PROCESS FRAMEWORK FOR SOFTWARE MODERNIZATION DECISION-MAKING

Lappeenranta–Lahti University of Technology LUT

Master's thesis

2023

Oskari Arpinen

Examiners: Professor Kari Smolander

M.Sc. Leena Tanttu

ABSTRACT

Lappeenranta–Lahti University of Technology LUT

LUT School of Engineering Science

Software Engineering

Oskari Arpinen

Process framework for software modernization decision-making

Master's thesis

2023

70 pages, 13 figures and 21 tables

Examiners: Professor Kari Smolander and M.Sc. Leena Tanttu

Keywords: legacy system, modernization, decision making

Legacy systems are serving various crucial business operations, being in the core of industries like banking and healthcare. Legacy systems can have better performance than new systems and be proven to work by test, but there is still frequent need for modernization of these systems due to other more important qualities. Although legacy system modernization has been conducted ever since computer systems existed, it is estimated that over 50 % of system modernization projects fail. It is therefore important to understand software modernization methods, process models and risks associated with modernization activities, as well as create tools and methods to make the modernization activities more successful. The purpose of this thesis is to study the development, maintenance and use of legacy systems and tools in small engineering organizations. The issues found related to legacy systems are mapped and used as the basis for process framework, which aims to support easier modernization of these systems. The use of this process framework is demonstrated by applying to a case modernization project. Main findings from the interviews and literature were limited resources to conduct modernization, no formal evaluation methods in use to evaluate the need for modernization and difficulties in argumentation for modernization. Process framework was created as the design science artefact to address these issues. The process framework utilizes MODEST and AHP to provide a complete decision making process. The process model was designed to fit organizations who may not have extensive software engineering knowledge and have very limited resources allocated to software development activities. The use the artefact was demonstrated by applying it to a case modernization process provided by the Process and Testing organization at Metso Corporation. The artefact was evaluated, and it was concluded that it serves the needs that small to medium sized organizations could face during modernization decision making.

TIIVISTELMÄ

Lappeenrannan–Lahden teknillinen yliopisto LUT

LUT Teknis-luonnontieteellinen

Tietotekniikka

Oskari Arpinen

Prosessikehys perintöjärjestelmien modernisoinnin päätöksentekoon

Diplomityö

2023

70 sivua, 13 kuvaa and 21 taulukkoa

Tarkastajat: Professori Kari Smolander ja M.Sc. Leena Tanttu

Avainsanat: perintöjärjestelmät, modernisointi, päätöksenteko

Perintöjärjestelmät palvelevat monia eri käyttötarkoituksia ja ovat monen liiketoiminnan kuten finanssialan ja terveydenhuollon ydinjärjestelmiä. Perintöjärjestelmät voivat olla tehokkuudeltaan ja nopeudeltaan parempia kuin modernit järjestelmät, mutta niiden modernisointi on silti yleistä muiden laadullisten ominaisuuksien tai funtionaalisuuksien vuoksi. On arvioitu, että 50 % modernisointiprojekteista epäonnistuu. Tämän vuoksi modernisointiin käytettävien metodien, prosessimallien ja riskien tutkiminen on tärkeää, jotta uusia työkaluja ja metodeita voidaan luoda modernisoinnin tueksi. Tämän tutkielman tarkoituksena on selvittää perintöjärjestelmien kehitystä, ylläpitoa sekä käyttöä pienissä ja keskikokoisissa organisaatiossa. Löydettyjen ongelmien ja haasteiden pohjalta luodaan prosessikehys, joka tukee organisaatiota modernisaatioprojekteissa. Tämän prosessikehyksen käyttöä demostroidaan käyttämällä sitä malliprojektissa. Onglelmiksi todettiin kirjallisuuden ja haastattelujen perusteella rajalliset resurssit modernisoinnin suorittamiseen, muodollisten modernisointitarpeen analysiontimetodien puute, sekä haasteet argumentoinnissa modernisoinnin puolesta. Prosessimalli luotiin suunnittelutieteen artefaktina vastaamaan löydettyihin ongelmiin. Prosessimalli hyödyntää MODEST ja AHP malleja, jotta kokonaisvaltainen päätöksentekoprosessi voidaan suorittaa käyttämällä yhtä prosessia. Prosessimalli suunniteltiin sovellettavaksi organisaatiossa, jotka eivät omaa kattavaa ohjelmistotuotannon osaamista, sekä joilla on rajatut resurssit modernisaatioprojekteille. Artefaktin käyttöä demonstroitiin soveltamalla sitä malliprojektissa, jonka tarjosi Metson ”Process and Testing”-organisaatio. Artefakti arvioitiin ja todettiin sen soveltuvan pienten ja keskisuurten organisaatioden käyttöön. Prosessikehyksen todettiin vastaavan näiden organisaatioiden ongelmiin hyvin.

Table of contents

Abstract

1 Introduction 5
2 Context & research methods 7
2.1 Context of the research 7
2.1 Current state of the case system 9
2.2 Design science research 11
2.3 Research methods 13
3 Previous research 15
3.1 Legacy systems and modernization 15
3.1.1 Legacy systems 15
3.1.2 Legacy system modernization 17
3.1.3 Modernization decision making 18
3.2 Software maintenance & maintainability 21
3.2.1 Software life cycle 22
3.2.2 Software maintenance challenges and issues 23
3.2.3 Software maintenance in non-IT organizations 25
4 Problem identification and motivation 27
4.1 General problem description & organization interviews 27
4.2 Expert interview 31
4.3 Objectives of the solution 33
5 Modernization process framework 34
5.1 Existing solutions 34
5.2 Design & development 39
5.2.1 Modernization initiative 40
5.2.2 MODEST evaluation 41
5.2.3 Modernization argumentation 48
5.2.4 AHP for modernization approach 50
5.2.5 Modernization approaches 59
6 Artefact evaluation 63
6.1 FEDS evaluation 63
6.2 Evaluation of the modernization process framework 65
7 Discussion 67
8 Summary 70
References

1 Introduction

Legacy systems are serving various crucial business operations, being in the core of industries like banking and healthcare. Industry professionals see legacy systems as business-critical core systems, and legacy systems are often proven to work by decades of use (Khadka et al., 2014). Legacy systems can have better performance than new systems but there is still frequent need for modernization of these systems due to other more important qualities. Need for modernization can be multifaceted, but common drivers include integration issues, cost of maintenance, knowledge issues and error proneness of the legacy systems. (Agilar et al., 2016) Although legacy system modernization has been conducted ever since computer systems existed, it is estimated that over 50 % of system modernization projects fail. One of the main reasons for failure has been observed to be unclear goals (Sneed, 2005). It is therefore important to understand software modernization methods, process models and risks associated with modernization activities, as well as create tools and methods to make the modernization activities more successful.

Studies of legacy system modernization often focus on large enterprise-size systems. This is the reason why processes suggested for modernization are usually designed to serve the needs of a large project (OMG, 2003. Pérez-Castillo et al., 2012. Assunção et al., 2022). Although these processes are fully capable and functional, small to medium sized organizations have no need or resources to use such complicated processes. Another important aspect of modernization is the decision making process, where modernization of the system is analyzed, planned and suitable approach is selected. When it comes to the decision making process of modernization, existing literature provides various methods, but there are no light processes suggested that could be used to complete the whole decision making process.

The purpose of this thesis is to study the development, maintenance and use of legacy systems and tools in small engineering organizations. The issues found related to legacy systems are mapped and used as the basis for process framework, which aims to support easier modernization of these systems. The use of this process framework will be demonstrated by applying to a case modernization project. Three organizations are interviewed to discover the issues and challenges that are caused by legacy systems, along

with analysis of previous work. After discovering the challenges and issues, design science artefact is created to address these issues. The use of the artefact is then demonstrated by applying it to the case project provided by Process and Testing organization at Metso Corporation.

The structure of this thesis follows the structure of design science research. Context for the research is presented in chapter 2, along with the research methods used in this thesis. Chapter 3 describes the previous research relevant to the topic of this thesis, including legacy systems, modernization of legacy systems, and software maintenance. Chapter 4 describes the issues and challenges in the organizations using legacy systems. In chapter 5, the design science artefact is created, described, and demonstrated by applying it to the case modernization project and the artefact is evaluated in chapter 6. The results, limitations, and future around the topic of this thesis is discussed in chapter 7, and the thesis is concluded in chapter 7.

2 Context & research methods

This section describes the context of this research, as well as the research methods that are used to achieve them.

2.1 Context of the research

This thesis was created in collaboration with Metso Corporation. Metso offers products, services, and end to end solutions for metals refining, aggregates, and minerals. Metso's portfolio includes equipment from the crushing stage of ore to the electrowinning of pure copper. (Metso, 2023) This thesis was written specifically with Process and Testing organization at Lappeenranta. The organization is responsible for testing the samples of customers with Metso's equipment and providing process calculations and product configurations, which are used to create a quote for the customer.

During the research presented in this thesis, the use of legacy engineering systems and tools in multiple organizations at Metso is studied. The purpose of this is to explore the general information about the legacy systems that are currently in use, as well as issues and problems that the systems possibly cause. In addition to this, reasons behind the continuation of legacy software usage are studied. Based on the findings, a process framework is suggested, that supports the modernization process of legacy engineering tools and systems. In the context of this thesis, engineering tools are small to medium sized desktop applications, that are used to assist for example in process flow calculations, quotation processes and product configuration tasks. Large and complex applications such as software built for process simulation are not considered during this thesis. General rule of thumb that can be used is that the legacy engineering software included in this study is usually built and maintained by the engineers that use it themselves.

The creation of the process framework is based on the issues or challenges discovered during interviews, meetings, and surveys working at Metso in different roles. Professionals across multiple organizations are interviewed about the use and maintenance of legacy tools. This way more complete view on the state of legacy system modernization can be gathered. The

objectives that are set to the process framework will be defined so that they address the most common issues found in both the legacy maintenance and modernization process as well as technical issues related to the legacy systems themselves.

The suggested process framework is demonstrated and tested by applying it to a case modernization project. The system modernized is used for product sizing of industrial filters. Industrial filters are used for separating solid particles from liquid in different processes, such as metal refining, mining, and various process chemistry applications. Example of an industrial filter is presented in Figure 1. Sizing includes process calculations and equipment configuration. Sizing is part of the quotation process, where combination of equipment and services is quoted to the customer based on the needs of the process and other requirements by the customer. The sizing is based on process requirements, as well as data gathered during dewatering testing. Test results obtained from laboratory or pilot scale testing along with the requirements that the customer has set are used to select and quote the suitable equipment for the customer.

The sizing program can be used to size different filters, thickeners, and the auxiliary equipment for these, such as tanks and pumps. The inputs to the sizing process in filtration are details about the slurry, details about optimal cycle, product data, and process requirements. Slurry is feed material to the filtration process, which is a mixture of suspended solids and liquid. Slurry has relevant properties that effect sizing, such as density, particle size distribution, and solids content. The information about optimal cycle time is specific to a certain slurry and filtration technology. Details about optimal cycle are obtained through the testing process. Filtration capacity is calculated using the test results and used in the sizing process. Product data is process-related information about the filtration products. Product data that is used in sizing depends on the equipment in question, but can include things like total filtration area, operational parameters of the filter such as waiting time and static head for pumps.

Figure 1 Larox PF60 pressure filter produced by Metso (Metso, 2023).

2.2 Current state of the case system

The current sizing program consists of three types of components, as seen in Figure 2. Macro-enabled Excel sheets are uses as the user interface. Four different sheets are used for different equipment types. These Excel files also contain formulas that are utilized in the sizing process, for example the calculation of required filtration area. In other words, the calculations are done on the user interface layer. The Microsoft Access database serves as the data layer. It contains product data that is required to complete the sizing process. This Access database is connected to the Excel sheets via macros written in Visual Basic for Applications (VBA). These macros also connect the Excel sheets into each other, which allows for transfer of data between the two sheets.

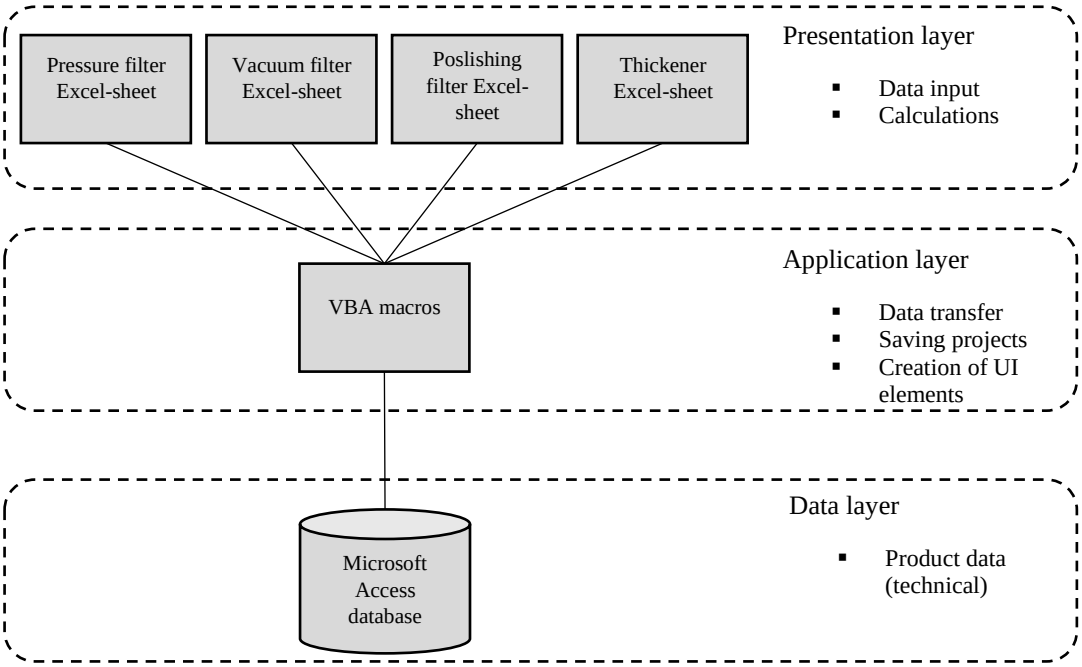


Figure 2 High-level architecture of the current sizing program

Sizing program is running locally on the user's machine, without any connections to other machines. The components described earlier are stored inside one folder, which the user has downloaded from a Teams channel created for sizing tool distribution. When new version of the sizing program is released, the users of the Teams channel are notified. It is therefore the user's responsibility to make sure that the version that they are using is the recent one. According to one of the developers, there have been multiple incidents, where users have been using old version of the sizing program. This has caused issues with the file format that is used to share the created sizing to other engineers. Additionally, the developers of the sizing program have no control over which version of the software is in use. Local copies of the software also allow for unwanted modifications to the software, such as modifications to the calculation formulas stored inside the Excel cells. The problems identified in the sizing program as well as other legacy software and systems found in other organizations are discussed more in depth in chapter 4.

2.3 Design science research

Design science was chosen as the research methodology of this thesis. According to Hevner et al., there are two primary paradigms in information systems research: behavioural science and design science (DS). While behavioural science focuses on finding and verifying theories about how humans and organizations operate, DS develops artefacts to extend the capability of humans and organizations. Design science originates from traditional engineering, where problem solving is at the core of the research methodology. DS research can be described as creation and evaluation of artefacts, which are used solve some identified problems. These artefacts can be various methods, models, and implementations but ultimately, artefacts can be any type of designed object which solves the identified problem. (Hevner, 2004) The artefact needs to be defined well and described in detail to allow for implementation and evaluation in the correct domain. According to Hevner, artefacts are a combination of the environment they are created into, and the constructs, models and methods defined in these contexts. In other words, the artefacts are also connected to the social and organizational structures that they are created into. (2004)

Peffers et al. (2007) have defined DS research methodology for information systems. In their paper they introduced the DS research process model which is depicted in Figure 3. The DS process model is divided into 6 distinct steps, which are completed in a sequence. Although the process seems to follow some form of waterfall methodology, it is suggested that the artefact is developed iteratively, as shown in Figure 3.

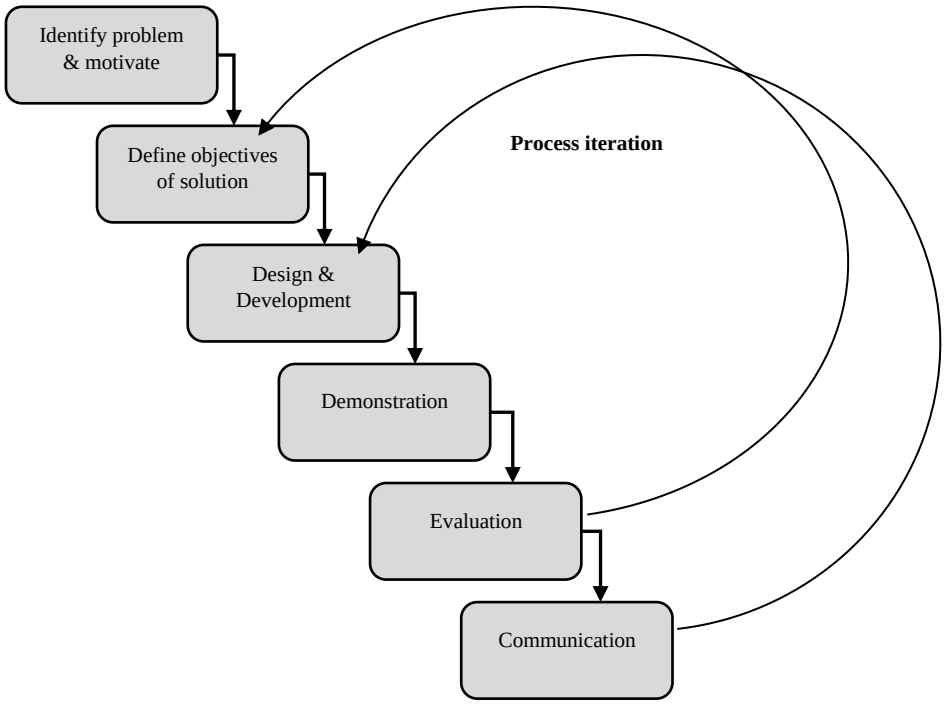


Figure 3. Design science process model (Peffers et al., 2007 (modified))

The six steps of DS research process model are as follows:

1. **Identify problem and motivate**: During the first phase, research problem is identified and defined. Additionally, the value of the solution is justified. During this phase questions like why modernization process framework is needed, what justifies the research on this topic and what value would the solution bring are answered. This phase requires knowledge of the current state of the problem and importance of solution.

2. **Define objectives of solution**: The objectives that are defined during this phase describe the desired qualities and features of the solution. These objectives are based on the problem definition and possible limitations. This phase requires knowledge of possible current solutions to the problem, and how well do these solutions work.

3. **Design & development**: During this phase of the process, the actual artefact is created. In practice, the artefact can be any designed object, which embeds the research contribution relevant to the identified problem.

4. **Demonstration**: The use of the artefact is demonstrated during this phase, meaning that the artefact is used in practice to solve the identified problem. Some examples of demonstration techniques are case studies, simulations, or experiments.

5. **Evaluation**: In the evaluation phase the artefact is evaluated based on how well it supports solution to the problem. In practice, the evaluation compares the objectives of the solution defined earlier to the ones achieved during the demonstration phase. Evaluation can be done by utilizing any empirical methods or logical proofing. After the evaluation, researchers can choose to return to phase 2 or 3 and iteratively create a new artefact or move on to phase 6.

6. **Communication**: Communication phase is for informing other about the problem and solution that was achieved. In this thesis, the communication part is the written thesis published. (Peffers et al., 2007)

In this thesis the design research process is started by defining the problem in chapter 4 by interviewing people across multiple organizations. During these interviews, problems and challenges in software maintenance and development especially related to legacy systems are assessed in these organizations. In addition, experts of individual legacy systems are interviewed, in order to understand the issues that the individual legacy systems have or cause. After the problems have been identified, objectives for the artefacts are created to best address them. In chapter 5, modernization process framework is developed as the artefact, which aims to solve the issues these organizations face. The use of this framework is demonstrated by applying it to case modernization project. Finally, the effectiveness of this artefact is evaluated, and the results are communicated via this thesis.

2.4 Research methods

Multiple research methods are used during this thesis. The research methods utilized are described in Table 1. Literature reviews are conducted informally. Literature review is utilized on two separate occasions: first to form understanding on topics related to this thesis and secondly to discover existing solutions to fit the design goal of this thesis. Interviews are conducted to gather qualitative information about legacy systems and tools used. Interviews are utilized on two separate occasions during this thesis. On the first occasion,

three organizations are interviewed on separate occasions. These interviews are described in chapter 4. Second occasion where interview is utilized is the expert interview. This interview was conducted once.

Table 1 Research methods used in this thesis.

Method	Used in	Description
Literature review	Previous work (chapter 3) Discovery of existing solutions (chapter 5)	Informal literature review, which is utilized to form understanding of the topics relevant to this thesis work. In addition to this, part of the design science research is to discover existing solutions to the design problem, which is done using literature review.
Interview	Problem identification (chapter 4) Expert interview (chapter 4)	The interviews conducted during this thesis are informal. The participants of these interviews included either product owner or developer of the legacy tool, person who is expert of the processes conducted in the organization, and the interviewer. For the expert interview, interviewer and expert are present. All the interviews are conducted via Teams.

3 Previous research

In this chapter, informal review of previous work related to topics relevant to this thesis is conducted. The topics presented in this chapter give context for the rest of this thesis work.

3.1 Legacy systems and modernization

Legacy system modernization or migration is the activity of replacing the old legacy systems with new systems partly or as a whole to reduce maintenance complexity and costs. These modernization activities are very common IT activities in organizations, where legacy systems serve critical business purposes. (Abdellatif, 2021) Although modernisation is a very common activity, it is estimated that over 50 % of system modernization projects fail. One of the main reasons for failure has been observed to be unclear goals (Sneed, 2005). It is therefore important to understand software modernization methods, process models and risks associated with modernization activities.

3.1.1 Legacy systems

The Merriam-Webster dictionary defines legacy as “of, relating to, or being a previous or outdated computer system” (Merriam-Webster, 2023). Legacy systems are typically large applications with millions of lines of code, older than decade, written in legacy languages such as COBOL, using outdated database management services and acting on its own without many interfaces to other applications. (Brodie & Stonebraker, 1995) While this definition is very specific, legacy system can also be described as “an existing application system in operation”. (Langer, 2016, p. 179) Most industry professionals (76.7 %) see legacy systems as business critical core systems. (Khadka et al., 2014). Since the use of computer systems in enterprises started over 70 years ago, many of the earlier systems have been replaced, some more than once. There still remains a large number of legacy systems that are critical to the business operations of organizations. Legacy systems are not just the software, but broader collections of hardware, supporting software, application software,

application data, business processes and policies. (Sommerville, 2016) Relationships between these components are showed in Figure 4.

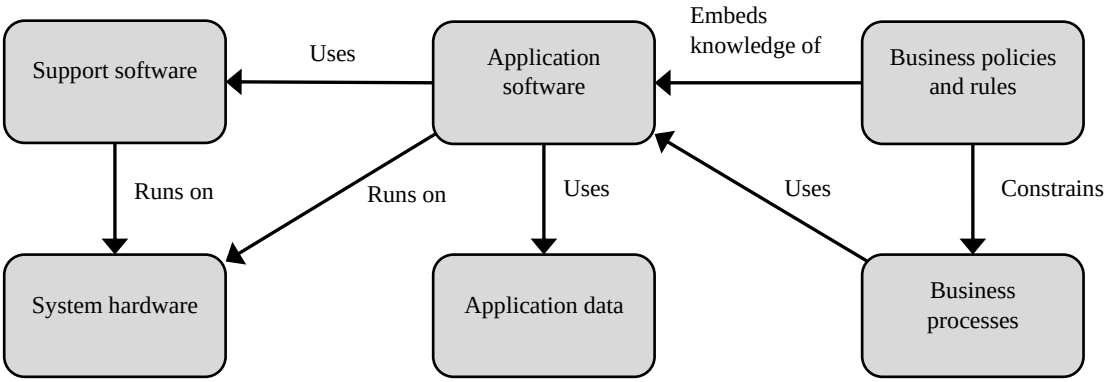


Figure 4. Legacy system elements (Sommerville, 2016)

Legacy systems were studied by Khadka et al. in their exploratory study. The researchers interviewed 26 practitioners from the software development industry about perceptions of legacy systems and the modernization activities related to them. They also identified benefits of legacy systems, modernization drivers and compared the perception of industry professionals and academic researchers. The general consensus within industry practitioners was found to be that legacy systems are old systems that serve core business purposes. Legacy systems have often also been proven to work by decades of use. In addition to business criticality, reliability, proven technology, and performance are also seen as benefits of legacy systems. The perceptions of the industry practitioners were compared to the ones from academia. While practitioners see legacy systems as core technology, academic researchers often seen legacy systems as obsolete. Practitioners also see legacy systems as reliable and proven technology, which are not descriptions used in the academic context. (Khadka et al., 2014)

Although many different definitions for legacy systems as a concept exists, legacy systems can be generalized to be systems developed using technologies and languages that are not used to develop new systems anymore. Since legacy software is usually written in a language that is no longer used to develop new systems, it might be difficult to find staff that is capable of maintaining software written for example with COBOL. Various other risks are associated with legacy systems such as information security issues, knowledge issues described earlier and hardware becoming obsolete. (Sommerville, 2016) In addition to these, all the maintenance related challenges and issues are present with legacy systems.

3.1.2 Legacy system modernization

Agilar et al. conducted a systematic mapping study, in which they explored the existing literature related to legacy system modernization. They studied the characteristics of legacy system modernization, tools and processes used during modernization as well as reasons why organization modernize their systems. General consensus among research is that modernization can be defined as "the evolution of systems towards new business requirements of the organizations, involving new functionalities, error correction, or technological updates" (Agilar et al., 2016, p. 3). Modernization can be seen part of the software life cycle, but different researchers have different definitions of the software life cycle itself, as described in chapter 3.2.1. In general, modernization is seen as that phase in the software life cycle, either by being its own stage or part of the evolution stage of the system. The researchers discovered that 64 % of academic contributions provide some solution proposal, while 70 % of the publications address modernization management. Modernization management contains the activities of managing the software lifecycle, strategy for software modernization in the organization and process models for modernization. Additionally, 74 % of the publications discuss white-box modernization, 24 % black-box modernization and 3 % are about replacement of the legacy systems. White-box modernization includes the techniques, where source code of the system and knowledge about the internals of the system is utilized in the modernization, while in black-box modernization there is no access to source code or extensive knowledge about the system internals. Reasons for modernization were identified to be issues with integration, cost of maintenance, knowledge issues and error proneness of the legacy systems. (Agilar et al., 2016)

Khadka et al. studied the perceptions industry professionals have of the reasons for legacy system modernization as well as the challenges that are faced when modernizing. The most common reason for modernization was identified to be the need to become flexible. Organizations want to be agile in order to adapt to changes caused by the dynamic nature of the current business environment. The second most frequent modernization driver was identified to be the cost of maintenance, followed by lack of knowledge meaning lack of experts, documentation, or both. When it comes to challenges faced during the modernization projects, time constraints are seen as the most challenging part. Other

challenges related to project management were also identified, including funding the project and resistance to modernize from staff. Technical challenges faced include data migration, difficulties in testing, business logic extraction and issues related to architecture. (Khadka et al., 2014)

Sommerville states that there are four approaches that organization select from when dealing with legacy systems and the modernization of them. The first option is to leave the system as it is and continue maintaining it. Second option is that organizations decide to stop maintaining and using the system in the case that the system has stopped to support the business process it was originally part of. Third approach to modernization is replacement, where totally new system is developed to replace the legacy system. This can be the case for example when organization want to implement Software as a service (SaaS) solution provided by some third party. Another reason why new system would need be developed is large changes in the business process and requirements of the system. Replacing the system can be done in phases, where the components of the system are replaced one at a time. The fourth option organization have is the re-engineering approach. (Sommerville, 2016) Which ever approach organizations select, some decision making process or method is needed in order to make the decision.

3.1.3 Modernization decision making

The decision making of what to modernize and when is critical part of the legacy software modernization process, since modernization often means large changes to the system. The changes to systems during modernization can include changes to the underlying technologies and business rules, or re-creation of the whole system or part of it, which makes the changes possibly the hardest during the system life cycle. This is why the decision to modernize software system needs to be well argued, while taking the related risks into account. (Koskinen et al., 2005) Sommerville provides multiple risks related to legacy system replacements:

1. lack of specification in form of documentations,
2. business processes entangled into the legacy system can be hard to translate into the new system,

3. some business rules might be present in the legacy system without being documented,

4. and developing new software is inherently risky. (Sommerville, 2016)

The decision making of modernization has been empirically studied. Koskinen et al interviewed 29 professionals in the software engineering industry about the software modernization decision making process and practices. Questions asked were categorized into seven groups. decision makers, decision making process, methods, tools, confirmation of decisions, decision criteria and relevance of the questions asked. It was discovered that the modernization decision is always done by the software customer, although it is done in co-operation with the supplier, with experts from both the customer and supplier organizations. The decision making process of modernization is rarely specified, apart from public administration projects, but experts still see that the quality of the decisions is good. There are various sources for the initiatives, such as technologies becoming obsolete, vendors suggesting modernizations, business processes changing, and suggestions from end users. The time spent on modernization decisions is seen insufficient, but it is not identified as an issue. It was discovered that decisions are sometimes based on intuition, although this question was very polarizing. Quantitative data used in decision making includes various cost estimations, such as return on investment, total cost of ownership, total cost estimate. Additionally, information about possible benefits, LOC and savings in employee-work-years were identified as quantitative data used in the decision making process. The quantification of possible modernization benefits was seen important, but not something that many experts had seen completed successfully. No tools for decision making are use, although majority of experts supports the usage of them. When asked to describe tool that could be useful, the experts could not describe one clearly, but the quantification of benefits was mentioned. (Koskinen et al. 2005)

Koskinen et al. provide a collection of software evolution decision making processes and methods, which is presented in Table 2. They present the VERDE decision making process, which can be utilized to support the selection of evolution actions taken. Software evolution strategies are strongly related to modernization activities, and the need for modernization can be evaluated using the same methods. There are various process models that can be used, and the selection of which processes and methods to select should be based on priorities and needs of the organization in question. (2008)

Table 2 Methods for evaluating software evolution, plus-sign means strength, minus-sigh weakness. (Koskinen et al., 2008)

Method	Purpose	Strengths and weaknesses
VERDE	Reveal relevant issues of system evolution decision making	+ Supports decision quality evaluation and improvement + Easiness of use, + Industrial needs addressed +/- Abstract-level framework
MODEST	Evaluate modernization pressure	+ Supports early estimation + Easiness of use + Industrial needs addressed - No refereed publications yet
VDM	Strategies regarding renewal of legacy systems	+ Supports component-level analysis
Bergey et al.	Risks of system reengineering	+ Easiness of application + Considers risks - No refereed publications yet
Sahin & Zahedi	Strategies regarding optimal maintenance service level	+ Clear model with exact thresholds for actions +/- Focuses on customer satisfaction effects - Lacks empirical validation in industrial settings
SABA	Strategies regarding transition of legacy systems	+ Covers both technical and organizational analysis
Renaissance	Strategies regarding evolution of legacy systems	+ Very detailed diagnostic criteria
ISEBA	Benefits of information system evolution and their measures	+ Considers benefits + Supports selection of proper benefit estimation methods +/- Abstract-level framework
Sneed	Profitability of reengineering	+ Costs are considered in detail + Benefits are covered - Requires much versatile and reliable input data - Lacks systematic empirical validation
Softcalc	Costs and effort of maintenance tasks	+ Considers costs - Lacks systematic empirical validation

3.2 Software maintenance & maintainability

IEEE standard 1219 defines software maintenance as “Modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment.” (IEEE, 1993). IEEE also defines that software maintenance refers to the modifications made to the software after it has been delivered to the customer. The scope of maintenance is to keep the software operational in its production environment, while appending and modifying the feature set and quality attributes of the software. (Marounek, 2012) Maintenance can be adaptive, corrective perfective or preventive. Adaptive maintenance is done when there are changes in the environment that the software operates in. Perfective maintenance is done when the requirements of the users change. Corrective maintenance is fixing defects in the software, preventive maintenance is done to prevent the defects or problems occurring in the first place. (Bennett & Rajlich, 2000)

Software maintenance is widely researched topic within the software engineering discipline. This can be explained by the fact that maintenance can account for 90 % of total cost during software’s life cycle (Koskinen, 2015) depending on the source. This percentage of maintenance costs in software projects has increased roughly by 50 % over the last two decades. There are multitude of factors that affect the cost of maintenance, such as insufficient documentation, low application experience or just individual product characteristics that make the software costly to maintain. (Dehaghani & Hajrahimi 2013) One of the factors making software difficult and thus expensive to maintain is the underlying architecture. Since these architectural decisions are usually done early in the software development process, many authors disagree with the IEEE: s definition of maintenance beginning after the software product is delivered. According to Marounek, maintenance begins very early in the development process, with something called pre-delivery tasks. These pre-delivery maintenance tasks include planning for post-delivery and designing for maintainability. Post-delivery tasks are those that most perceive as being the whole maintenance process, like software modifications. (Marounek, 2012) Especially the pre-delivery tasks can influence the overall maintainability of the software greatly.

According to ISO standard 9126-1 (2001), software maintainability is one of the six quality characteristics of software products. Maintainability describes how difficult it is to make

maintenance modifications to the software. The standard also defines multiple sub-categories of maintainability, which are analysability, changeability, stability, testability, and maintainability compliance. Many factors during the development phase contribute to the maintainability of software products. Chen & Huang conducted a literature survey, during which they discovered 5 different problem dimensions in software development, that deteriorate the maintainability of software. These problem dimensions are documentation quality, programming quality, system requirements, personnel resources, and process management problems. Documentation and programming quality problems were identified as the principal problem dimensions affecting software maintainability. (2009)

3.2.1 Software life cycle

The software life cycle can be divided up to in any way depending on the source. Rajlich & Bennett provide five stages for software life cycle, which are:

1. initial development,
2. evolution,
3. servicing,
4. phaseout,
5. and closedown (2000).

During initial development, the first operating version of the software is developed. Some core things about the system are defined in this phase, like the main architecture. During evolution, the software system goes through iterations, where the functionality is added, removed, or modified. The feature set of the application is evolved further, making the system more capable. Once the software reaches servicing stage, it is modified only to correct defects, or due to some small changes to the functionality. This stage considered to be the legacy stage of the software. Software systems enter this stage of the life cycle when developers who are experienced leave the project and the architecture of the system deteriorates. Changes made during servicing stage are expensive and difficult to complete, and further deteriorates the systems architecture. During phaseout, the software does not receive any more maintenance, but the system is still available for use. In the final stage

closedown, the system is discontinued, and it is not served for the users. Users are directed into some other service if one is available. (Rajlich & Bennet, 2000)

This life cycle model is rather old and does not align so well with the modern agile software development methodologies, where the initial development, evolution and servicing are merging into one continuous stage, where companies implement continuous integration and delivery process models. Weiderman et al. suggest a model that consists of three stages: maintenance, modernization, and replacement. The maintenance stage consists of small changes to the system, which are mainly done to fix bugs or add new features. During modernization phase, the changes with largest impact are made. These changes are modifications due to changes in business processes, migration of the system to another technology or platform, and major changes in the systems architecture. Legacy systems enter the replacement stage when the resistance for change in the system becomes too strong. During this stage new system needs to be re-created in order to for the system to satisfy the requirements set by the business process. (Weideman et al., 1997)

3.2.2 Software maintenance challenges and issues

There are various challenges and issues organizations conducting software maintenance can face. Figure 5 presents challenges that organizations face during software maintenance. The issues presented were collected by analyzing studies related to software maintenance. The challenges have been categorized into management, knowledge, and software issues.

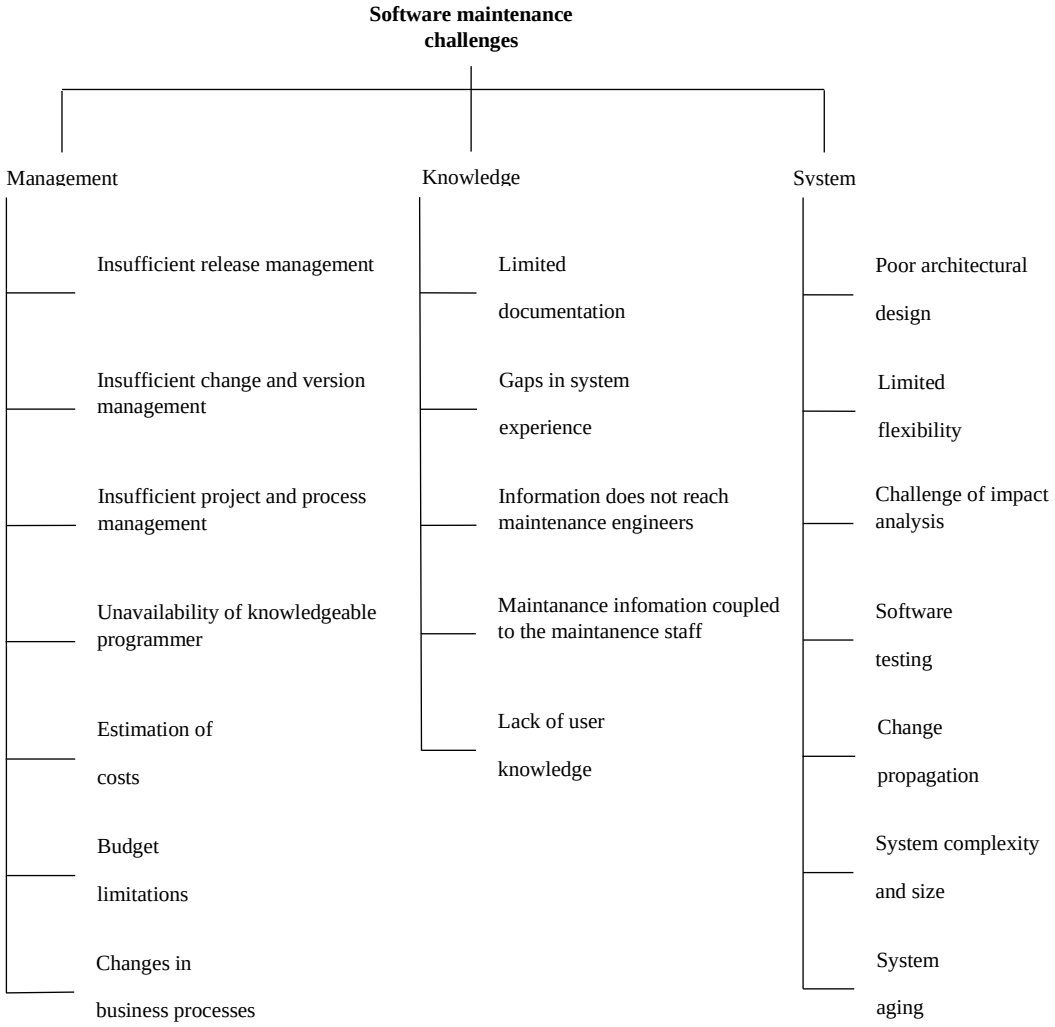


Figure 5. Challenges faced in software maintenance (Sharon, 1996; Khanna et al., 2017; Kaur & Singh, 2015).

Software maintenance can be very knowledge intensive process, which is why the maintenance process can be described as a knowledge management issue. This is especially true with legacy systems, where the maintenance of the software has been conducted by the same person or group for the whole life cycle of the software. These people have become so called "immortals", who are critical to the system to stay operational, since they are the only ones holding the important knowledge about the system and how to maintain it. This information rarely ends up in any written form and stays inside the someone's head. (Anquetil et al., 2007) Knowledge issues are therefore natural occurrence, when people eventually move on from software projects.

3.2.3 Software maintenance in non-IT organizations

It is very common for engineering organizations to work with legacy software that is decades old and maintained by the engineers themselves. The engineers responsible for developing and maintaining this software are not usually formally trained software engineers. Additionally, these organizations commonly have a very limited budget for software development and maintenance. (Harrison & Walton, 2002) Since these companies are commonly operating with low resources allocated for software maintenance, it might not be possible to outsource the software maintenance.

If organizations lack the resources of conducting maintenance, they have little options apart from outsourcing the work. In this context, outsourcing is the activity of having a third party organization do the IT activities for the organization. Outsourcing can include management of hardware such as servers, software maintenance work, IT support and development of new software systems. It is common and efficient for companies to focus their efforts to their main business and core competencies, which has led to outsourcing being popular. Long distances and limited communication channels used to be the limiting factor for software outsourcing work historically, but as the era of the Internet began, global outsourcing became possible. There could be many reasons to outsource maintenance, such as cost savings, knowledgeable people leaving the organization or to be more efficient with the use of the employee's time. One of the most common reasons for organizations to outsource maintenance work is to increase operational performance. (Bhatt, 2006)

Outsourcing is a way that organizations might end up with knowledge issues. Sometimes it is not possible or desired that software is maintained in the same organization it was developed in. Ahmed identifies issues and risks faced by software organizations outsourcing maintenance work, but the same apply to non-IT organizations, who need third-party organizations to maintain the tools that they use, such as engineering organizations (Ahmed, 2006). Common maintenance outsourcing risks are presented in Table 3.

Table 3. Risks related to outsourcing maintenance activities. (Ahmed, 2006)

Risk	Description
Cost and time	Additional and unexpected costs might be created due to wrong selection of company, issues due to cultural differences or language barrier.
Data privacy & security	Outsourcing maintenance activities to third party organizations might expose the organization to additional data privacy and security concerns.
Intellectual property rights	It is harder for organizations to make sure their intellectual property rights are not violated, when sharing the source code of their products to third party organizations.
Quality	Quality might be harder to achieve and monitor when the work is done outside the company. Additionally, quality might be hard to specify in the contract. The components supplied by outsourcing organizations might be of low quality.
Portability	Problems might occur in portability when utilizing outsourced maintenance components.
Loss of control	Organization might lose control over the development of certain parts of their products.
Outsourcing organization stability	The organizations conducting the work might face financial, technical or other issues, deteriorating the performance of the outsourcing organization.
Vague contract	The requirements of the maintenance might be difficult to capture in the contract. There is that the contract is too vague, in which case the required quality might not be achieved.

4 Problem identification and motivation

The purpose of this research phase is to identify the research problem. In the following sections the use, maintenance, and development of legacy applications is studied across multiple organizations at Metso. Organizations using legacy software and tools are identified, and the managers and developers of these legacy applications are interviewed when possible. The purpose of the interviews is to discover information about the business processes the tools are used in, what kind of tasks these legacy applications are used for and what the typical legacy application in engineering organization looks like. In addition to this, the process of maintaining and developing these applications is studied. Some important topics discussed with the developers and managers are the process of software maintenance and issues faced in it, software evolution process and challenges, version control of the tools, and previous modernization initiatives related to the tool.

4.1 General problem description & organization interviews

The use of legacy engineering tools was studied in three different organizations at Metso. The use of legacy tools was surveyed through email, by sending short questionnaire to five organizations. Three out of these organizations were selected to be interviewed. 1 hour meetings with either the manager or the developer of the tools was arranged. During these meetings, business processes of the organization was shortly introduced, after which the legacy tools used during their processes were demonstrated. Although the meetings were mostly unstructured, there were some common topics that the discussion was guided towards. Themes discussed during these meetings were general information about the tool such as user base and system age, the business process the tool is used in, maintenance process and workflow and evolution of the system. The organizations that were interviewed are described in Table 4.

Table 4 Organization interviewed for the purpose of problem description.

Organization	Organization description	Interview participants
Organization 1	Organization 1 configures hydrometallurgical equipment and plants based on the needs of the customer. The organization also provides process support to customers, including process calculations for singular equipment or for whole plants. The organization consists of 13 people, who mainly product engineers with technical knowledge about certain pieces of equipment.	Main developer, organization manager, interviewer
Organization 2	Organization 2 is responsible for testing and configuring pumps based on the needs of the customers. They also provide cross-organization support related to pumps. The organization consists of 7 people.	Main developer, interviewer
Organization 3 (case organization)	Organization 3 is the case organization that the artefact will be implemented in. The organization is responsible for conducting testing with thickeners and filters using sample from the customer. Based on this testing, the organization provides equipment configurations to be used in quotation.	Main developer, organization manager, interviewer

Organization 1 utilizes mainly Excel workbooks for their quotation process. One of the main tools contains the workflow from process calculations to the creation of itemized quotation to the customer. This Excel workbook contains all the data about their products, including product pricing information. There are also multiple other smaller tools used during the quotation process, that supplement the main workbook. The maintenance of these tools is conducted by one person. During the meeting it was identified that these Excel workbooks are seen as a security issue as they contain a lot of confidential information, such as product price data and process calculations that are seen as intellectual property of Metso. These tools are only used by 5-6 people, which is partly due to the sensitive data stored in it. The organization mentioned that they would like to allow more users to access the tool, but don't want to share the version with confidential data with more people. Lack of resources to develop new tool is seen as the reason that one has not been developed. The organization identifies the need for modernization but does not see modernization feasible with so small user base. The organization has been exploring off-the-shelve options available for their process.

The second organization had been using a legacy software for product configuration as a part of their quotation process. The tool was used to make some basic process calculations based on input data, and the user was able to configure the correct equipment with help from the system. The tool was created in Excel at the start but was implemented in Visual Basic eventually. This implementation of the system was maintained and used for 20 years after the initial development. The maintenance of the legacy software was conducted by two people. The system was used by 20 – 30 people. New versions of the software were distributed through email, and users were responsible for installing the patches. During the 20 years that the tool was in use, there were multiple attempts to start modernization projects. There were various sources of initiative, but the organization had difficulties to get funding for the modernization project. Eventually in 2019, the organization was able to start the development of new tool. The organization facilitated multiple workshops, where various stakeholders were invited. During the workshops the stakeholders of the system expressed their needs from the system, which included various new features. The outcome of these workshops was digital roadmap for the whole organization as well as secured funding for the tool modernization. This was achieved by productizing the tool, so that it could also be sold to customers. In the end, the organization launched the new replacement tool, and it is currently used by over 500 users.

Third organization is using a legacy tool to conduct process calculations related to their quotation process. The tool is used to assist in process calculations and selection of the correct equipment based on the customer and testing data that the user inputs to the tool. The tool consists of Excel workbooks as user interfaces, macros written in Visual Basic for Applications for different data manipulations and a Microsoft Access database where product data related to the process is stored. The tool has been in use for approximately 30 years now. The userbase of the tool is approximately 50 people, estimating based on the number of people on the Teams channel which is used for software update distribution. The organization has mentioned problems in version control of the software. There have been cases where users have used old version of the software, and the calculations created have not been forward-compatible. Maintenance of the tool is conducted by one person internally who is responsible for the calculations implementing business logic, and one external contractor who is responsible for editing the macros. The maintenance process is seen slow, with sometimes slow response times from the external contractor. Modernization need for

the tool has not been evaluated formally, but the need for modernization is recognized inside the organization.

During the interviews, it was discovered that all of the organization had some need and identified issues that motivate to modernize their systems. All three organizations have rather small resources allocated to the maintenance and developing activities of the legacy systems. This can cause situation where the organization has not time to allocate resources to evaluate the need for modernization. Since the resources used for the legacy tool maintenance are small, funding for the modernization project can be hard to argument for. The interviews indicate that the organizations using legacy systems face various challenges and issues during the maintenance and use of legacy systems. Both organizations currently using legacy tools identify some form of security issues related to their data. Knowledge management issues were identified in all organizations, since the maintenance of tools is dependent on one or two persons, while organization 3 utilizes external contractor to conduct the maintenance. Software version control is also seen as a challenge. The problems identified during the organization interviews are presented in Table 5.

Table 5. Problems identified from organization interviews.

ID	Problem	Description
ORG-01	Lack of modernization resources	Organization 1 mentioned the lack of resources as the reason they have chosen not to modernize their tools. All organizations have 1 – 2 people working on the systems
ORG-02	Modernization project funding	Organization 2 could not secure funding for modernization project in 20 years, despite multiple modernization initiatives during this time
ORG-03	Data security	Both in organization 1 and 3, product & price data was stored either directly available to the user on Excel workbooks or stored in local Microsoft Access database. This method of storing the data can allow for distribution of the data to unwanted parties.
ORG-04	Modernization argumentation	Argumentation can be used to justify modernization decisions, but meaningful and quantifiable arguments are difficult to provide.
ORG-05	Version control (software)	Organizations with legacy software lack necessary means to control the version of the software that is being used.
ORG-06	Knowledge management	All three organizations suffer from some form of knowledge management issues. The maintenance of the systems is conducted by one or two people, which makes the systems dependant on these people.
ORG-07	Decision making	The modernization decision is not evaluated frequently, and if the need for modernization is evaluated it is based on intuition
ORG-08	Evaluation of modernization need	The need for modernization is not evaluated using formal methods

4.2 Expert interview

User and former developer of the organization 3 was interviewed, in order to acquire more specific information about the issues related to the case modernization project. The themes discussed during the interviews were background and role of the interviewee, corrective maintenance process, the process of adding new functionality, version control of the system and modernization of the tool.

The interviewee is a user of the filter sizing tool with approximately 30 years of experience using the tool. They had been involved with the sizing tool since the start of development and were part of the development process as a supporting member and an advisor. During the time they were using the sizing tool daily, they would often make modification suggestions to the software that they would send to the developers to be implemented

officially. When it comes to corrective maintenance and fixing bugs, the user usually just fixes the defect on their copy of the software. Since the sizing tool is an Excel file that each user downloads to their physical machine, it allows the users to directly modify the software. If the defect is something that needs to be done by the external contractor, it is seen as long process. According to the user, defects are collected and sent to the developer periodically, which causes the long wait times. Adding new functionality to the software is seen as easy process. Excel as a platform is seen very flexible, and it is not seen as limiting technology. Version control is seen as an issue both for the tool and the projects created with it. The user is not always aware when new version is available, and they check once or twice a year if new version is available. Modernization of this tool is seen important, since the current version is 20 years old, and things have improved since then. According to the interviewee, transitioning to a web based version of the sizing tool will improve the safety of intellectual property. In addition to this, they think that data validation and version control could be improved.

The problems identified during the expert interview are presented in Table 6.

Table 6. Legacy application related problems identified from expert interviews specific to the case modernization project.

ID	Problem	Description
EXP-01	Information security	Intellectual property stored in legacy systems is seen as problem, since extraction of this data is possible and rather easy.
EXP-02	Software integrity	End users are able to modify the software systems, especially in the case of Excel-based tools.
EXP-03	Distribution of updates	End users have a lot of responsibility when it comes to keeping their version of system up to date in the case of offline local systems. Additionally, end users need to fetch the updates manually, and they need to be aware when new version is released.
EXP-04	Version control of the sizing projects	Version control for the sizing projects would be beneficial since multiple users will be making modification to the same sizing project.

4.3 Objectives of the solution

The objectives of the solutions are based on the exploration of previous work, organization interviews and expert interview. The main design objective of the artefact is to provide a process model, that first helps the organizations identify and evaluate the problems in their legacy system and secondly helps select the correct modernization strategy. All of the interviewed organizations are small teams, with 5-10 people, which indicates that the process needs to be light and easy to use. Resources was observed to be limiting factor when starting modernization projects, which is why it should be as easy as possible to begin the project. This way the organizations have lower threshold to start using the process framework and can collect valuable information about their system with limited amount of resources allocated to the modernizations. Organization 2 had struggled with funding for a decade, which would indicate that more effective argumentation for modernization is required. The result that the process model outputs should be absolute, providing repeatable and analytical results for the organizations. This way any human error can be minimized, and the results from different implementations of the process can be compared. Another way to reduce human error and resource intensity is to create as simple process as possible. Finally, the process should be documentable, so that future knowledge issues can be avoided. The sub-objectives set to the process model are presented in Table 7.

Table 7. Objectives for the design science research artefact

ID	Description
OBJ-01	**Evaluation methods:** Provide light and easy-to-use methods to evaluate modernization need, which can be used as part of argumentation.
OBJ-02	**Argumentation:** Provide methods to argument modernization needs. Preferably some that provide quantitative data.
OBJ-03	**Decision making:** The process needs to support modernization decision making process, so that organizations can make informed decision regarding their legacy systems and tools.
OBJ-04	**Ease of use, light process:** The use of the process needs to be easy, so that it does not increase the workload of the target organizations. The organizations that the process is targeted to generally have limited resources when it comes to tools and software, which means the process should be light for the organization to use.
OBJ-05	**Absolute result:** The result that the process model provides should not leave any room for interpretation, meaning the result should be absolute. This is required to make the process repeatable and quantifiable.
OBJ-06	**Simplicity**: The process should only contain the necessary things to reach the main design objective
OBJ-07	**Documentation:** Since of the main issues in the legacy systems are knowledge related, the process should be well documented

5 Modernization process framework

Development of the modernization process framework is described in this chapter. In the first sub-chapter the existing solutions related to the artefact are analyzed. The artefact is designed in the second sub-chapter.

5.1 Existing solutions

According to the design science process model, the researchers should have knowledge of current solutions to the problem and how well do these solutions work. To achieve this, process models and frameworks were searched from literature. The search process involved the usage of LUT university database Primo, as well as the Elsevier Scopus database. Keywords used during the search of research papers were “software modernization”, “process”, “model” and “decision making”.

A general process for legacy software modernization has been developed by the Object Management Group (OMG) called Architecture-driven modernization (ADM). ADM is a concept that tries to capture the current architecture of the legacy system and transform the architecture into the new system. The OMG taskforce define ADM as the understanding and evolving of existing software artefacts and systems. Organizations can restore the value and recover their investments in legacy applications by extracting the knowledge stored in them. (OMG, 2003).

The ADM concept can be implemented by carrying out re-engineering process using the model-driven architecture (MDA) standard, which is also developed by OMG. MDA defines the notations and structure of system models, which then comply with the standard making them “MDA models”. It was developed to emphasizes the use of models to specify and design software systems and to derive value through these models. MDA models include various levels of abstractions, ranging from models describing high level enterprise architecture to models depicting the technical implementation of systems. In some cases, the models themselves can become the executable part of the system. For example, database management system schemas are well known to be executable models in many cases. (OMG,

2014) The MDA standard defines three types of models which are computation independent models (CIM), platform independent models (PIM) and platform specific models (. Computation independent models are high abstraction level models, platform independent models serve as an intermediate level and platform specific models are only applicable to certain system platform. (Pérez-Castillo et al., 2012) The ADM process is like a horseshoe, as depicted in Figure 6.

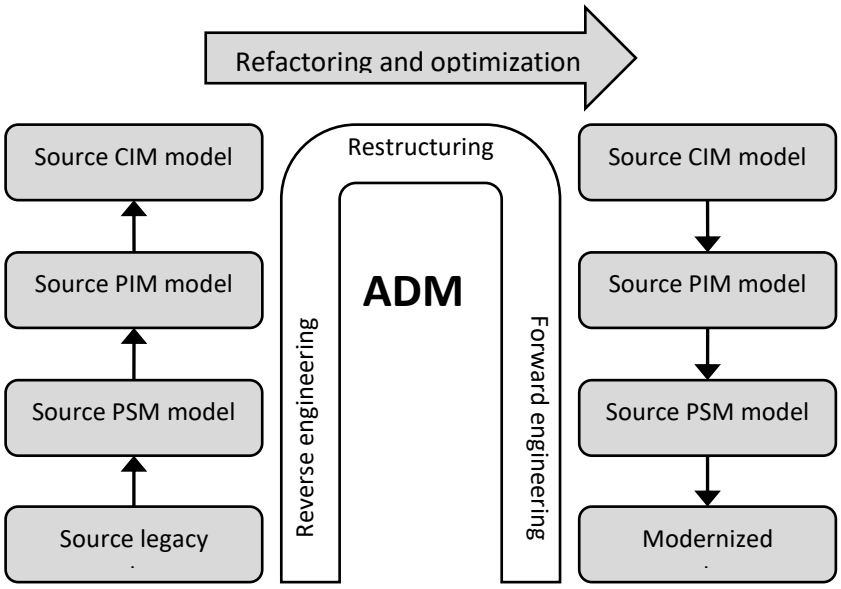


Figure 6 The ADM process model (OMG, 2003)

Pérez-Castillo et al. suggest a model-based approach for creating web services from old legacy databases. The model they propose would allow organizations to develop such web services, that the data stored in the legacy databases could be accessed. This process approach promotes the idea of service-oriented architecture, while based on the ADM. They implement the horseshoe process model from the MDA, meaning that the modelling of the system is started from low abstraction level. This model is transformed into high abstraction model and forward engineered to be better while still containing the business logic of the legacy system. Finally, the high-level model is translated to lower-level abstraction for implementation. The process they suggest is therefore a more specified implementation of the MDA in the context of legacy databases. (Pérez-Castillo et al., 2012) The process suggested by Pérez-Castillo et al. is presented in Figure 7.

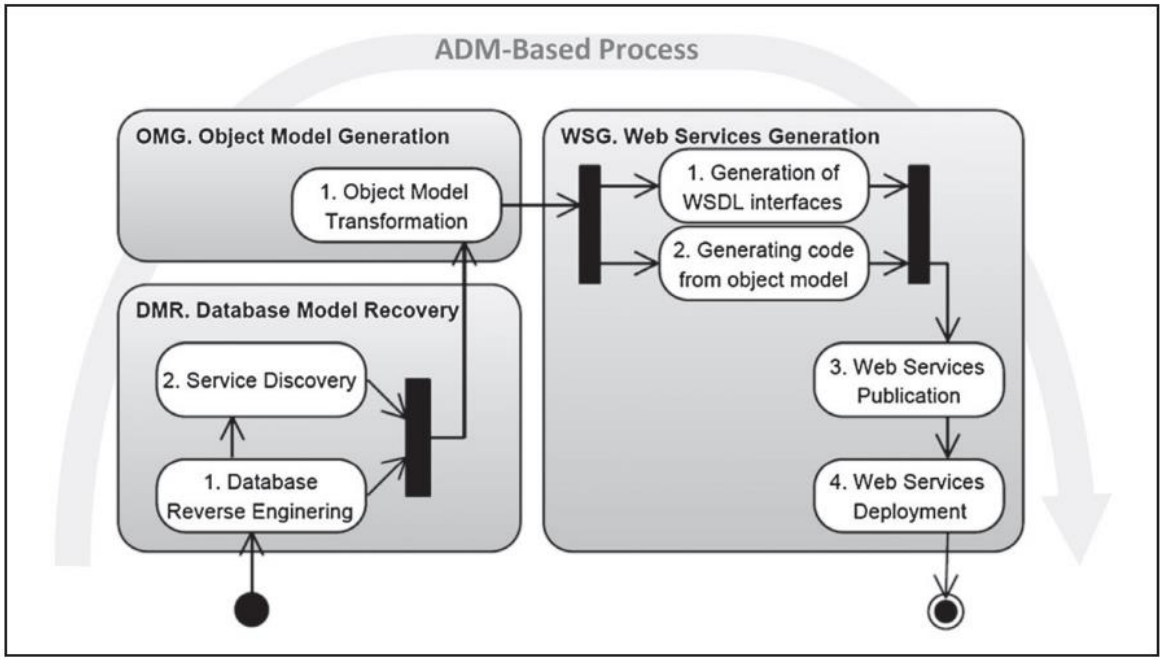


Figure 7 The PRESICO process framework (Pérez-Castillo et al., 2012).

Various process models have been suggested that help developers divide legacy monoliths into microservices. Microservices are small autonomous modules, that work in union to produce whole business processes. These modules are interconnected and communicate by utilizing internet protocols. Microservice architecture has become trending architectural style for dissecting monolith legacy applications into smaller independent cloud services. Microservice architecture is used to create highly available and scalable systems. Well defined boundaries in the designing phase are important when modernizing legacy monolith applications. Boundary definition is also identified as the most difficult part of migration to microservice architecture. It is considered difficult, since there is large number of possible service combinations that would work and that there are multiple criteria that can be used as the optimization metric. (Assunção et al., 2022)

Microservices are a trending approach to system re-engineering. The microservice architecture has benefits of being highly maintainable and scalable. Since maintainability is often important for enterprise applications, microservices is seen as viable option in the future. Knoche & Hasselbring present a process model to transform legacy application into microservices gradually. The suggested process consists of five steps, which are presented in Figure 8. In the paper they use the process to modernize a large legacy application, which

is serving customer data to multiple client applications. They managed to identify 29 service interfaces, that were later translated into independent microservices. The team faced some issues with the microservice implementation of the project. Since microservices do not guarantee transactions in the database layer, there has been some organizational barriers, and after four years the transfer to microservices has not been completed fully. (2018)

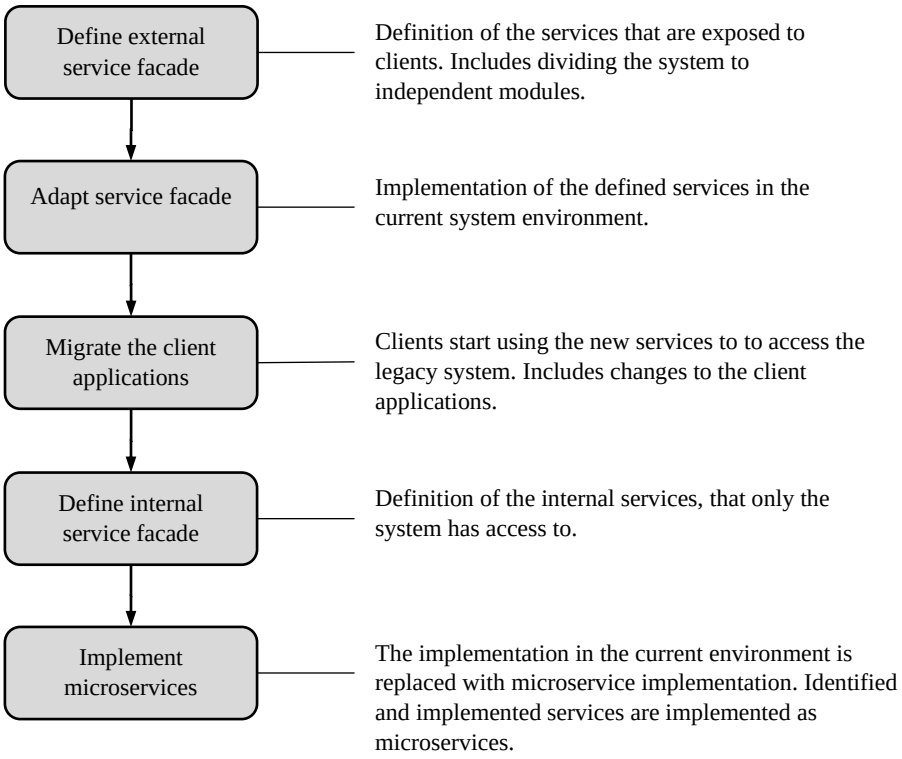


Figure 8 Process of modernizing legacy systems following microservice architecture. (Knoche & Hasselbring, 2018)

Modernization evaluation method called MODEST can be utilized to evaluate the need for modernization, which can help organizations in decision making. MODEST was also shortly described in Table 2. Tilus describes MODEST as light, easy to use and effective method, that can help organizations to identify the legacy systems that need to be analyzed further. In other words, MODEST does not analyze what should be done to the system, but if modernization should be considered and the need for it analyzed further. The process model is based on identifying individual sources of modernization motivation and evaluating the weight of them. The evaluation part of the process consists of three steps, while there are two additional steps for calibration process. In the first step, the system in question is divided

into sub-systems, in order to make it easier to identify the modernization motivation. MODEST includes 15 factors that are used to assess each sub systems. For each factor score is given ranging from 1 to 5, 1 meaning insignificant and 5 meaning extreme motivation for modernization. After each factor has been assessed, the score is generated which can be used to decide if modernization is needed. (Tilus, 2006)

Azani suggests two decision models to support modernization decision making: Analytic Hierarchy Process (AHP) and Goal Programming (GP). Both are methods that help organizations with multiple legacy tools or systems to make informed modernization decisions and allocate their resources effectively. AHP is a process that can be utilized to analyze and resolve complex decisions. The problem in the case of modernization is which modernization solution to select. To rank the solutions, there needs to be a set of qualities or features, that are seen as important qualities. The qualities need to be prioritized, by comparing each quality, thus assigning a weight for each of them. After the options and qualities are defined, the organizations need to compare each modernization solution to one another. This way the organization will create an ordered list of modernization solutions, each with some weight value which describes how well it suits the needs of the organization, compared to the other solutions. The result is not absolute, but relative to the selected qualities and solutions. (Azani, 2009)

5.2 Design & development

The first version of the process framework is presented in Figure 9 at a high level. The three main steps of the process are formal evaluation of the modernization needs using MODEST, gathering of arguments for modernization and AHP decision making process to select a suitable evolution strategy. MODEST was selected since it is light and effective way to evaluate the need for modernization (Tilus, 2006). MODEST allows organization to minimize the use of resources that they need to use for evaluation of modernization. AHP on the other hand was selected, since the process is very analytical, repeatable, and well documented (Azani, 2009). AHP minimizes the use of intuition in modernization decision making. Each of these steps will be described in detail in the sub-chapter below, as well as the demonstration of each step by applying it to the case modernization project.

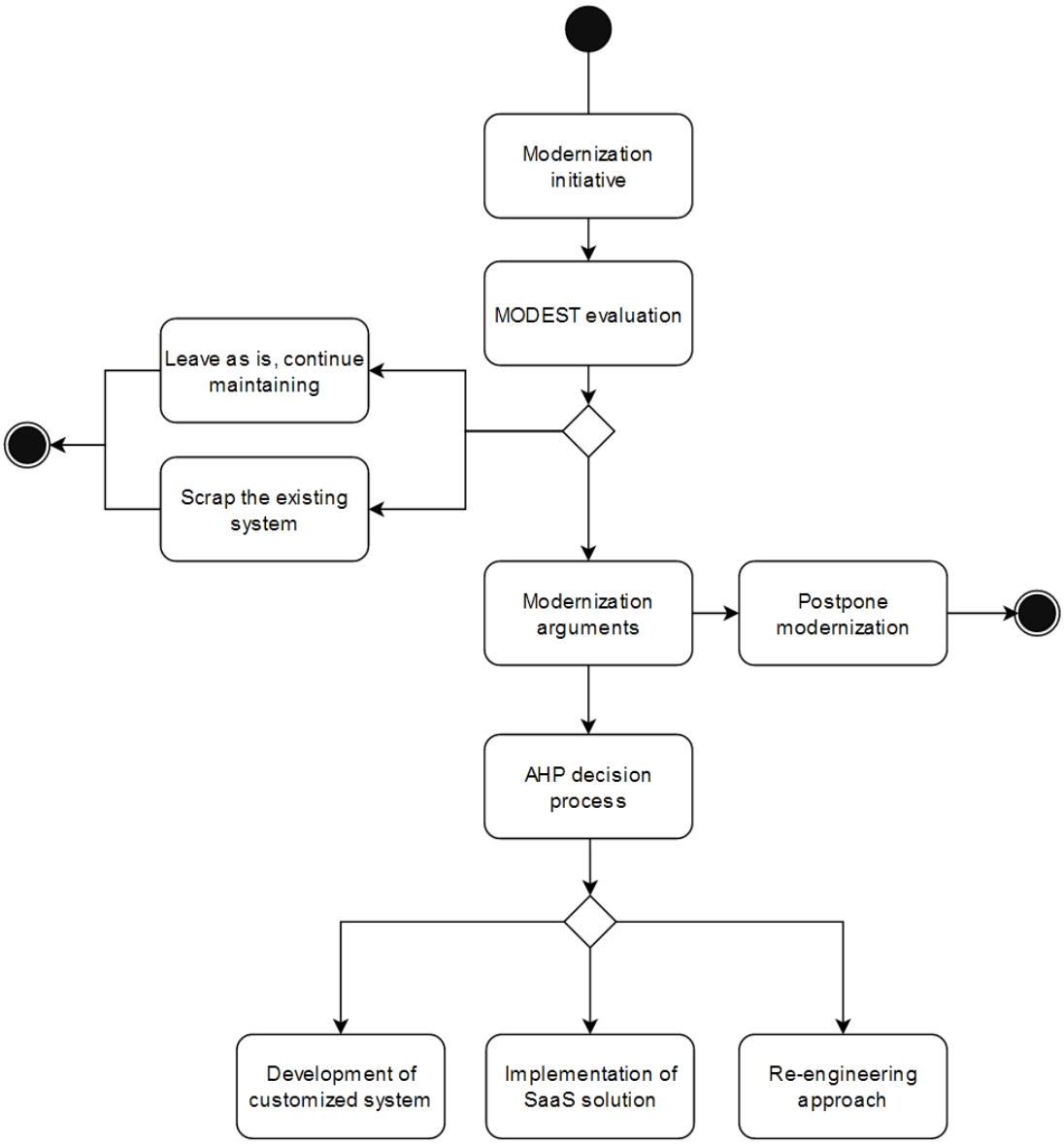


Figure 9 High level process modernization process framework

5.2.1 Modernization initiative

Every modernization project starts with an initiative. It is not necessarily part of the modernization process, but it is essential for the modernization of system to be considered and studied further. There are various sources for modernization initiatives proposed in literature, such as software consultants or vendors that have developed the systems. In the case of engineering organization that is developing and maintaining their own software, the initiative generally needs to come from inside the organization. Initiative can come from

managers, developing engineers or the users of the system. There is also the possibility for external consultant to be hired by the organization, who is responsible for analysing the business processes and the systems used to complete them. The consultant can then inform the organization about the possible modernization initiatives. The initiative can be motivated by various reasons. When it comes to legacy systems, technologies becoming obsolete is common event that triggers modernization initiative. Organization may choose to integrate their business processes, which will require the modernization of legacy systems. Changes in business processes might also require changes to the legacy system, which can further lead to the need for system modernization.

The initiative for the case modernization project came from multiple people within the organization, who raised their concerns related to the state of the current software. Similar issues that were identified during the interviews had been raised, including the challenges of software version control, intellectual property security, risks related to external contractors and technology. The biggest issue was identified to be knowledge related. The internal developer of the current system retired over a year ago, and with them large quantity of system knowledge was lost. Although relevant issues were already identified, the decision to modernize is to be formally evaluated by utilizing the MODEST process.

5.2.2 MODEST evaluation

The MODEST method for analysing modernization motivation was shortly introduced in chapter 5.1. This method was selected as part of the modernization process since it requires very minimal resources, while being very effective method for analysing the need for modernization. The MODEST process is used as the initial filter for modernization projects in his process framework. By utilizing the MODEST process, organizations can quickly identify the tools and systems that they should prioritize using their limited resources. During this chapter, basic instructions and working principles are described. For more in-depth guide for MODEST, Tilus has created an extensive guide on how to conduct the analysis (2006).

MODEST evaluation can be conducted with the internal resources available in the organization, assuming that there is a person available with extensive knowledge about the system in question. Estimated work effort for MODEST evaluation is one workday.

MODEST evaluation has 5 steps, which are division, scoring, calculation, and retrospective analysis of parameters (Tilus, 2006). The last two steps are relevant when the organization plans to evaluate the needs for modernization more than once for the same tool. Otherwise, the retrospective steps can be ignored, which might be the case for most engineering tools. The general process of MODEST evaluation is presented in Figure 10.

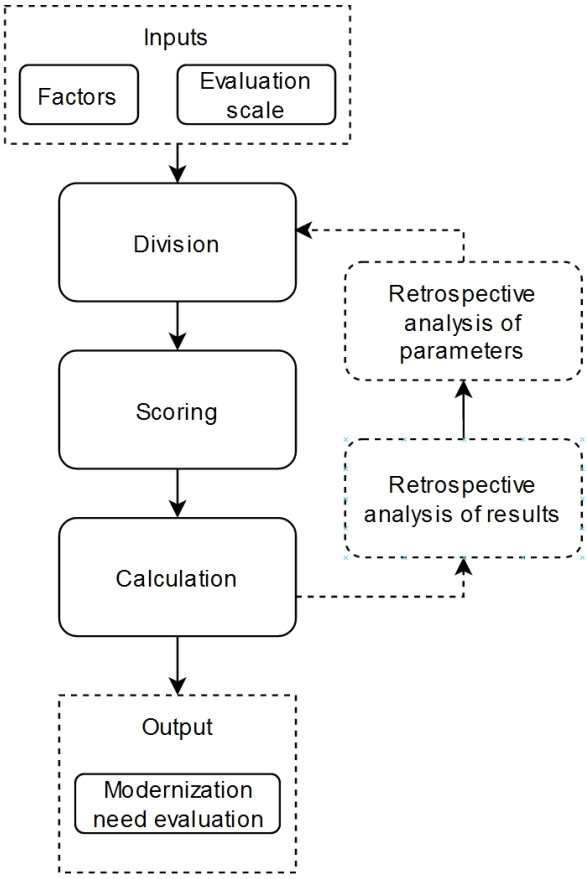


Figure 10 MODEST process model (Tilus, 2006)

The evaluation process can be conducted in one session. The session requires one person who has expert-level knowledge about the system in question, as well as a person who is responsible for noting the results of the evaluation (facilitator). The evaluation can be conducted by the expert alone, but the benefits of facilitated documented conversation is lost this way. The facilitator is responsible for guiding the process and conversation, while the expert needs to have up-to-date knowledge about the system. The facilitator does not need to be familiar with the system. The process requires factors and evaluation scale as input parameters and the end result is evaluation of modernization need for the system in question. The factors are a pre-determined set of known themes or subjects that are known to affect

the modernization need of a system. During the evaluation process, modernization needs are identified and added under corresponding factors. The list of factors suggested to be used in this process is extended version of the original set presented by Tilus, since it was discovered that information security was identified as an issue in all legacy systems. This factor was identified from the organization and expert interviews. Full list of factors is presented in Table 8. Factors suggested by Tilus are identified with a superscript. The factors are not meant as the final set that should be followed. There might be factors that arise, when conducting the evaluation with the system expert. These factors should be also considered and added to the current evaluation, as well as appended to the list of factors for that system to be used in the possible next evaluation.

Table 8 MODEST modernization evaluation factors ([1] Tilus, 2006)

Factor	Description
Usability [1]	Modernization motivation related to usability of the software or system are listed under this factor. This factor includes issues related to user experience.
Support for used technologies [1]	Motivations such as discontinuation of certain programming languages or frameworks can be included in this factor.
Effectiveness [1]	Any motivations that are related to how well the current system is providing the desired result are included in this factor.
Complexity [1]	System complexity contains motivations such as the structure of current system being overly complicate, or that the current process is more complex than it needs to be.
System knowledge [1]	This factor can include motivations such as system experts leaving the organization and issues related to documentation.
Scope of modifications required [1]	If changes that would be required to the current software are extensive, motivation for modernization could be high in this factor.
Cost of modifications required [1]	This factor can be strongly related to the scope of modifications. Cost can be motivation for modernization, depending on how extensive modifications are required.
Changes in the operating environment [1]	Motivation for modernization can arise from changes in the environment where the system or software is deployed
Expected remaining system lifetime [1]	System lifetime can be strong indicator for modernization but can be hard to estimate.
Changes in the business process [1]	Business processes are constantly evolving, which can cause systems to become outdated, which can motivate for modernization.
New technical possibilities [1]	Technical possibilities that new technologies bring can be motivation to modernize. This can be the case when the current technology is limiting the possible functionalities.
Information security	This factor is used to evaluate the criticality of the information security risks related to the legacy tool

The evaluation scale for the MODEST analysis is used to measure the weight of each modernization motivation identified during the evaluation. The scale used is a generic six step scale, that has values from 0 to 5. The definitions of the classes are the following.

0: None, no reason for modernization.

1: Noticeable, slight but noticeable motivation to modernize.

2: Apparent, there are regular issues related to the factor which have been extensively noted.

3: Troubling, action needs to be taken in day to day basis to deal with the problems regarding the factor.

4: Defeating, resources are regularly lost to problems cause by the factor. Change would bring quantifiable benefits but not necessarily more than drawbacks.

5: Intolerable, there are and will be serious issues with the problems related to the factor. The modernizing is definitely more effective than dealing with the problem (Tilus, 2006)

In order to be able to make decision using the scale, limits for the scale needs to be set. As a starting limit values, value for the upper limit is set to 5, while the lower limit is set to 3. If the calculated modernization indicator is smaller than the lower limit, modernization does not need to be considered further. If the indicator is 3 or higher, modernization needs to be considered, while modernization cannot be delayed if the indicator reaches the upper limit of 5.

First step of the evaluation process itself is the division of the target legacy system into smaller sub-systems. The purpose of division is to make the motivations for modernization easier to recognise. (Tilus, 2006) In the case of smaller engineering tools, like simple macro-enabled workbooks, this step might be irrelevant. As a general rule, if the system cannot be divided into at least three separate entities, the division of the system is not necessary. The resulting components should cover the legacy system in question totally, without exceeding the limits of the system. It is therefore essential to define the legacy system analyzed as well as the business process associated with it before division. The division of the system should be done in a natural way, but other than that there is no right or wrong way to do it. System can be for example divided based on functionalities in the system, based on different user groups and their needs, or by separating interface, logic, and data layers (Tilus 2006). The division can be reconsidered during the evaluation if factor cannot be scored by giving one value. This means that the system could be divided into more parts so that the modernization motivation score can be given naturally.

The case system was divided into three sub-systems: interface layer, macro layer, and data layer. This way of division was the most natural way of analysing the system, since these three parts can be clearly identified from each other by used technology. The division of the case legacy tool is presented in Table 9.

Table 9 Division of the case legacy tool for MODEST analysis

Sub-system	Scope
Interface layer	Three Excel workbooks used for inputting data, making selections. Some simple calculations are also done in the interface layer.
Macro layer	Contains functionality that is used for calculations related to the business process, as well as various supporting functionality. Handles data transfer between the three workbooks, fetches data from the data layer and saves the work done to a predefined data format.
Data layer	Contains product data used for calculations. Data layer consists of one Microsoft Access database, that is locally stored on each user's machine.

The second step in the process is scoring. During the scoring phase, sources of modernization motivation are identified in facilitated setting, and the strength of them are evaluated using the scale defined above. When a motivation for modernization is recognized, it is added under the corresponding factor. If no suitable factor is present, factor can be created and added to the evaluation. For each of the modernization motivations recognized, score is given. If the system was divided during the first step, the score is given for each part individually. The scoring for the case legacy tool was conducted by one person. The person evaluating the system used an Excel workbook as a tool, which had all the factors listed with explanations, as well as columns for each sub-system. The factors were used as aid when recognizing the sources of modernization motivation. In addition, the scale used for giving each motivation sources score was presented on the Excel sheet. The assessor went through each of the factors and analyzed the different sub-systems from that point of view. When a motivation for modernization was identified, the assessor added it under the corresponding factor. The score of the factor was assessed using the provided scale for each sub-system individually. The identified motivations for modernizations are presented in Table 10.

.

Table 10 Modernization motivations identified for the sizing tool project and their respective score for each sub-system

Identified motivation	Factor	Score		
		Interface	Macro	Data
User is responsible for using the latest version of the software available	Usability	3	3	3
The user interface is crude, and not user friendly	Usability	2	-	-
Support for VBA is not guaranteed, since Microsoft is pushing the use of TypeScript	Support for used technologies	-	3	-
Lot of the sizing work is manual interpreting of pump diagrams, which could be automated	Effectiveness	2	2	-
The sizing process itself contains too much chance for human error	Complexity	3	-	-
Main developer and maintainer of the system has departed the company	System knowledge	3	3	3
Only one person (external contractor) is familiar with the source code of the VBA macros	System knowledge	1	5	1
There is no integration with other systems/software used in the business process	Changes in the operating environment	1	3	3
Data collected from other business processes is not utilized to the fullest	Changes in the business process	-	2	2
Modernizing could allow for use of data analysis when sizing data is collected in systematic way	New technical possibilities	-	-	2
Utilizing external services is not possible without compromising IP	New technical possibilities	-	3	3
Current technology does not support the use of AI models for data analysis	New technical possibilities	-	2	2
Product data can be distributed uncontrollably, when it is stored locally on users' machine, with weak password protection	Information security	-	-	2
Sizing program can be distributed to any actor, and the application is only protected by weak password	Information security	2	-	-

Final step of the evaluation process is calculation, which is the simplest and easiest part of the process. The result of this calculation is so called modernization indicator is defined simply based on the largest value observed during the evaluation. (Tilus, 2006) The calculated indicator is then compared to the limits set at the start of the process and decision to evaluate the modernization need further. The score for the case MODEST analysis is 5, since that is the largest score observed during the analysis, originating from the issue that only one external contractor is familiar with the source code of the VBA macros. Utilizing the lower and upper limits set earlier, it can be stated that based on this evaluation, modernization of the system should be analyzed further and started without unnecessary delays.

5.2.3 Modernization argumentation

The modernization argumentation step is a supporting step, that is meant to help organizations gather solid reasoning for why the modernization of the system in question is necessary. The arguments gathered can then be used to get the modernization project approved. This step of the process is structured or defined in detail, since the need for arguments or quantified data can be very different from case to case. Instead, some common methods are described, as well as one example from engineering organization.

Workshops are identified as effective ways to gather modernization arguments, as demonstrated by organization 2. Organization 2 invited stakeholders to a two day workshop that was facilitated by a professional external contractor. During these two days, information about the experiences, needs and opinions from the users and other stakeholders were gathered. The developing organization was able to use this information to form a complete picture of the current state of the system and what the users and other stakeholders of the system wanted it to be like. It is suggested that workshops are organized if there are resources to do so.

Some general methods of modernization reasoning include cost estimation methods such as return on investment (ROI) calculations and the COCOMO method. These methods of evaluating the costs can be very resource intensive and time consuming to conduct. Sometimes it might be reasonable to use less formal methods of estimating the costs, especially when the system in question is not complex or large. Such methods can include

expert evaluations and simple comparison between different modernization option costs. The cost comparison can also be included in the AHP process, where different options are ranked based on various criteria.

For the case project knowledge issues, possible end of support and improvement of intellectual property security were selected as the main arguments for modernization. The case project did not require any formal quantifiable results to begin, since the need for modernization had already been identified by management, and the project had been approved.

The knowledge issues are a twofold issue with the current system. First of all, the available documentation for the software was either never produced or it has gone missing during the 30 years the system has been in production. Even if the documentation from 30 years could be retrieved, it would be highly unlikely that it would contain the modification made to the system during those 30 years. The other aspect of the knowledge issue is the use of external contractor, that is responsible for the modifications to the macros programmed with VBA. There used to be an internal expert that was also familiar with the source code, but they departed the company few years earlier. The use of a singular external contractor is inherently risky since the contractor is a single point of failure, meaning maintenance operations stop when this person is not available. The effect of this can also be seen in the long maintenance cycle times described in the problem definition chapter.

The possible end of support is related to the programming language and technologies used both in the macro layer as well as in the data layer. Visual Basic for Applications or VBA is the macro language that is used in the sizing program. Microsoft has not announced that they would stop supporting VBA, but they released the Office Scripts in 2020 for both the desktop version and the online version of the Office suite. The benefit of Office scripts to VBA is that they can be run on both the online environment and desktop environments, while VBA macros are only available for desktop applications. In addition to this, Office Scripts are programmed in TypeScript. (Microsoft, 2023) The benefit of using TypeScript is that JavaScript is one of if not the most popular programming language, so it is not difficult to find people who can program these new Office Scripts. All of these things hint to the direction that the support for VBA will end at some point and Office Scripts being the replacements. The technology used in the data layer could also be facing end of support at some point, although now the support for Microsoft Access databases seems to continue.

Microsoft has removed the Access tool from their online tool suite in 2018, they still continue to develop and support the desktop version of the software. Support for integration of Microsoft Power Applications has also been released, which can help organization to shift to the cloud-based Power suite.

Third strong argument for modernization is the security of intellectual property. The data layer of the sizing tool contains product data used for process calculations, which are considered Metso's intellectual property. Additionally, the equations that are used to calculate various process variables used to select and size the equipment. The current state of the software could allow for third party actors to access the data. If for example the sizing tool Excel workbook gets shared to third party, the equations can be accessed after figuring out a weak password. The same situation can occur with the Access database, since it is as easy to share as the Excel workbook, while being protected with as weak password as the workbook.

Based on the arguments that can be generated, the manager or the product owner of the system should be able to make the decision to continue the process or not. This is the case only when score from the MODEST analysis did not reach the upper limit set but was between the lower and upper limit. In the case modernization project, the arguments support the result from the MODEST analysis, and modernization process will be continued.

5.2.4 AHP for modernization approach

AHP or analytical hierarchical process is the second sub-process of the process framework. In general, the AHP is a decision-making process can be used to select the best option for a certain goal by comparing the available options. The different options are compared by utilizing attributes that the evaluator sees as important qualities for the goal. In order to order the possible options from best to worse, the evaluator needs to compare the qualities to each other, and this way create a prioritized list of the qualities. After this, the each of the options can be compared to each other for each of the qualities, thus creating a prioritized list of the possible approaches. In this process, the goal of the AHP is to decide on the best modernization approach. The options for the goal are developing new system, implementing SaaS solution and re-engineering the system. For each of these approaches, multiple

different options are available, which is why the exact options should be defined case-by-case. The general process of AHP is presented in Figure 11.

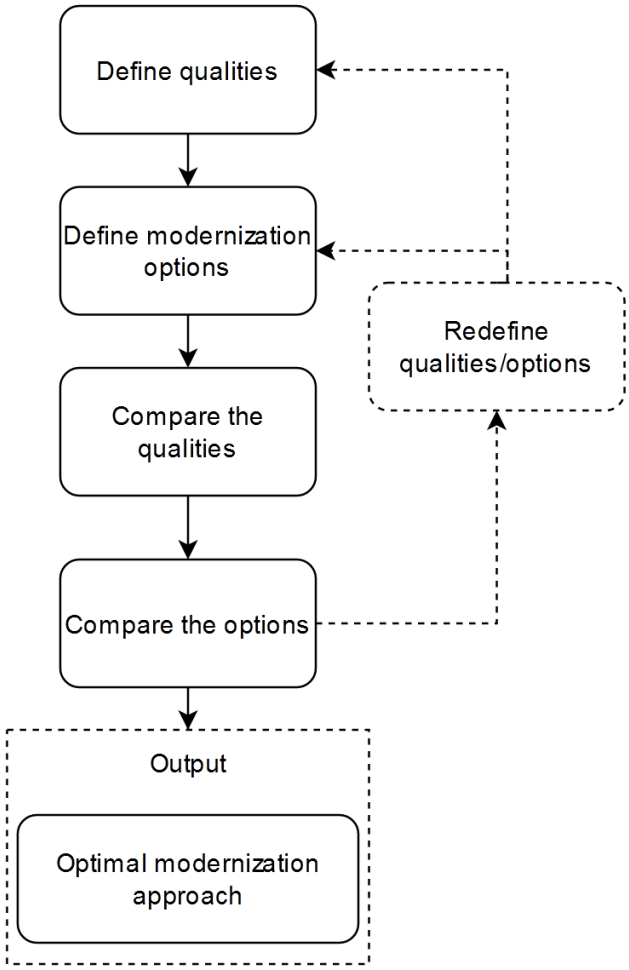


Figure 11 AHP for selecting optimal modernization approach.

AHP includes activities that require knowledge about current technologies used to build software, technical understanding about differences between these technologies and possible off the shelve solutions that would suit the application. During this phase people with modernization experience and software engineering should be involved in the process. Otherwise discovering and defining the possible approaches can be hard to do. The comparison of the approaches also requires technical knowledge, but defining and comparing the qualities can be done with basic knowledge about the system that is being modernized, as well as understanding of the requirements that the new system has.

AHP starts with discovering and defining the qualities that are seen important for the system in question. The qualities or objectives that are selected for the evaluation should address the issues with the current system, as well as other factors that are seen as important. MODEST

evaluation results should be revisited at this point of the process, as it will underline the issues that the modernized system needs to address. There are of course other qualities that can be seen important, such as total cost of ownership, future maintenance efforts and costs, integration with other systems and development time. There is no set limit for how many qualities can be included in the analysis. General rule of thumb would be to have twice as many qualities as options, in order to guarantee clear separation of the different options. It should be kept in mind that the number of qualities affect the workload, since one comparison matrix is created for each of the qualities.

The qualities and features selected for the AHP of the case project are presented in Table 11. The qualities were selected based on the problems identified in the existing system, as well general requirements that have been set for the modernized system. Some of the qualities would require large amount of designing and planning to analyze in detail, such as cost of maintenance. As the purpose of this framework is to be as light as possible, the qualities are estimated rather than quantifiably analysed.

Table 11 Qualities selected for the AHP of the sizing tool project

Quality or feature	Description	Abbreviation
Cost of maintenance	Maintenance cost is estimated for the different solutions, considering both corrective and evolutionary maintenance.	COST
Enterprise integration	Enterprise integration describes how easy or difficult it is to integrate other enterprise systems to the modernized tool.	ENTER
Customizability	Customizability describes how much room there is for customization in the proposed solution.	CUSTOM
Evolvability	Evolvability describes the ability to add new features, and how easy or difficult this is.	EVOLVE
Autonomy of administration	Administration automation describes how independent the engineering organization can be using the modernized system. For example, administrative tasks include modifying the product data, updating the calculation formulas, managing user access.	AUTO
IP security	Measures the security of intellectual property in this proposed solution. Intellectual property includes the product data as well as the methods of calculation	IP
User collaboration	User collaboration describes a feature of the proposed system. Collaboration describes how well users can work on the same projects in the system, how easy it is to share work with others and how smooth and easy the workflow could be.	COLLAB

Second step of AHP is discovering and defining the modernization approaches that are seen suitable for the system in question. There are mainly 3 different types of approaches that can be considered: developing a new system, re-engineering the current system and implementing SaaS solution. There are practically limitless solutions for each of these types, so it is necessary to select specific solutions in order to analyze the approaches effectively. As with the qualities, there is no limit for how many options can be evaluated at once, but it is suggested that at least three options are considered. The workload of the process does not increase significantly when more options are included, which is why it is suggested that all feasible options are evaluated. For the case project, three approaches were selected for further analysis using AHP. The modernization approaches are presented in Table 12. The first approach is implementation of the sizing tool in a low-code environment. The low-code environment selected for the evaluation was the Microsoft Power Apps platform. Power apps is a low-code platform, which helps organizations build apps for various business processes without extensive software development knowledge (Microsoft, 2023). Low-code platforms in general allow for so called “citizen developing”, meaning that people with no programming knowledge or training are able to create full applications. The Microsoft platform was selected since the organization the new system is developed for is using the Microsoft suite very extensively at the moment. In addition to this, the current application is implemented using the Microsoft suite, which can be helpful when implementing the new version of the application. Additionally, the data layer could be easily migrated over since it is stored in Microsoft Access. Microsoft has built tools to assist in data migration from old Office suite to the cloud.

Table 12 Modernization solutions selected for AHP for case project.

Modernization solution	Description
Microsoft Power Apps	Current business logic embedded in the sizing tool would be implemented in Microsoft Power Apps. Functionality from interface layer and macro layer would be combined as Power Apps implementation, while the data layer would be migrated to Microsoft's Dataverse
Modernization of the data layer	Data layer of the current sizing program would be migrated. SQL database would be storing the data, and an API would be created for the VBA macros to fetch the data from the database. This requires some modifications to the macro layer, creating an API and the migration of the database. The functionality in the macro and interface layers could be implemented incrementally as part of the API, while the old familiar Excel sheets serve as the user interface.
Web application	Development of a full stack web application to replace the current sizing tool. Data layer of the system would be migrated to SQL database hosted on the cloud. The functionality from the macro layer and calculations from interface layer would be implemented to a backend service, that would be exposed via an API. The backend service of the system would be implemented using Python's Django framework. Interface for the sizing tool would be created using some JavaScript web framework.

Second modernization option is to migrate the current data layer into a SQL database hosted on the cloud. API would be created, allowing the Excel workbook to fetch the data stored in the database. The backend service could be built for example using Django. This modernization approach would also allow for incremental implementation of the functionality and features in the backend. During the implementation the interface would remain familiar for the users. The migration of the data layer would immediately increase the security of the organizations IP, as well as improve the data integrity of the system.

Third option considered is development of a totally new system. The modernized sizing tool would be a full stack web application, consisting of SQL database, Django backend service and user interface implemented in some JavaScript front-end framework, such as React. This approach would allow for the most amount of flexibility, as well as reconstruction of the embedded business processes. The old tool would be used along with the new tool, until the feature set of the new application can match the old tool. The new development would also have broader user base since it is planned as a simple tool for people in sales and other organizations to use as preliminary equipment sizing.

The qualities are compared and prioritized in the third step. Each quality is compared to one another, and preferred quality out of these two is selected by giving a score. The scale for the scoring is from 1-9, 1 meaning the qualities are equally preferred and 9 meaning the first quality is extremely preferred. This way a matrix is generated, which can be used to calculate the weights of the qualities included in the analysis. The calculation of weights is done using the Approximate eigen vector method, which is described as the AHP for the case project is presented. There are tools available for AHP, which eliminates the tedious task of calculating the weights for the qualities manually, although in this case the calculations were done in Excel. The comparison of the qualities selected for the case project is presented in Table 13.

Table 13 Comparison of the qualities selected for case project AHP and the sum of each column for further steps.

	COST	**ENTER**	**CUSTOM**	**EVOLVE**	**AUTO**	**IP**	**COLLAB**
COST	1.00	0.20	4.00	0.13	5.00	0.13	0.25
ENTER	5.00	1.00	5.00	1.00	6.00	0.25	3.00
CUSTOM	0.25	0.20	1.00	0.20	3.00	0.20	2.00
EVOLVE	8.00	1.00	5.00	1.00	5.00	0.50	4.00
AUTO	0.20	0.17	0.33	0.20	1.00	0.11	0.33
IP	8.00	4.00	5.00	2.00	9.00	1.00	6.00
COLLAB	4.00	0.33	0.50	0.25	3.00	0.17	1.00
SUM	26.45	6.90	20.83	4.78	32.00	2.35	16.58

After each quality has been scored against each other, the results are normalized. The normalization is done by taking a sum of each column and dividing each cell in that column by the sum. This way the sum of each column will be equal to 1. The normalized scores for the case project are presented in Table 14.

Table 14 Normalization of the quality comparison values for the case project.

	COST	ENTER	CUSTOM	EVOLVE	AUTO	IP	COLLAB
COST	0.038	0.029	0.192	0.026	0.156	0.053	0.015
ENTER	0.189	0.145	0.240	0.209	0.188	0.106	0.181
CUSTOM	0.009	0.029	0.048	0.042	0.094	0.085	0.121
EVOLVE	0.302	0.145	0.240	0.209	0.156	0.213	0.241
AUTO	0.008	0.024	0.016	0.042	0.031	0.047	0.020
IP	0.302	0.580	0.240	0.419	0.281	0.425	0.362
COLLAB	0.151	0.048	0.024	0.052	0.094	0.071	0.060

After normalizing the columns, weight of each quality can be calculated by taking the arithmetic mean of each row. The weights of the qualities in case project are presented in Table 15.

Table 15 Weights calculated for the case project qualities used for AHP.

Quality	Weight based on arithmetic mean
COST	0.073
ENTER	0.180
CUSTOM	0.061
EVOLVE	0.215
AUTO	0.027
IP	0.373
COLLAB	0.072

In the fourth step similar comparison is done where modernization solutions are compared to each other. One comparison matrix is created for each quality, in this case seven in total. Example of this matrix for the quality “Cost of maintenance” is presented in Table 16.

Table 16 AHP modernization solution comparison for the quality “Cost of maintenance”.

Cost of maintenance	**Power Apps**	**Data layer modernization**	**Web application**
Power Apps	1	5	8
Data layer modernization	0.2	1	3
Web application	0.125	0.333333333	1

Once all matrices have been created, the values are normalized, and the weights are calculated by taking the arithmetic mean of each row as with the quality matrix. The normalized values of the comparison and the calculated weight for the quality “Cost of maintenance” are presented in

Table 17. The final normalized weights are presented in Table 18.

Table 17 Normalized comparison values and the weights for quality "Cost of maintenance".

Cost of maintenance	**Power Apps**	**Data layer modernization**	**Web application**	**Weight**
Power Apps	0.755	0.789	0.667	0.737
Data layer modernization	0.151	0.158	0.250	0.186
Web application	0.094	0.053	0.083	0.077

Table 18 Final comparison values for each modernisation solution.

	COST	ENTER	CUSTOM	EVOLVE	AUTO	IP	COLLAB
Power Apps	0.74	0.39	0.06	0.16	0.67	0.38	0.31
Data layer modernization	0.19	0.12	0.26	0.11	0.18	0.15	0.32
Web application	0.08	0.49	0.68	0.73	0.15	0.47	0.37
Weight of the quality	0.073	0.180	0.061	0.215	0.027	0.373	0.072

Once normalization of the solution comparisons has been completed, the best modernization solution based on AHP can be calculated. The final results are calculated by multiplying each quality score with the weight assigned to it and taking a sum of whole row. For example, the final weight of the Power Apps in this case would be calculated as

$$Power\ Apps\ weight = 0.74 * 0.073 + 0.39 * 0.180 + \cdots 0.31 * 0.072 = 0.342. \quad (1)$$

The final weights for the modernization solutions are presented in Table 19. In this evaluation, Web application scored the highest, while Power apps and data layer modernization came second and third respectively. The weight of the web application is significantly higher than Power apps, which signals that based on these specific qualities, modernization options and evaluators the web applications is the most optimal solution for the case modernization project.

Table 19 Final weights for the modernization solutions based on AHP.

Modernization solution	Weight based on AHP
Power Apps	0.342
Data layer modernization	0.159
Web application	0.481

5.2.5 Modernization approaches

Once the AHP process is finished, the process of modernizing continues on one of the two paths presented in the high level diagram: development of a new system or re-engineering of the current system, depending on which solutions were selected for evaluation. For each of these approaches there are basically unlimited number of options when it comes to the processes, which is why it is difficult to create a general process model for the development or re-engineering. Although no general process model for the implementation phase is defined, suggestions for each approach is given in this chapter.

When organization decide to develop a new system, the options are basically limitless. The implementation can range from customized full stack applications to implementations of off-the-shelve solution. For engineering organizations, the prioritizing of different system qualities and functions is the key factor that decides which implementation to select. If the system requires a lot of integration or implementation of the business process requires a lot of customization to implement correctly, developing custom system should be considered. The organization should consult these types of projects from professional software system vendors, especially if integration with other enterprise systems is to be considered.

On the other end of the spectrum there are low/no-code platforms. The decision to select low/no-code platform as the solution can be caused by many reasons. Benefits of low/no-code include minimal development time and effort, maintenance of the system can possibly be done by the engineers, and implementations requiring very minimal external consulting. Some drawbacks of low/no-code solutions are limited flexibility and functionality, reduced integration possibilities to enterprise systems, and system modularity might be hard to achieve. Low/no-code suits well for organization, that have very limited resources at their disposal, and want to keep the initial investment costs to a minimum. This approach can also be considered the fastest way to implement modernized tool, which also makes it a great choice for organization that need immediately to modernize their tool for example due to changes in business process. The implementation of low/no-code solution is solely based on the vendor and technology selected. Organizations can also choose to implement their software on a low-code platform such as Mendix (2023) or Appian (2023). Low-code platforms allow for development of tools utilizing the building blocks provided by the

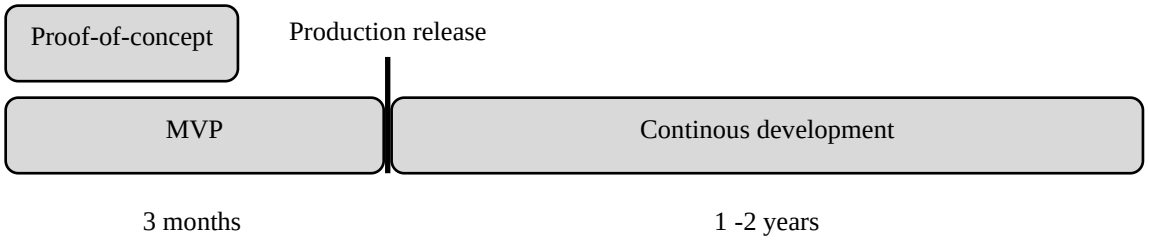
service. This way the engineers are able to implement the tools without hiring external contractors to conduct the programming work.

Off-the-shelve solutions can also be available depending on the business process. It was identified that one of the most common processes the legacy tool is used was product configuration, as in organizations 1 and 3. Companies such as Tacton (2023) provide customizable product configuration tools, that could be used to that specific business process. Off-the-shelve solutions can be considered to be most resource effective, but has limitations related to customizability, integration and evolution. The feature set of such system later in the life cycle could be very expensive and difficult to change.

Re-engineering approach is a suitable solution for systems that are complex and large in size. Re-engineering methods allow the modernization of the software to be done partly, which can reduce the total development efforts required to modernize the system. For example, in some engineering modernization of the data layer could solve most issues related to the security of intellectual property, as well as improve the data validity and consistency of the system. Re-engineering can also be required in those cases where the support of the current technology is ending. In these cases, re-engineering processes can be utilized to transform whole systems from one technology to another while utilizing automatic code generation with model based approaches. Modernization approaches were described at the start of chapter 5. Any of the suggested model can be used for re-engineering purposes, but as every case is unique, experts should be consulted on the best re-engineering model. In general, Architecture driven modernization is suggested for all of the modernization projects, where re-engineering is considered. This is why the ADM-based process suggested by Pérez-Castillo et al. is suggested as the process model for re-engineering (2012).

For the case project, development of a totally new system was selected as the modernization approach. Based on results from MODEST and AHP, best option with the knowledge available is to create a full stack web application to replace the current sizing tool. The process of modernization is rather simple. The development of the new system is separate process from the maintenance of the current one. The current tool will stay in use until the feature set on the new development matches the old tool. Roadmap for the modernization project is presented in Figure 12. During the first one to two months, the selected technology stack is validated by creating the initial “skeleton” of the system. This includes building all the core things, getting them running their intended production environment, and testing the

components as a system. The proof-of-concept contains end-to-end testing to validate the selected technologies serve the purposes well. If the technologies are seen as suitable, the development is continued, and the MVP of the modernized sizing tool is expected to be completed in 3 months. This MVP will contain the simple version of the sizing functionality, that will be available for use for every user. The basic feature set required at this point is described in Figure 12 Roadmap for the sizing tool modernization project. After the MVP is considered completed, the system evolution will continue with the addition of new features, such as integrating the pumps sizing system, adding the other auxiliary sizing functionality and integration with product database.



Proof-of-concept:

- Technology validation
- Skeleton of the system
- End-to-end testing

MVP developmentt:

- Feature set including:
  - Authentication
  - Basic filter sizing for all common technologies
  - Saving/sharing the sizing results
  - Outputting the sizing report
  - Outputting legacy file to be used in the old program
  - Technology selector based on customer data, mainly for test engineers and sales
- MVP will be considered completed when users are able to use the production version

Continous development

- Extending the feature set of the system with the ”advanced” features
  - Integrating the pump sizing using the service provided by Pumps organization
  - Implement other auxiliary equipment sizing features
  - OPEX calculations
  - Integration with testing database
  - Integration of product database

Figure 12 Roadmap for the sizing tool modernization project.

The technologies selected for the modernized tool along with high level architecture are presented in Figure 13. The core of the modernized system will be backend service built with the Django framework available for Python. This service will be available via REST API, which will allow for future integrations of other systems, as well as secured access to the product data and process calculations considered to be intellectual property of the organization. The backend service takes data that the user inputs, combines it with the product data found in the SQL database, and outputs results based on the sizing calculations.

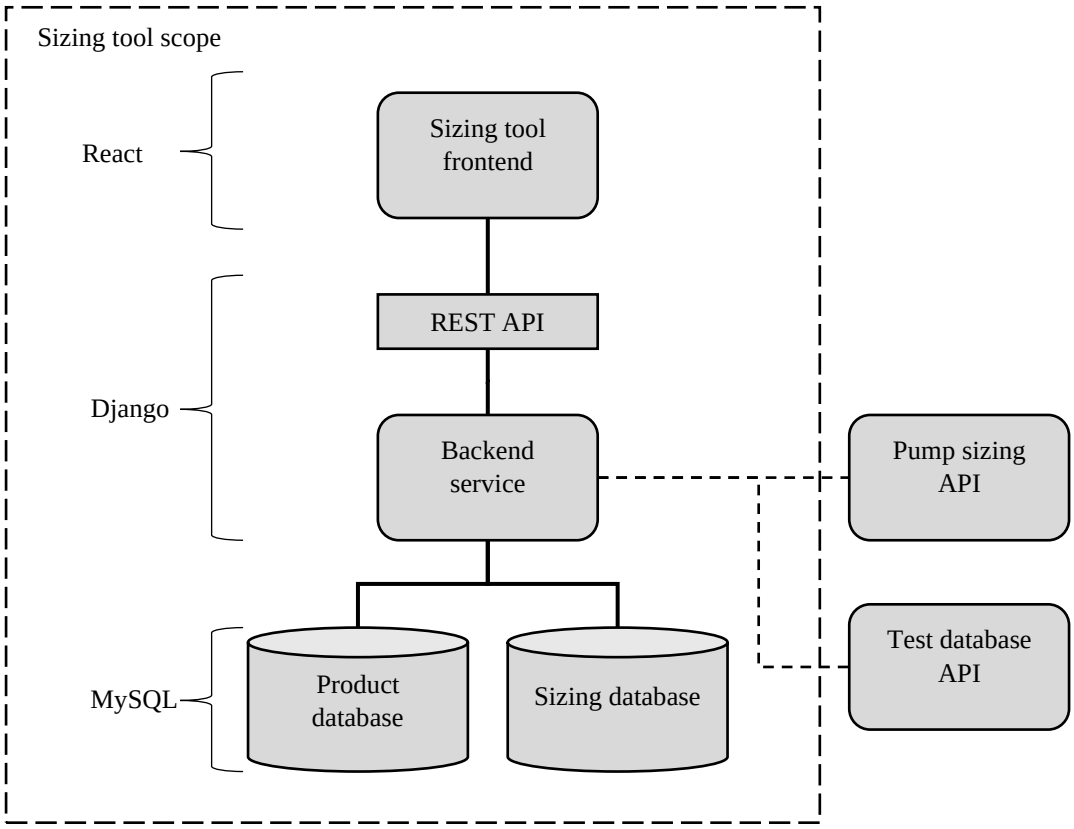


Figure 13 High-level architecture of the modernized sizing tool

6 Artefact evaluation

In this chapter the artefact developed and demonstrated is evaluated. The artefact is evaluated as a whole, after which the sub-processes are analysed further. According to Hevner, evaluation is crucial part of design science. Evaluation methods are used to assess the quality, effectiveness, and utility of the artefact. The evaluation step of design science provides the feedback required to iteratively develop the artefact further. (Hevner, 2004) The use of the artefact was demonstrated in detail in chapter 5, which is used to descriptively build an argument for the artefact's utility. In addition to this, the objectives set for the artefacts are revisited, and the artefact is evaluated based on how well it addresses the objectives set to it.

6.1 FEDS evaluation

Venable et al. suggest a four step method for design artefact evaluation in design science called FEDS. The framework provides support for selecting and executing the correct evaluation method based on the artefact being evaluated. The evaluation steps are defining goals of evaluation, choosing the evaluation strategy, determining the evaluated properties of the artefact, and designing of evaluation episodes. (Venable et al., 2016)

The evaluation starts with defining the goals for the evaluation. The goal of the modernization process framework evaluation is to evaluate if it meets the main design goal set to it, which was to help organizations to identify and evaluate the problems in their legacy systems and select the correct modernization strategy based on their needs. In addition to this, the usability and effectiveness of the artefact is evaluated.

Second step of the evaluation is the selection of evaluation strategy. Four different evaluation strategies are presented in the paper: quick and simple, human risk & effectiveness, technical risk & efficacy, and purely technical artefact. (Venable et al., 2016) The selection criteria for these strategies are presented in Table 20. Quick and Simple evaluation strategy was selected for the evaluation of the modernization process framework. The strategy was selected, since the process framework contains no technical risk, and the social risk that the

process framework has is very minimal. Further testing of the artefact in real context with real users is not feasible in the scope of this thesis work, which is why Human Risk & Effectiveness strategy was not feasible. Quick & Simple strategy does not contain much formal evaluation but utilizes more natural evaluation of the artefact. For this strategy it is common to have only few or possible just one evaluation episode during the research process. The strategy is desirable when low costs and faster project completion is prioritized. (Venable et al.,2016)

Table 20 Evaluation strategies for design science artefacts with the selection criteria (Venable et al., 2016)

DSR evaluation strategies	Circumstance selection criteria
Quick & Simple	If small and simple construction of design, with low social and technical risk and uncertainty
Human Risk & Effectiveness	If the major design risk is social or user oriented and/or If it is relatively cheap to evaluate with real users in their real context and/or If a critical goal of the evaluation is to rigorously establish that the utility/benefit will continue in real situations and over the long run
Technical Risk & Efficacy	If the major design risk is technically oriented and/or If it is prohibitively expensive to evaluate with real users and real systems in the real setting and/or If a critical goal of the evaluation is to rigorously establish that the utility/benefit is due to the artefact, not something else
Purely Technical Artefact	If artefact is purely technical (no social aspects) or artefact use will be well in future and not today

The evaluated properties of the artefact are defined in the third step. The properties selected for evaluation are functionality, evaluation methods, argumentation, decision making, absolute result, simplicity, documentation, usability, and effectiveness. The first seven evaluate how well the suggested artefact serves the purpose defined for it based on the objectives, while the last two evaluate organizational fit and performance.

Fourth step of the process is to design the evaluation episodes. For this project, one round of evaluation is conducted at the end of the first design iteration and demonstration. The design of the artefact will be retrospectively analyzed utilizing the general design goal, objectives set to it, observations made during demonstration of the artefact, as well as properties defined for the sake of evaluation.

6.2 Evaluation of the modernization process framework

The main design goal for the artefact was to provide a process model, that first helps the organizations identify and evaluate the problems in their legacy system and help select the correct modernization strategy. The process model was designed to fit organizations who maybe do not have extensive software engineering knowledge, are very limited in their resources allocated to software development activities and are utilizing various small to medium sized legacy systems. The process model addresses the main design goal well, since during the demonstration it was observed to function as required. User of the modernization framework was able to first discover drivers for modernization from the legacy system. After the need for modernization was identified, AHP was used successfully to select the optimal modernization approach based on the requirements and qualities that were seen important. Both MODEST and AHP have also been studied in previous work and proven to be effective methods in the context of legacy software modernization. Since the process framework context was small organizations with limited resources, the framework needed to remain light enough so that the use of it can be practical and justified. Completion of the MODEST analysis takes approximately one workday, which means that the need for modernization can be assessed very quickly and cost effectively. If the modernization process is continued further, the combined workload of the argumentation and AHP depends on various factors related to the organization, but the total effort can still be considered light. It can be stated that suitable process framework was designed to fit the purposes of small organizations.

Evaluation based on the properties defined is presented in Table 21. Since the evaluation strategy for the evaluation was selected to be Quick & Simple, no formal evaluation methods are used in the evaluation. The evaluation based on the properties is conducted by natural reasoning and some numerical data supporting those arguments. Weaknesses identified during the evaluation were found to be that the user of the process framework has no tools to define if the system should be scrapped totally.

Table 21 Evaluation of the modernization process framework.

Property	Description
Functionality	The artefact provides a complete solution to the main design goal, as well as the sub goals set to it in Table 7. In addition to this, the artefact sets some guidelines on the modernization approaches that the organization should consider.
Modernization evaluation methods	MODEST evaluation was found to be very effective method for evaluating the need for modernization. The use of MODEST can be slightly challenging at first, but with good system knowledge and some knowledge of modern software solutions the evaluation is simple.
Argumentation	Modernization argumentation was not defined strictly, since the need for arguments is very case sensitive. Some cost estimation methods were suggested in the framework, such as COCOMO and ROI calculations, but the use or effectiveness of these methods in the context of software modernization was not conducted.
Decision making	AHP was discovered to be very effective way of comparing and analyzing different modernization options. Although the process takes some time, especially when multiple qualities are taken into consideration, the effectiveness of the method makes it worth the resources. The process requires knowledge about possible modernization solutions, which is why some form of consultation is suggested, if the organization does not have their own software engineering knowledge. This is not seen as a disadvantage of the system, since some form of consultation will be required anyway during the modernization process. Decision for scrapping the system was not addressed, which can be seen as a weakness of the process framework.
Usability	The use of the process framework can be described to be easy, as the sub processes themselves do not require any specific technical knowledge, other than knowledge related to the modernized system. In addition to this, the workload caused by this process is small, estimated one workday for MODEST analysis, and possibly 2 – 4 workdays for the argumentation and the AHP analysis. This is all only estimation, but these results were observed during the demonstration project. The use of the process framework was also improved by providing tools for MODEST and AHP.
Effectiveness	Based on the case project, the modernization process framework can be described as effective. Modernization decision and approach was decided on using the process model, and modernized system will be developed based on results from the process.
Absolute result	The result that the process model provides leave no room for interpretation, meaning the result are absolute. To receive absolute results, some assumptions are required, for example during AHP evaluation. For a qualitative analysis the error is minimized. In order to receive absolute results, numerical data can be used in AHP, for example formal cost estimations for different modernization solutions.
Simplicity	The process contains only the necessary things to reach the main design objective. The process can be concluded after MODEST if no need for modernization is observed. The steps included in the process are only the necessary ones, while the argumentation step can be skipped for most of the cases.
Documentation	The process requires writing things down into the templates by nature, which promotes the documentation of the process.

7 Discussion

The main research objective of this thesis was to study development, maintenance and use of legacy systems and tools in small engineering organizations. The first research question of this thesis was what kind of challenges and issues these organization face when developing, maintaining, and using these legacy systems or tools. Main findings from the interviews and literature were that small to medium sized organizations commonly have limited resources to conduct modernization, no formal evaluation methods in use to evaluate the need for modernization and difficulties in argumentation, when resources would be needed for modernization. When it comes to issues during the maintenance and evolution of legacy tools, both previous work and the interviews indicate that knowledge management is the most common issue. Knowledge management is one of the key factors, when it comes to keeping a system maintainable. Issues discovered in the legacy systems themselves were also discovered during the interviews. Security issues and version control were found to be the most challenging in these particular systems. Previous work suggests that the ability to remain flexible or agile would be the most common reason to modernize a system, no such reason to modernize was identified during the interviews or evaluation. Smaller engineering organization might be flexible in nature, which would allow them to remain agile both in the ways that they operate and how the software is maintained. In addition to this, while the Excel workbooks have their disadvantages, the systems are surprisingly flexible even after 30 years.

To address the challenges discovered, the scope of the suggested process framework was selected to include the decision making process of the modernization project. The process was designed to be light, so organizations with limited resources would have lower threshold to implement it. Functionalities required from the process framework were way of evaluating the modernization need, argumentation methods to provide quantifiable data and method to select the modernization approach. The suggested process model was evaluated to meet the requirements set to it. MODEST process suggested by Tilus was found to be effective and light to use. One round of MODEST evaluation was estimated to require one day of work. By using MODEST, organizations can quickly select the systems and tools that they need to evaluate further. Resources can be efficiently used when the need for modernization can be

quickly recognized. Limitation of MODEST is the proprietary scale of the method. Different people would interpret the severity of modernization need differently. AHP was suggested as the method for selecting between different modernization approaches. AHP was successfully used to select between three modernization options. The limitation of AHP is that it requires some knowledge of current technologies and solutions in the problem domain. No obvious workaround for this issue was discovered. The engineering organizations need consultants at this point of the project. This might not be a bad thing, since it might be beneficial to include professionals at the planning stage of the modernization. The success of AHP requires well defined modernization solutions. Without specific solution proposal the technologies or solutions might be difficult to compare.

Evaluation of the suggested process framework was conducted after the demonstration. The main goal of set for the artefact was reached, and most of the sub-goals set to the artefact were also met. In addition to this, the artefact sets some guidelines on the modernization approaches that the organization should consider. The limitation in the context of functionality is the limited tools offered for argumentation. It is not practical to provide exact methods for modernization argumentation since all project are different in nature. When it comes to getting resources allocated for modernization, cost estimation tools such as COCOMO and ROI calculations were considered useful. In most of the cases, these methods can be too resourceful compared to the value they provide.

The process suggested of course also has some shortcomings, since one of the key requirements was for the process to be light. The process does not contain any modernization process models if the organizations decide to re-engineer their system. It was left for the organization to decide the re-engineering approach with the system vendor, since it is not assumed that these types of organizations would conduct re-engineering independently.

One of the research limitations of this thesis was the demonstration and thus the evaluation of the artefact. The model was implemented only on one example project. The effectiveness of the process model was excellent in this particular project, defining the real effectiveness would require multiple process implementations by different organizations and people. Process implementation by different people would allow the discovery of possible issues with the MODEST scale, as well as required knowledge of technical modernization solutions and approaches in AHP. Another interesting aspect to study would be to complete the suggested modernization process for one system by multiple different facilitators and system

experts. This type of experiment would expose, if the process model does not provide consistent analytical results despite of the people using it.

Low/no-code platforms can be seen as a solution to some of the problems identified during this thesis. Many of these platforms were considered when selecting the modernization solutions to be selected from. Platforms such as Microsoft's Power Apps, Mendix and Appian seem to have enough functionality to create simple tools. In the case project, these platforms were not seen suitable since the tool is planned to be integrated to multiple other enterprise systems. Integrating could be difficult depending on the interfaces available on these platforms. These platforms come with the benefits of independency, fast development, and resource efficiency. Low/no-code implementation allows the organization to develop their own tools without extensive software engineering knowledge, while still getting the benefits of modern software architecture. In addition, Microsoft's Powers Apps seems to include support functionality for migrating old Access and Excel databases/tables to the Power platform.

8 Summary

The purpose of this thesis was to study the development, maintenance and use of legacy systems and tools in small engineering organizations and develop an artefact to support these organizations in system modernization. The issues found related to legacy systems were discovered by interviewing three different engineering organizations. Design science was applied in this thesis, which included the creation of a design science artefact. The artefact created during this thesis was the modernization process framework. The main design goal for the artefact was to provide a process framework, that first helps the organizations identify and evaluate the problems in their legacy system and secondly guides the organization to select the correct modernization strategy. The artefact also had smaller sub-goals, that were used to guide the design to fit the target organizations better. The artefact proposed in this thesis contained two processes from previous work: MODEST and AHP. In addition to these, the process suggested contains steps for modernizations initiative and argumentation. The demonstration of the artefact was conducted by applying it to the modernization decision making of the sizing tool in Process and Engineering organization at Metso. The artefact was evaluated utilizing the FEDS evaluation. Finally, the findings, limitations and future work around this topic was discussed.

In conclusion, issues caused by legacy systems and tools were identified at different organizations. The issues discovered from previous work and organization interviewed were similar. Demonstration and evaluation showed that the process framework suggested for the decision-making process fulfilled the design goal set to it and that it fits to the small and medium sized organization it was planned to be used at. The contribution presented in this thesis compliments the complex modernization processes designed to be implemented in large project well and should help small projects to be evaluated quickly and efficiently. Future work related to this topic would include more extensive implementation and analysis of the process framework in other organizations.

References

Abdellatif, M. et al. 2021. A taxonomy of service identification approaches for legacy software systems modernization. The Journal of systems and software Vol 173. 110868.

Agilar, E.et al. 2016. A Systematic Mapping Study on Legacy System Modernization. Proceedings of the 28th International Conference on Software Engineering and Knowledge Engineering. San Francisco, California.

Anquetil, N. et al. 2007. Software maintenance seen as a knowledge management issue. Information and software technology. Information and software technology. Vol 49 (5). 515–529.

Assunção, W. K. G. et al. 2022. Analysis of a many-objective optimization approach for identifying microservices from legacy systems. Empirical software engineering: an international journal. Vol 27 (2).

Appian- Home page. 2023. https://appian.com [website] (accessed 14.4.2023)

Azani, C. 2009. A multi-criteria decision model for migrating legacy system architectures into open system and system-of-systems architectures. Defense AR journal. Vol 16 (3).

Bennett, K. & Rajlich, V. 2000. Software maintenance and evolution: a roadmap. International Conference on Software Engineering: Proceedings of the Conference on The Future of Software Engineering; 04.-11. June 2000. 73–87.

Bhatt, P. et al. 2006. An influence model for factors in outsourced software maintenance. Journal of software maintenance and evolution. Vol 18 (6). 385–423.

Brodie, M., & Stonebraker, M. 1995. Migrating legacy systems: Gateway, interfaces & the incremental approach (Morgan Kaufmann Series in Data Management Systems). San Francisco: Morgan Kaufmann Publishers Inc.

Chen, J.-C. & Huang, S.-J. 2009. An empirical analysis of the impact of software development problem factors on software maintainability. The Journal of systems and software. Vol 82 (6). 981–992.

Dehaghani, S. M. H. & Hajrahimi, N. 2013 Which factors affect software projects maintenance cost more? Acta informatica medica. Vol 21 (1). 63–66.

Harrison, M. S. & Walton, G. H. 2002. Identifying high maintenance legacy software. Journal of software maintenance and evolution. Vol 14 (6). 429–446.

Hevner, A. R. et al. 2004. Design Science in Information Systems Research. MIS quarterly. Vol 28 (1). 75–105.

IEEE. 1993. IEEE Std 1219-1993: IEEE Standard for Software Maintenance.

IEEE. 2012. ISO/IEC 19506:2012: Information technology — Object Management Group Architecture-Driven Modernization (ADM) — Knowledge Discovery Meta-Model (KDM)

International Standard Organization. 2001. ISO/IEC 9126-1 Software Engineering – Product Quality – Part 1: Quality Model.

Kaur, U. Singh, G. 2015. A review on software maintenance issues and how to reduce maintenance efforts. International Journal Of Computer Applications. Vol 118 (1).

Khadka, R. et al. 2014. How do professionals perceive legacy systems and software modernization? Proceedings of the 36th International Conference on software engineering. 2014 ACM. 36–47.

Khanna, S. et al. 2017. Software Maintenance: Challenges and Issues and Models for Reducing the Maintenance Cost. International journal of advanced research in computer science. Vol 8 (3).

Knoche, H. & Hasselbring, W. 2018. Using Microservices for Legacy Software Modernization. IEEE software. Vol 35 (3). 44–49.

Koskinen, J. et al. 2005. Empirical study of industrial decision making for software modernizations. 2005 International Symposium on Empirical Software Engineering, 2005.

Koskinen J. et al. 2008. Developing support for industrial software evolution decision making. Communications in dependability and quality management. Vol 11. 98-106.

Koskinen, J. 2015. Software Maintenance Costs. University of Eastern Finland, School of Computing.

Langer, A. M. 2016. Guide to Software Development Designing and Managing the Life Cycle. 2nd ed. 2016. London: Springer London.

Marounek, P. 2012 Simplified approach to effort estimation in software maintenance. Journal of systems integration (2010). Vol 3 (3). 51-63

Meriam-Webster Dictionary. 2023. https://www.merriam-webster.com/dictionary/legacy [website] (accessed 22.2.2023)

Mendix. Mendix - Homepage. 2023. https://www.mendix.com/ [website] (accessed 14.4.2023)

Metso. Metso – About us. 2023. https://www.metso.com/corporate/about-us/ (accessed 25.5.2023)

Microsoft. Build applications faster - with fewer resources. https://powerapps.microsoft.com/fi-fi/ [website] (accessed 12.5.2023)

Object Management Group. 2003. Why do we need standards for the modernization of existing systems? OMG Task Force.

Object Management Group. 2014. Model Driven Architecture (MDA): MDA Guide. Revision 2.0.

Peffers, K. et al. 2007. A Design Science Research Methodology for Information Systems Research. Journal of management information systems. Vol 24 (3). 45–77.

Perez-Castillo, R. et al. 2011. Reengineering Technologies. IEEE software. Vol 28 (6). 13–17.

Perez-Castillo, R. et al. 2012. Software modernization by recovering Web services from legacy databases. Journal of software: evolution and processes. Vol 25 (5). 507-533.

Rajlich, V. T. & Bennett, K. H. 2000. A staged model for the software life cycle. Computer (Long Beach, Calif.). Vol 33 (7). 66–71.

Sharon, D. 1996. Meeting the challenge of software maintenance. IEEE software. Vol 13 (1). 122–125.

Sneed, H. M. 2005. ‘Estimating the costs of a reengineering project, 12th Working Conference on Reverse Engineering (WCRE’05). IEEE.

Sommerville, I. 2016. Software engineering. Tenth edition. Boston: Pearson.

Tacton – homepage. 2023. https://www.tacton.com/ [website] (accessed 14.4.2023)

Tilus, T. 2006. MODEST: Menetelmä tietojärjestelmään kohdistuvien muutospaineiden arviointiin. Thesis. University of Jyväskylä

Venable, J. Pries-Heje, J. Baskerville, R. 2016. FEDS: A Framework for Evaluation in Design Science Research. European Journal of Information Systems 25. Vol 25 (1). 77–89

Weiderman, N. et al. 1997. Approaches to legacy system evoluation. Technical report. Software Engineering Institute, Carnegie Mellon University. Pittsburgh, PA.