., & Devos, J. (2014). ERP implementation methodologies and frameworks: A literature review. In *8th European Conference on IS Management and Evaluation ECIME*, Academic Conferences and Publishing International Limited, 309-316.

Kronbichler, S. A., Ostermann, H., & Staudinger, R. (2009). A review of critical success factors for ERP-projects. *The Open Information Systems Journal, 3*(1).

Laplante, P. A., & Kassab, M. (2022). *Requirements engineering for software and systems*. Auerbach Publications.

L'ecuyer, A., & Ahmed, S. A. (2016). Controlling change on agile software development projects. *Universal Journal of Management, 4*(1), 42-49.

Lunesu, M. I., Tonelli, R., Marchesi, L., & Marchesi, M. (2021). Assessing the risk of software development in agile methodologies using simulation. *IEEE Access, 9*, 134240-134258.

Lutovac, M., & Manojlov, D. (2012). The successful methodology for enterprise resource planning (ERP) implementation. *Journal of Modern Accounting and Auditing, 8*(12), 1838.

Markus, M. L., Axline, S., Petrie, D., & Tanis, S. C. (2000). Learning from adopters' experiences with ERP: Problems encountered and success achieved. *Journal of Information Technology, 15*(4), 245-265.

Matende, S., & Ogao, P. (2013). Enterprise resource planning (ERP) system implementation: A case for user participation. *Procedia Technology, 9*, 518-526.

McGraw, K. L., & Harbison, K. (2020). *User-centered requirements: The scenario-based engineering process*. Taylor and Francis.

Mirić, A. A., Černe, M., & Hernaus, T. (2020). Managing knowledge in organizations: On knowledge creation, renewal, hiding and forgetting. *Dynamic Relationships Management Journal, 9*(1), 1-3.

Mockus, A., Weiss, D. M., & Zhang, P. (2003). Understanding and predicting effort in software projects. In *Proceedings of the 25th International Conference on Software Engineering*, IEEE, 274-284.

Nagpal, S., Khatri, S. K., & Kumar, A. (2015). Comparative study of ERP implementation strategies. In *IEEE Long Island Systems, Applications and Technology*, 1-9.

Nuseibeh, B., Kramer, J., & Finkelstein, A. (1994). A framework for expressing the relationships between multiple views in requirements specification. *IEEE Transactions on Software Engineering, 20*(10), 760-773.

O'Brien, B. C., Harris, I. B., Beckman, T. J., Reed, D. A., & Cook, D. A. (2014). Standards for reporting qualitative research: A synthesis of recommendations. *Academic Medicine, 89*(9), 1245-1251.

PMI. (2021). *The Standard for Project Management and a Guide to the Project Management Body of Knowledge (PMBOK Guide)* (7th ed.). Project Management Institute.

PMI. (2017). *Agile Practice Guide*. Project Management Institute.

Rahy, S., & Bass, J. M. (2022). Managing non-functional requirements in agile software development. *IET Software, 16*(1), 60-72.

Ramesh, B., Cao, L., & Baskerville, R. (2010). Agile requirements engineering practices and challenges: An empirical study. *Information Systems Journal, 20*(5), 449-480.

Rasheed, A., Zafar, B., Shehryar, T., Aslam, N. A., Sajid, M., Ali, N., Hanif Dar, S., & Khalid, S. (2021). Requirement engineering challenges in agile software development. *Mathematical Problems in Engineering, 2021*, 1-18.

Regnell, B., Höst, M., Natt och Dag, J. N., Beremark, P., & Hjelm, T. (2001). An industrial case study on distributed prioritisation in market-driven requirements engineering for packaged software. *Requirements Engineering, 6*(1), 51-62.

ReqView. (n.d.). *ISO/IEC/IEEE 29148 Requirements Specification Templates*. https://www.reqview.com/doc/iso-iec-ieee-29148-templates/

RESG. (n.d.). *Requirements Engineering Specialist Group (RESG)*. https://www.bcs.org/membership-and-registrations/member-communities/requirements-engineering-specialist-group/

Robertson, S., & Robertson, J. (2012). *Mastering the requirements process: Getting requirements right*. Addison-Wesley.

Robson, C. (2002). *Real world research*. Oxford: Blackwell Publishers.

Royce, W. (2011). Measuring agility and architectural integrity. *International Journal of Software Informatics, 5*(3), 415-433.

Royce, W. (2009). Improving software economics. *IBM Corporation Software Group*, 1-40.

Royce, W. W. (1987). Managing the development of large software systems: Concepts and techniques. In *Proceedings of the 9th International Conference on Software Engineering*, 328-338.

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering, 14*(2), 131-164.

San Cristóbal, J. R., Carral, L., Diaz, E., Fraguela, J. A., & Iglesias, G. (2018). Complexity and project management: A general overview. *Complexity, 2018*, 10p.

Sawyer, P., Sommerville, I., & Viller, S. (1997). Requirements process improvement through the phased introduction of good practice. *Software Process: Improvement and Practice, 3*(1), 19-34.

Scherer, F. M. (2000). The pharmaceutical industry. In *Handbook of Health Economics* (Vol. 1, pp. 1297-1336).

Schwaber, K. (1997). Scrum development process. In *Business Object Design and Implementation: OOPSLA'95 Workshop Proceedings 16 October 1995, Austin, Texas*, Springer London, 117-134.

Shanks, G. (2000). A model of ERP project implementation. *Journal of Information Technology, 15*(4), 289-303.

Sommerville, I. (2005). Integrated requirements engineering: A tutorial. *IEEE Software, 22*(1), 16-23.

Sommerville, I. (2009). *Software engineering* (9th ed.). Addison-Wesley.

Stacey, R. D., Griffin, D., & Shaw, P. (2000). *Complexity and management: Fad or radical challenge to systems thinking?*Psychology Press.

Sudhakar, G. P. (2012). A model of critical success factors for software projects. *Journal of Enterprise Information Management, 25*(6), 537-558.

Takeuchi, H., & Nonaka, I. (1986). The new new product development game. *Harvard Business Review, 64*(1), 137-146.

Theunissen, T., van Heesch, U., & Avgeriou, P. (2022). A mapping study on documentation in continuous software development. *Information and Software Technology, 142*, 106733.

Tukel, O. I., & Rom, W. O. (1998). Analysis of the characteristics of projects in diverse industries. *Journal of Operations Management, 16*(1), 43-61.

Wysocki, W. (2023). Macrosimulation model of software development process. *Procedia Computer Science, 225*, 746-755.

Wysocki, W. (2020). A hybrid software processes management support model. *Procedia Computer Science, 176*, 2312-2321.

Wysocki, R. K. (2014). *Effective complex project management: An adaptive agile framework for delivering business value*. J. Ross Publishing.

Wysocki, R. K. (2019). *Effective project management: Traditional, agile, extreme, hybrid* (8th ed.). John Wiley & Sons.

Wysocki, R. K. (2011). *Effective project management: Traditional, agile, extreme* (6th ed.). John Wiley & Sons.

Zainal, P., Razali, D., & Mansor, Z. (2020). Team formation for agile software development: A review. *International Journal of Advanced Science, Engineering and Information Technology, 10*(2), 555-561.

Zhang, Z., & Sharifi, H. (2000). A methodology for achieving agility in manufacturing organisations. *International Journal of Operations & Production Management, 20*(4), 496-512.

Žužek, T., Kušar, J., Rihar, L., & Berlec, T. (2020). Agile-concurrent hybrid: A framework for concurrent product development using Scrum. *Concurrent Engineering, 28*(4), 255-264.