The Impacts of Automating Manual Processes in Software Maintenance: A Case Study

University of Oulu
Faculty of Information Technology and Electrical Engineering / Degree Program of Information Processing Science
Master's Thesis
Sanna Kemppainen
2.6.2020

Abstract

The purpose of this study was to find out how automating a software maintenance task affects the software developers and end users of the information system. The study was conducted as a case study in a Finnish IT organization that provides information systems for organizations.

This study focused on software maintenance tasks that are done in the production environments. The main research question was "How has the new automated process affected the stakeholders in the case?" In order to address the research question, the stakeholder groups had to be identified. Two stakeholder groups were defined and two supporting questions were formulated in order to find answers to the main question. The two supporting questions are "How do the software developers experience the changes that took place after automation?" and "How do the customers experience the changes that took place after automation?"

The study utilised triangulation, combining qualitative and quantitative research methods. Qualitative data was collected with interviews and a questionnaire, and quantitative analysis was conducted to the maintenance request tickets with an MMG (Maintenance Model Graph) analysis. The findings of the quantitative MMG analysis were used to support the results found with qualitative methods.

Automating a repetitive maintenance task was found to affect especially the software developers, who experienced the manual task to be time-consuming and arduous. Based on the developer interviews, four factors were found to have been affected by the automation: (1) degree of difficulty, (2) overall workload, (3) incoming maintenance requests and (4) future development.

Customers were affected by the automation indirectly. The new solution was found to provide them with more accurate data and enhanced documentation, but it was also experienced to be arduous to familiarize with. Overall the results from customer questionnaire pointed out that the new solution was experienced as an upgrade. The familiarization will be handled in the case organization by providing training sessions directly to the customer organizations.

The contribution of this study is the additional knowledge it provided about automating the repetitive tasks of software maintenance. As software maintenance is a very expensive part of the SW life cycle, it is beneficial to consider automating some of the most frequent tasks.

Keywords
Automation, Basic Automation, Software Maintenance, Continuous processes

Supervisor
PhD, Adjunct Professor and University Lecturer Raija Halonen

Technical Advisor
PhD Jukka Mononen

Foreword

It has been an exceptional spring because of the ongoing COVID-19 pandemic and the restrictions it has brought to our daily lives. However, every cloud has a silver lining and in this case the partial social isolation has proven to be useful. I have been able to finish this thesis and at the same time finish my master's studies. Finally.

I would like to thank the case organization for making this thesis possible. Thank you all who participated in the making of this thesis, especially the case organization representatives, interviewees and respondents. To all of you working in the reporting team, a big thanks for your exceptional support and help. Special thanks to Aliisa Värttö, with whom I have had the opportunity to engage in the most inspirational discussions and to share my study years at the university. MSc Anneli Karjula, Jukka Uhlgren and Ilpo Kekäläinen, thank you all for taking the time to proofread this thesis (multiple times) and provide constructive feedback. Your comments were most helpful.

Thanks also to my employer and co-workers for their amazing flexibility and support. Because of you, I have been able to combine my work and studies effectively. These have been busy, interesting and fun times.

My supervisor, PhD Raija Halonen, thank you for all of your guidance along the way. The lessons you have given have helped me to enhance my scientific writing and expression. Another person I wish to thank is Jukka Mononen, who has been a great mentor and a huge help. Thank you Raija and Jukka for sharing your expertise with me.

My deepest gratitude goes to my family, for always supporting me and encouraging me to try my best and keep studying. Your persistence, kindness and advice are the best lessons I have ever had. Thank you mother, father and big brother for everything you have done for me, you are the best.

Sanna Kemppainen

Oulu, June 2nd, 2020

Abbreviations

DB	Database
DBMS	Database Management System
IS	Information System
IT	Information Technology
ITIL	Information Technology Infrastructure Library
ITSM	Information Technology Service Management
MMG	Maintenance Model Graph
QoS	Quality of Service
SQL	Structured Query Language
SW	Software

Contents

1. Introduction

The purpose of this study was to analyse the impacts of automation of manual processes in software maintenance. Especially this study aimed to find out how the automated process has affected (1) the software developers, who previously handled the maintenance tasks manually by doing repetitive tasks after version updates and (2) the end users, who contacted the organization whenever the maintenance tasks were required.

Automation is a very current topic in software industry, as phenomena like continuous integration, continuous deployment and DevOps have become more familiar during the last decade (Düllmann, Paule, & van Hoorn, 2018; Leppänen, Kilamo, & Mikkonen, 2015). The modern research on automation emphasizes software testing and deployment, but automating the tasks related to the maintenance of the existing information systems has been given a lot less attention. Due to the ambiguous meaning of both manual processes and automation, it is necessary to define these terms more precisely. In this study, the manual processes concerned multiple (usually digital) tasks that are required to be done by a person in order to achieve the wished results. The term automation was used to present the process of transforming the manual subtasks into machine functions that can be set to run autonomously, this way decreasing the amount of human resources required to do the task. Automation was limited to concern only the automation happening in IT organizations and their continuous operations. The research discussed both the risks and possible downsides of automation but also the benefits of automating the repetitive tasks.

Motivation for this research came from a gap found in previous automation research. Earlier studies propose that the automation of work tasks is a common habit in the IT industry (Cai and Arney, 2018; Krishnan & Ravindran, 2017), but there is a lack of field research or case studies to showcase (1) the concrete workload impacts of automation and (2) software maintenance automation. After searching for scientific articles related to especially the workload of software developers and software maintenance automation, a gap in the research was found. Most of the automation studies had discussed test automation and automating industrial processes, but the software developer point of view was a lot less studied subject. Automating these maintenance tasks may also impact the customer organizations, who are involved in the software maintenance process according to ITIL v3 processes (OGC, Service operation, 2007). This particular case is also personally interesting, as I was part of the team that created this new solution. Significant interest was also brought up by the case organization that is planning to do further automation and additional features to the new solution. Knowing the effects of this solution will aid them in finding the best ways to utilise automation scripts in the future.

Prior research has pointed out that software maintenance is often declared to be the most expensive phase of software life cycle, creating 40-70% of the total costs during the software's life cycle (Mookerjee, 2005; Stavrinoudis, Xenos, & Christodoulakis, 1999). Utilizing automation in any IT operations, especially in change management (Brown & Keller, 2006) and continuous processes, such as deployment (Bartusevics, 2017; Herrick & Tyndall, 2013) has benefited both the service/product provider and customer organizations. Since software maintenance is part of software life cycle and it consumes a lot of resources, the actual benefits of automation should be analysed. Automation

itself, used in IT operations, had been found to affect for example in the profitability, lowering the costs of the organization and also affecting positively on productivity, quality of service and efficiency (Krishnan & Ravindran, 2017).

The study was conducted as a case study in a Finnish IT organization that provides information systems and services to customer organizations. The case organization had recently automated the processes of maintaining database views and creating database documentation for the end users of their information system. The automation was implemented as basic automation, using PowerShell scripts and Java-based documentation software to transform the original manual tasks into machine functions. This study did not discuss cognitive automation, or the use of artificial intelligence and machine learning, but was focused on basic automation. To find out the impacts of automation, qualitative research was conducted in the form of interviews or questionnaires for members from the stakeholder groups mentioned before. The workload of the original maintenance task was also considered with the help of Maintenance Model Graph, or MMG, which is a tool for doing a quantitative analysis and evaluation on the effectiveness of software maintenance (Marques-Neto, Aparecido, & Valente, 2013). Choosing MMG for collecting quantitative data was based on its practicality and suitability for the processes of the case organization.

The research problem was presented as one research question, with two assisting questions (RQ1, RQ2) to support the RQ:

RQ: How has the new automated process affected the stakeholders in the case?

RQ1: How do the software developers experience the changes that took place after automation?

RQ2: How do the customers experience the changes that took place after automation?

Due to non-disclosure agreement (NDA), technical details were not presented in depth in this study. To find answers for the RQ, the study first discussed RQ1 and RQ2 separately. The findings regarding RQ1 and RQ2 were then combined to discuss the RQ. The effects on the SW developers were analysed using qualitative interviews to find information about their own personal experiences. The end users were approached via online questionnaire that utilised both open-ended questions and Likert scale. To support the qualitative findings from the interview and questionnaire, a quantitative MMG analysis was conducted to find out the complexity of the original manual task.

The research contributed to both scientific community, case organization and possibly other organizations where software developers are struggling with continuous manual software maintenance tasks. Impacts that had been noticed after implementing the new automated solution were discussed and both qualitative and quantitative data were provided.

The structure of the study starts by presenting the existing knowledge on automation. As automation is a wide subject, the context of automation was limited to IT organizations and IT operations. The found downsides, benefits and ways to implement automation were showcased. Software maintenance was discussed as a part of software life cycle. Maintenance requests are often triggered by the customers or end users, so the prior literature on IT service management and the most used framework (Alonso, Verdún, & Caro, 2013; Iden & Eikebrokk, 2013; Iden & Eikebrokk, 2016), ITIL, were also presented. After prior literature was discussed, the research methods and data collection methods were explained. The presentation of the analysed data and conclusions based

on the findings were handled in Chapter 5 and Chapter 6 reported the limitations and suggestions for future studies related to the subject area.

2. Prior literature

This section presents the most important existing literature on the subjects related to this research. Background information about automation, software maintenance and IT service management are introduced. Automation is limited to concern only the automation of tasks happening in IT operations because of the vast range of different automation solutions. The ITSM best-practice framework chosen to be handled in this study is ITIL v3, because it is one of the most used frameworks in IT organizations (Alonso, Verdún, & Caro, 2013; Iden & Eikebrokk, 2013; Iden & Eikebrokk, 2016).

2.1 Automation

Automation in general is a very wide concept, as it can range from large industrial machines or production lines to short programming code snippets. The commonality between these devices and code snippets is that they aim to decrease the human workload (Groover, 2019). In prior literature, automation is often described to be the process of replacing human tasks with automatic devices and computers (Bainbridge, 1982; Tschiersch & Brandt, 1996). The nature of these tasks can be related to manual control, planning or problem solving (Bainbridge, 1982). Groover (2019) defined automation to be a technology that is concerned with handling a certain process with the means of programmed commands with automatic feedback to ensure correct execution of the commands. Groover (2019) also described three essential pieces of an automated system; (1) a source of power, (2) feedback controls and (3) machine programming. In modern discussion about automation, computers and computer processors are at the center of attention, as digital computers and algorithms have opened up new possibilities for more efficient automation (Janssen, Donker, Brumby, & Kun, 2019), which has also made automation processes more complex (Groover, 2019). Right now especially cloud computing and machine learning are boosting the automation trend (Krishnan & Ravindran, 2017) and there seems to be no reason to assume that the habit of automating would be slowing down in the near future.

Automated tasks can sometimes be said to be autonomous (Janssen et al., 2019; Limoncelli, 2018). Automation gains autonomy if it can be fully run without any effort from human operator (Limoncelli, 2018), for example in a situation where a computer script is set into a system where it runs in certain intervals, without human operator specifically triggering it.

2.1.1 Automation in IT operations

According to many automation definitions (Bainbridge, 1982; Tschiersch & Brandt, 1996; Groover, 2019), information systems and tasks done with the help of computers can be considered to be automation themselves and this brings us to a question about the very essence of automation. In Finnish language, the old term for information technology can be translated to “automatic information processing” (Automaattinen tietojenkäsittely, ATK). However, even after computers were introduced to different working environments, for example in manufacturing plants, the employees still had to do certain tasks in order to achieve their goals despite the nature of the tasks turned from manual to digital (Groover, 2019).

Modern automation, especially in IT industry, has evolved from automating manual tasks to automating digital tasks, or tasks done with the computer. Modern automation research shows that IT organizations are interested in turning the repetitive or time consuming digital tasks into automated or autonomous processes (Krishnan & Ravindran, 2017). For example automated solutions related to change management (Brown & Keller, 2006), software testing (Thummalapenta, Sinha, Singhania, & Chandra, 2012) and deployment (Düllmann, Paule, & van Hoorn, 2018) have been studied a lot. Also new organizational cultures emphasizing automation have been defined during the last two decades, the most famous likely being DevOps, which utilises automation to enhance the communication, continuous integration, quality assurance and software delivery (Jabbari, bin Ali, Petersen, & Tanveer, 2016).

This trend is emerging due to (1) larger and more complex information systems and their environments, causing the workload in IT organizations to grow (Brown & Keller, 2006) and (2) the beginning of the so-called "post-agile era" (Leppänen, Kilamo, & Mikkonen, 2015) where the principles of agile development have been met, but new issues are found, often regarding quality assurance and the deployment speed. These two factors combined will eventually lead the organization to a situation where the human resources are stuck with repetitive and time consuming processes instead of providing new value.

The automation happening in IT organizations can be divided into two categories, basic automation and cognitive automation (Krishnan & Ravindran, 2017). Basic level of automation is achieved with scripts and small changes in the workflow, with the aim of reducing the repetitive tasks of employees. Cognitive automation, however, aims to provide tools for decision making or information-heavy processes, utilizing big data, artificial intelligence and machine learning (Bruckner, Zeilinger, & Dietrich, 2011; Krishnan & Ravindran, 2017).

Software development departments often have some scripts which they can run to speed up processes that are related to continuous services, or in other words, the processes that take place during continuously updating the environments (Bartusevics, 2017). Bartusevics' study, including answers from more than 35 ICT companies of Latvia of which 60% are large companies with 200 or more employees, shows that the most popular scripting language among these organizations is Unix Bash/Shell, with Windows Bash as the second. Figure 1 contains the full results of the questionnaire according to Bartusevics' (2017) results.

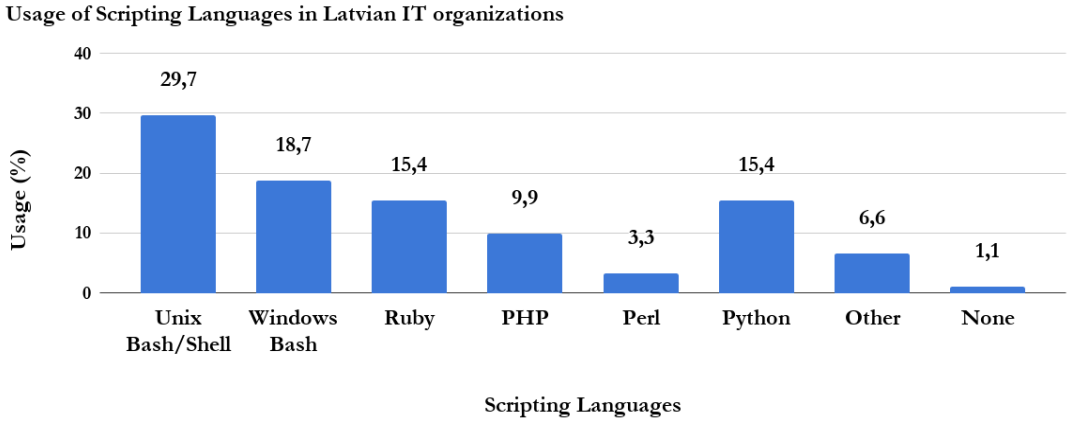

Figure 1. Most popular automation scripting languages in IT organizations of Latvia. (Bartusevics, 2017)

From Figure 1 it can be seen that among the studied Latvian IT organizations there are also some companies that do not use scripts in their continuous processes at all. Also the level of automation in IT organizations has been a subject of interest in modern research in IT field. Figure 2 (Bartusevics, 2017) describes the level of automation in continuous processes of Latvian IT organizations.

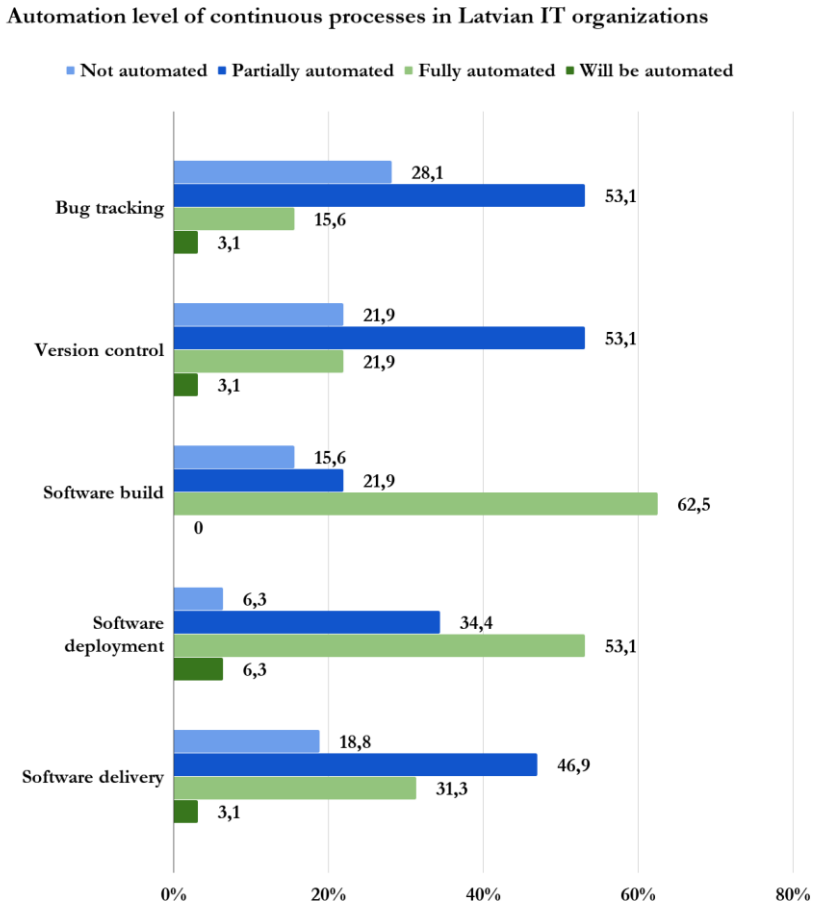

Figure 2. Automation levels of continuous processes in Latvian IT organizations. (Bartusevics, 2017)

As reported in Figure 2, the builds and deployments are the most popular processes to be fully automated, with version control, bug tracking and software delivery being usually only partially automated. These continuous processes presented in Bartusevics' (2017) study are, according to ITSM best-practice framework ITIL, part of the so-called service transition processes (OGC, Service transition, 2007). From all IT operations, apart from service operation processes like access or incident management, these are found to be the easiest processes to automate by basic automation scripts (Krishnan & Ravindran, 2017).

The IT organizations who do not use automation in their processes base their decision on the risks that occur when utilizing new tools, refactoring existing architectures or altering the structure of the software or databases. However, given the chance to automate processes free of charge and with the possibility to return to using the old solution, organizations were found to be much more open to automation. (Bartusevics, 2017.)

2.1.2 Known effects of automation in IT operations

A very much cited article on automation is Lisanne Bainbridge's *Ironies of Automation* (1982), discussing potential problem areas and paradoxes of automation. Even though the paper is almost 40 years old and focuses mainly on process industry instead of information technology, there are multiple similarities to modern day automation. For example, Bainbridge (1982) expressed worry towards the potential loss of human expertise, or manual control skills, as automation is implemented. These manual control skills are developed when the task is repeated and it becomes a normal routine to the operator. In IT industry such task could be version deployment. After automating the said task, the operator still has two tasks left: (1) monitoring the automation and (2) possibly intervening it if something goes wrong. The irony behind this is that after the task has been moved away from the operator, their manual control skills might decrease and if the automation fails, they will have to do a non-routine tasks to get the situation under control. These non-routine tasks often require specific skills or expertise on the automated process. (Bainbridge, 1982.)

The ironies of automation have been discussed and analysed multiple times after Bainbridge's (1982) original paper, for example study by Strauch (2017), which discussed whether these ironies can still be found in modern automation. To analyse the situation of automation, the main "ironies" behind automation were divided into two problem areas, (1) human operators are integral to sociotechnical systems and thus removing them from automated process does not fully erase the possibility for human errors and (2) attempting to remove human operator from the original task, even with fully automated and autonomous systems, still leaves the responsibility of monitoring and fixing potential automation problems to a human operator. (Strauch, 2017.)

Automation adds hidden complexity to the process (Brown & Hellerstein, 2005). While the logic behind the automation should complement the original manual process, it is often integrated with multiple levels, which slows down the process of detecting and recovering from error situations (ElMaraghy, ElMaraghy, Tomiyama, & Monostori, 2012; Brown & Hellerstein, 2005). Automation is also known to take time when it is developed and the amount of time may vary a lot, depending on the extent of design, testing and implementation (Brown & Hellerstein, 2005). In order to start implementing automation in IT operations, some preconditions have to be taken into account. The preconditions may differ, depending on the automation scope and tasks to be automated. The factors known to affect the implementation of automation are the overall mindset towards automation in the organization (Limoncelli, 2018; Bartusevics, 2017) and understanding the existing processes (Brown & Keller, 2006; Limoncelli, 2018).

Even though being an expensive process that may increase the complexity of the process, automation has been found to be very useful when implemented in IT operations. Krishnan and Ravindran (2017) studied the impacts of IT service management automation and found the automation to positively affect to for example productivity, QoS (quality of service) and IT service efficiency. Similarly Brown and Keller (2006) conducted a case study in which they automated change management by utilizing knowledge of ITIL best practices and scoping their automation towards the most critical tasks included in change management. The perceived benefits of automation can be found in Table 1.

Table 1. Found benefits of automation.

	Krishnan & Ravindran, 2017 (IT service management)	*Herrick & Tyndall, 2013 (Deployment)*	*Brown & Keller, 2006 (Change management)*
Benefits for the service or product provider	Perform value added activities rather than simple repeated tasks Increase in efficiency and effectivity of IT services	Reduced effort and inconsistencies in deployments Reduced time usage and workload on IT administrators	Standardized processes Substantial time savings, for example 33% reduction in the installation time of a complex Java based enterprise application Automation generates reusable automation assets, which may reduce the long-term costs of future automation processes
Benefits for the customer organization	Benefits from improved Quality of Service (QoS)	Faster service Frequent updates and consistent builds	Faster deployments and decreased time to resolution in change requests

Although the papers written by Krishnan and Ravindran (2017), Herrick and Tyndall (2013) and Brown and Keller (2006) all had different scopes and subjects for automation, the benefits found in their studies are somewhat similar. Herrick and Tyndall (2013) automated software deployment tasks and their research discussed the impacts of automating the tasks related to deployment process. Brown and Keller (2006) automated the change management processes and Krishnan and Ravindran (2017) focused on ITSM. The benefits for the service or product provider can be seen to revolve around value creation and more effective resource usage. For example time savings have been reported by both Herrick and Tyndall (2013) and Brown and Keller (2006). This means that instead of using a lot of time on repetitive deployment or change management tasks, the developers can focus on value creating activities. It is also important to notice the standardization and future automation assets reported by Brown and Keller (2006), because one downside of automation has been said to be the costs and the time it tasked to build a working automated solution (Brown & Hellerstein, 2005). After the first task is automated, it will be easier to start automating other processes with the existing artefacts and reusable assets. This means that the cost

of automation may vary, depending on how much automation is already in use for similar tasks (Brown & Keller, 2006).

For the customer organization the benefits found are varying, but mostly concerning the quality and speed of service. The variance of the benefits for customers is wider than for the service/product provider. This is due to the difference of the automation scope and subject; automating tasks in service management do not necessarily make the deployment times faster, but they enhance the overall QoS, whereas automating change management leads to shorter time-to-resolution in customer tickets. (Brown & Keller, 2006; Herrick & Tyndall, 2013; Krishnan & Ravindran, 2017.)

2.1.3 Strategies for implementing automation

There are multiple recommended ways to automate a known process. One strategy, presented by Tschiersch and Brandt (1996) consists of three phases: (1) Modelling the problem to match the human operator's mental models, (2) choosing an appropriate mathematical algorithm and describing the problem through the algorithm, and (3) choosing the appropriate computer language, and translating the algorithm into this language. This presented automation strategy focuses on creating an equivalent machine function based on the original human function with the help of mental model diagrams and mathematical models (Tschiersch & Brandt, 1996). A more detailed versions, including task identification and documenting the automation implementation process are presented by Brown and Keller (2006) and Limoncelli (2018).

A case study conducted by Brown and Keller (2006) showcased a process-oriented insight to the automation implementation. They approached the problem from the best-practice point of view. Their strategy consisted of six phases:

1. Identify the best practice processes for the domain that will be automated
2. Define the scope of automation
3. Identify delegation opportunities
4. Identify links between delegated activities and external activities
5. Identify, design and document induced processes whenever required
6. Implement automation for the process flow

To identify the best practices (phase 1), Brown and Keller utilised ITIL. This choice was based on their problem domain, which was automating the IT service delivery processes. ITIL provides a framework of best practices to support the said processes but does not discuss or give guidance about automation. To decide the processes to automate, the organization should possess a deep understanding of the costs of different activities. The main targets for implementing automation should be the most expensive activities. If such information is not available, it is also suitable to focus on key domains in daily operations. The scope of automation (phase 2) should be limited and for this reason it is recommended to divide the larger processes into small subsets. From these subsets it is easier to start the automation. After automating one smaller task, there will often be reusable functions that can be utilised in the future automation as well. Delegating some tasks from the automation (phase 3) means that the original workflow should be analysed and the possible automation targets identified. Also the effort and used resources should be considered and compared to the possible benefits gained by

automation, as all automation is not necessary or suitable. The cost of automation should never exceed the gains. The links between new automated solution and external activities (phase 4) should be clarified to make sure everything still works together. New data formats, APIs and required new codifications and standards need to be identified. The new automated solutions also requires new induced processes that need to be documented and designed (phase 5). Activities such as error recovery and automation maintenance should be identified. There should be enough documentation to ensure that the automation will work in the future and that it can be updated or maintained to suit in possible future requirements or changes. In Brown and Keller's study, the final step to automation is implementing the automated solution to the process flow. The difficulty of this last phase depends highly on the scope and complexity of the automation and also the quality of work in earlier phases. (Brown & Keller, 2006.)

Limoncelli (2018) suggested documentation to be used as a tool for automation. Their recommendation for automating tasks consisted of four phases, which all include careful planning and iterative structure of the process: (1) Documenting the original task, (2) creating automation equivalents, (3) creating the automation and (4) turning the automation into autonomous system. The first phase, getting familiar with the manual task, requires either existing knowledge on the process or exploring the task by doing it multiple times. The existing knowledge can be found by interviewing the experts or reading operation manuals, if such artifacts exist. Instead of focusing on mental models, like Tschiersch and Brandt (1996), Limoncelli suggested that instead of molding the mechanical task to fit into human mental models, the automation designer should change their approach to reflect computer processing unit: What subtasks do you have to do in order to get the process done? How many different tasks are there? The goal is to produce a clear written documentation about the current process to be able to succeed in the automation. However, Limoncelli (2018), emphasizes that it is not necessary to spend a lot of time on this phase, as the written document does not need to be perfect or very detailed. It is important to get fast into the second phase, which is very iterative and might take a lot of time.

The phase of "augmented documentation" (Limoncelli, 2018) consisted of turning the subtasks into machine functions. The process is still done manually, but the designers are creating command-line snippets or other machine-run pieces to take the necessary steps towards automation. These commands are added to the written documentation in order to avoid the hidden complexity or, as Tschiersch and Brandt (1996) express it, the chaos of automation. This phase also contains quality assurance and testing, as the designers need to run these automated subtasks in such a way that reveals the possible edge cases and bugs. Limoncelli's (2018) third phase can be compared to the third phase of Tschiersch's and Brandt's (1996) third phase, as it also includes the choice of programming or scripting language. After the individual snippets are tested and possible bugs are resolved, the pieces must be put together. Usually this means that the designers must choose whether to create an automation script or a standalone program to fully automate the task that was formerly done manually. The choice depends on the scope of automation, possible existing systems and also the preferences of the designers, as translating the existing snippets to another language may be time consuming and, eventually, unnecessary step, if the existing code lines can be easily combined into an efficient script or program.

The fourth step that exists only in Limoncelli's (2018) strategy, is setting the automation to be autonomous. This is not a mandatory step, as a process can be said to be automated without it being autonomous. However, if the task is done so often that it can be considered to require too much human resources, it is possible to set it run autonomously. This often happens by integrating it with an existing system or using a

modern DevOps/Continuous Integration tool, such as Jenkins, to set it run in certain intervals. (Limoncelli, 2018.)

2.2 Software maintenance in software life cycle

Similarly to other products, software products have their own life cycle stages, which consist of conception, development, production, utilization, support and retirement. These six stages occur during the life cycle of a software product, and each stage contains some general processes that are presented in ISO standard 12207:2017. Software maintenance is described as a technical process, alongside with other processes such as validation and transition, or the testing and deployment processes of a software system. Each life cycle process can be invoked at any point throughout the software life cycle, depending on the project requirements, chosen development model and the general context. This means that for example maintenance activities can occur in any of the six stages presented in Table 2, but they are required in the support stage. (ISO/IEC, 2010.)

Table 2. Software life cycle stages. Based on ISO/IEC TR 24748-1 (2010).

SW life cycle stage	**Purpose**
Conception	• Finding and assessing business opportunities • Defining system requirements and design solutions
Development	• Developing a system (prototype) that meets stakeholder requirements and can be produced, tested, evaluated, operated, supported and retired
Production	• Producing or manufacturing the system • Testing the system • Producing supporting systems as needed
Utilization	• Operating the system • Delivering services • Ensuring continuous operational effectiveness
Support	• Providing logistics, maintenance, and support services
Retirement	• Removal and archiving of a system and related services

There are multiple different models with which the organization can execute the processes related to each life cycle stage. Such models are, for example, waterfall model, incremental model, spiral model and iterative model. The decision to implement a certain model depends highly on the nature of the software project and its objectives. The sequence of common life cycle processes varies when utilizing different models. (International Organization for Standardization, 2017.)

As stated before, software maintenance activities are part of software life cycle processes (International Organization for Standardization, 2017). Releasing the software

does not mean that the developers can turn their full focus on developing new software, but the existing software will require active monitoring, changes and fixing (Singh, Bhatia, & Sangwan, 2009; International Organization for Standardization, 2006). Studies have shown that the maintenance phase is actually the most expensive phase in software life cycle, taking 40-70 percent of the total software life cycle costs (Stavrinoudis, Xenos, & Christodoulakis, 1999). In addition to the high costs of maintenance activities, the developers working in IT organizations often find such tasks to be boring and laborious (Mookerjee, 2005).

The need for maintenance activities may be triggered by multiple specific reasons, varying from failures in implementation or performance (Singh and Goel, 2007) to adapting the system to an entirely new operating environment or newly added requirements (Basili, 1990). Roughly the change requests can be said to be focusing either on correction or enhancement (International Organization for Standardization, 2006), meaning that they either repair something that is found to be broken (correction) or add a whole new requirement that needs to be implemented to the system (enhancement).

ISO/IEC 14764 standard divides maintenance activities into four different groups; (1) corrective, (2) preventive, (3) adaptive and (4) perfective. These four maintenance types can be further classified into correcting and enhancing activities. Figure 3 presents the structure behind software maintenance task, originating from change request usually triggered by the end user of the software.

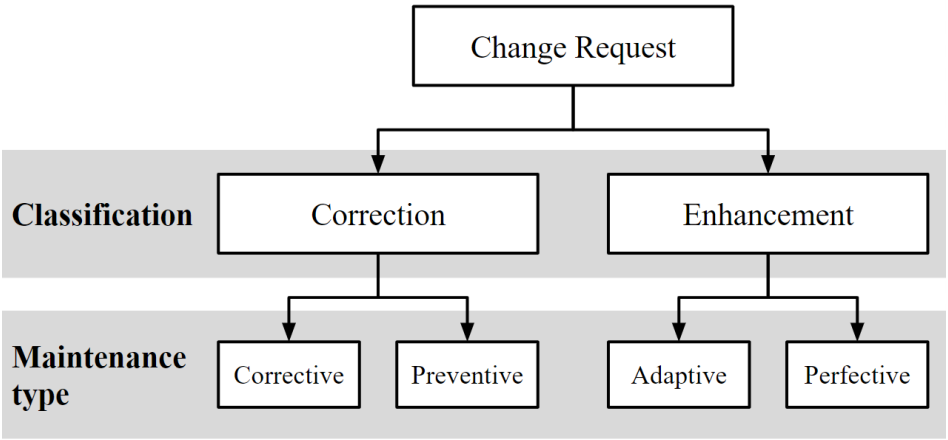

Figure 3. Maintenance types and their classifications. (International Organization for Standardization, 2006)

As reported in Figure 3, a change request can be classified to be either focusing on correction, fixing some noticed failure, or enhancing, aiming to add something to the existing software or changing some part of it in order to make the software suit better for its purpose or enhancing its performance in some other way. Maintenance classified as correction may also be referred to as an emergency maintenance, if the failure within the system is found to be critical and it causes a blockage in the use of the system. Enhancement maintenance includes changes that are not listed in the original requirements of the system. (International Organization for Standardization, 2006.)

The occurrence between different maintenance types varies. Lientz and Swanson (1980) found out that the most usual maintenance task is perfective. In other words the most typical change request aims to make the software easier to maintain in the future or improves its performance (International Organization for Standardization, 2006). The

distribution of different maintenance tasks by type can be seen in Figure 4, which is a visual representation of the maintenance type distribution by Lientz and Swanson (1980).

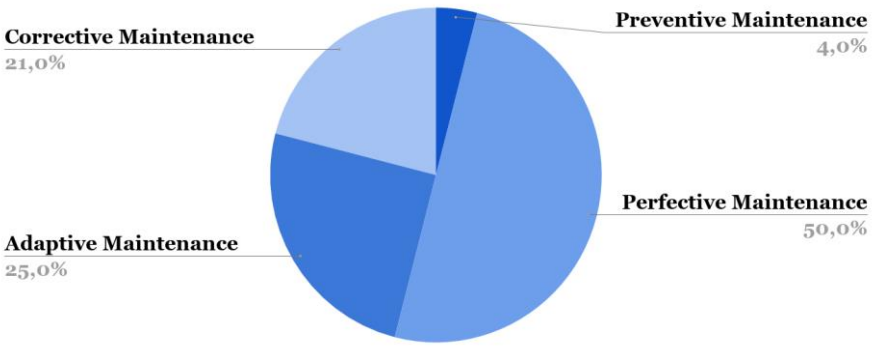

Figure 4. Division of different maintenance tasks by type. (Lientz & Swanson, 1980)

According to the study by Lientz and Swanson (1980), the least frequent type of software maintenance task is preventive, with only 4% occurrence in all software maintenance tasks. This means that software is rarely altered with the goal of better error detection in the future. Half of all maintenance tasks are classified as perfective, which means that the changes done to the SW are not included in the original requirements specification. The reason for perfective maintenance might be, for example, a change in the environment or the tasks that are performed with the system. (Lientz & Swanson, 1980; International Organization for Standardization, 2006.)

In a more recent case study conducted by Hatton (2007), the division of maintenance types was analysed in a small IT organization. Figure 5 presents a similar pie diagram from Hatton's (2007) case study research.

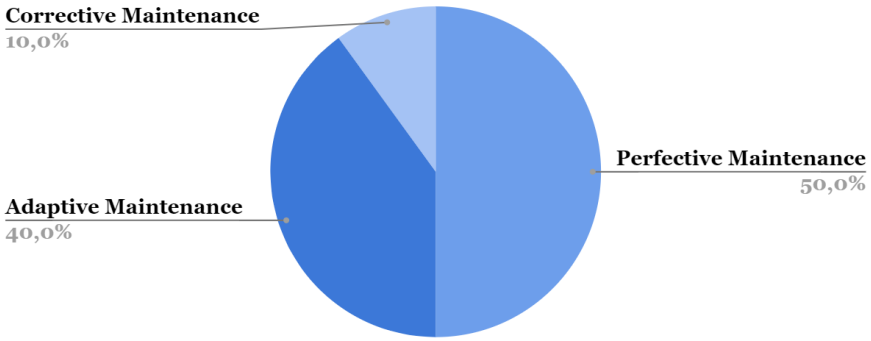

Figure 5. Division of SW maintenance types in small IT organization. (Hatton, 2007)

The findings were quite similar as in the study by Lientz and Swanson (1980), but Hatton (2007) had limited their analysis to three maintenance types; corrective, perfective and adaptive. As presented in Figure 4, half of the maintenance tasks were found to be perfective, similarly as in Figure 3. Adaptive maintenance, or the maintenance with which the software is altered to suit for new tasks or new integration platforms, was found to cover 40 percent of the change requests. Only ten percent of the tasks were found to be corrective by their nature.

2.3 IT service management using ITIL

IT service management, or ITSM, is a process-focused discipline which aims to optimize IT services and create business value with different standards, processes, frameworks and methodologies (Cusick, 2020; Galup, Dattero, Quan, & Conger, 2009; Gunawan, 2019; Mesquida, Mas, Amengual, & Calvo-Manzano, 2012). In the field of information technology, the importance of service quality and strategic planning is higher than ever and the management tasks are getting more complex. This is happening because the number of assets, such as hardware, software and human resources is rising (Gunawan, 2019) and thus organizations are in need of clearly standardized processes in order to be able to fulfill the service requirements of their customers.

There are multiple frameworks to support the service processes of IT organization, for example ITIL, COBIT (Control Objectives for Information and Related Technology), DMTF (Distributed Management Task Force) and ISO / IEC 20000 to name only few. Although their contents are different, these frameworks do serve the same goals; to provide tools for the IT product and/or service provider to answer to demand with strategic approach and to provide quality services to their customers (Alonso, Verdún, & Caro, 2013; Gunawan, 2019). A highly popular choice for a process framework is especially ITIL, a trademark owned by AXELOS Limited (Alonso, Verdún, & Caro, 2013; Iden & Eikebrokk, 2013; Iden & Eikebrokk, 2016). Even though it is impossible to provide the accurate and current numbers of organizations using ITIL best practices, in year 2016 itSMF, an international organization promoting ITIL, had around 6000 member organizations and over 70,000 individual members all over the world (Iden & Eikebrokk, 2016).

ITIL, originally an abbreviation of Information Technology Infrastructure Library, is a documented set of best practices describing the ITSM processes and helps organizations to realize and organize their assets (Gunawan, 2019; Krishnan & Ravindran, 2017; Brown & Keller, 2006). ITIL was first introduced in the 1980's and the first version of IT was used between 1989 and 1995 (ITSMF, 2007). The most recent version is ITIL 4, which was introduced in 2019 (Gunawan, 2019). Because the ITIL 4 has only recently been published and it is expected that it takes time for organizations to change their processes, this study focused on the ITIL v3, which has been the base of best practices from 2007 to 2019 (Gunawan, 2019).

ITIL v3 presents four categories: (1) Service strategy, (2), Service design, (3) Service transition and (4) Service operation, with a later addition of fifth volume, Continual Service Improvement (Gunawan, 2019; Eikebrokk & Iden, 2012; Krishnan & Ravindran, 2017). Each category has a set of processes, varying from five to eight processes per category.

Service strategy

Service strategy processes are designed to support the existing business strategies of the organization. They help the organization to create value to customers and to evaluate the Quality of Service (QoS). The value provided to customers depends on how useful the current services are and whether they fulfill the customer requirements. For this reason it is very important to create a service strategy where the necessary services and required resources are defined. This service strategy will be a part of the organization's IT strategy. (OGC, Service strategy, 2007.)

Service design

The processes under service design are crucial when defining which services will be used to support the service strategy. The aim is to provide the customers with high QoS and support all the possible requirements to create as much value to the customers as possible. The ideal result is to have high quality services that would last throughout the service life cycle, without many changes. (OGC, Service design, 2007.)

Service transition

Whenever changes, additions or removals are made to the services, it is important to make sure that the services still support the original requirements and both the service and business strategies. For this reason ITIL v3 includes service transition processes with which it is possible to keep track of the changes that are taking place and also evaluate the necessity of the changes. Service transition processes ensure that the services will be in accordance with the requirements specified in the service design. (OGC, Service transition, 2007.)

Service operation

Service operation processes are continuous daily operations to maintain the agreed service level in order to create customer satisfaction. These processes happen directly with the customer and often yield a lot of useful information to the organization providing the services. Service operation can be classified to be either reactive or proactive, which means that the service provider can be either slightly passive and react only when they are being approached by the customer (reactive) or they can actively seek to be in contact with the customer frequently (proactive). Often service operation is seen to be a mix of both, executing processes reactively and proactively. (OGC, Service operation, 2007.)

Continual service improvement (CSI)

The fifth ITIL v3 volume presents seven steps with which the IT organization can ensure that their current services answer to demand and fulfill the requirements that are set to them. By executing the seven steps, the organization might be able to predict possible changes happening in the future or find some shortcomings in their current services. The need to enhance the existing services is based on three to five key performance indicators (KPI's) that can be evaluated by the service provider. It is important to compare the current situation to the situation after implementing the change. This way it is possible to find out whether the effect is positive or negative. The seven steps of CSI are presented in Figure 6. (OGC, Continual service improvement, 2007.)

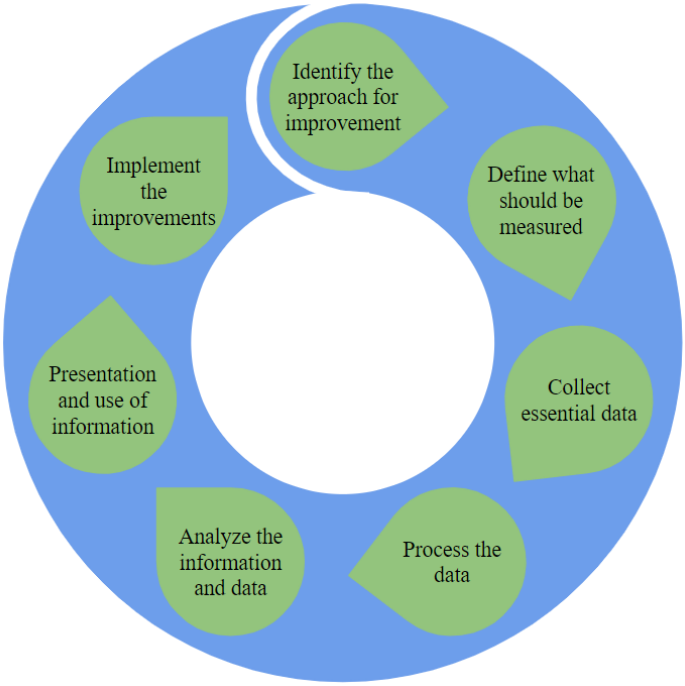

Figure 6. The seven steps of Continual service improvement described in ITIL v3. (OGC, Continual service improvement, 2007*)*

Table 3 presents the ITIL v3 structure, with symbols demonstrating the level of automation that can be implemented to each process. The green square indicates that the process can be automated using basic and cognitive automation, the orange triangle indicates that the process can be partially automated using either basic or cognitive automation, and the red circle indicates that the process cannot be automated at all.

Table 3. ITIL v3 structure with automation potential of each process. (Krishnan & Ravindran, 2017)

■ = Can be automated by basic and cognitive automation ▲ = Can be partially automated ● = Cannot be automated		
Service strategy	▲ ▲ ▲ ● ●	Business relationship management Demand management Service portfolio management Financial management Strategy generation
Service design	■ ■ ■ ▲ ▲ ▲ ▲ ●	Availability management Capacity management Service catalog management Design coordination Information security management IT service continuity mgt. Service level management Supplier management
Service transition	■ ■ ■ ▲ ▲ ▲ ●	Change management Release and deployment mgt. Service validation and testing Change evaluation Knowledge management Service asset and configuration mgt. Transition planning and support
Service operation	■ ■ ■ ■ ▲	Access management Event management Incident management Request management Problem management
Continuous service improvement	▲	7 step improvement process

Krishnan and Ravindran (2017) studied the possibilities to automate different ITIL v3 processes by creating an automation scope matrix and analyzing whether it is possible to use either basic or cognitive automation on each process. In their research they state that the most automation can be implemented to service operation processes, as it is possible to automate 75-85% of the service operation process activities. This finding is based on the statement that service operation processes are often similar and repetitive, following a certain pattern that can be automatized using, for example, an automation script. The least favorable ITIL category for automation is service strategy, where only

10-15% of the activities can be successfully automated. This is because of the basic nature of the strategy level processes; they do not follow any known patterns and they are not executed frequently. Cognitive automation was found to be usable in some tasks regarding demand management and portfolio management, but this would require the use of advanced analytical models and machine learning. (Krishnan & Ravindran, 2017.)

3. Research Approach

This chapter contains information about the chosen research methodologies and approaches, focusing on case study and qualitative research methods. Case study approach and research methods of this study are presented. In this study, method triangulation, or combining different research methods in order to increase the reliability and validity of the research (Hussein, 2009; Strauss & Corbin, 1998), was used for data collection. Qualitative data was collected with the use of interviews and questionnaire and quantitative data was collected by using Maintenance Model Graph by Marques-Neto, Aparecido, and Valente (2013).

3.1 Context

The case study was conducted in a Finnish IT organization that provides services and information systems for their customer organizations. Their information system uses a relational PostgreSQL database and the IS's end users need to utilise database views and documentation in their work. For this reason they are offered a reporting database where they can, for example, create queries against the DB views and collect data.

New version deployments for the IS triggered changes in database columns and tables. Because of the nature of relational database views, this caused problems. The basic PostgreSQL database views are not physically materialized, but they are stored as a query pointing to the core tables (PostgreSQL, n.d.). Even if the CREATE VIEW SQL clause is used with * (selecting all columns), the view will only include the columns that are in the table at the time of view creation. Columns that are added later will not be automatically included in the view definition query. This caused repetitive tasks in the case organization. Whenever the new software version included changes to the core tables, the view queries were outdated and the database view queries did not automatically update to include the new tables or columns. This triggered maintenance requests from the customers, as they were not able to utilise the new tables or columns that were inserted in the version update.

Three stakeholder groups were identified when planning this research. When the event of outdated database views occurred, the end users were the first ones to notice it. They created a service request ticket to the case organization's customer support. After receiving this service request, customer support employees created a change request ticket for the development team responsible for the database views. After considering the width and scope of this research, the focus groups were decided to be software developers and end users, as the automated solution is most visible for these two groups.

The automated solution was developed during September – December 2019 and it was implemented to some customer environments in spring 2020. The automated solution consisted of a PowerShell script that was implemented into existing deployment database scripts. In addition to creating database views to a reporting schema, the script also utilised open source Java-based tool SchemaSpy to create HTML documentation (ER diagram and metadata information) of the newly created database views. This documentation was included to help the customer organizations to build SQL queries to

get information from the database views. More detailed technical information is not available in this research, as it is under non-disclosure agreement.

After implementing the automated view updates and documentation creation, the views were created or updated during each version deployment. This way the stagnation of view updates was eliminated and the end users were able to also use the new columns or tables in their queries.

This research was conducted to find out, how the new solution affected the two stakeholder groups. The research aimed to find out the possible benefits and also difficulties or shortcomings of the automated solution.

3.2 Case study

The approach chosen for this research was a single-case study with positivist paradigm. As suggested by Yin (2017), case study is a favorable approach when (1) the research question is a 'how' or 'why' question, (2) the researcher has small or very small control over the turn of events and (3) the research handles a contemporary phenomenon, a current case. As the research question and its sub questions in this study were all 'how' questions, thus exploratory by their nature (Yin, 2017; Benbasat, Goldstein, & Mead, 1987) and the research subject was a contemporary event of automating a process in single organization, the case study approach was chosen. Case study approach also allows the researcher to gain information and experiment with theories in their real context. This way it is possible to gain the so-called 'face validity', when the scientific theory is implemented into a real case. It gives the researcher a possibility to take into account all the variables, whether relevant or not, and analyse their real-life impact in the research. (Myers, 2019.)

The philosophical paradigm that guided this research was of a positivist tradition. Myers (2019) has proposed that for each research there is an underlying paradigm, or an assumption, that guides the researcher with their work. These paradigms are divided into three classes; positivist, interpretive and critical. Interpretive paradigm focuses on social or psychological meanings or constructions, and it is usually highly preferred in linguistics or social studies, where the context and subjectivity plays a big part. Critical research paradigm aims to challenge the current theories or constructions by aiming to prove the need for changes. Critical paradigm researches often also aim to provide suggestions on how to make these changes. Positivism, on the other hand, focuses on objectivism, usually by proving or disproving existing hypotheses. Positivism is often connected to variables and quantitative research, but it can be used together with mixed methods. (Myers, 2019.)

In this study, there exists a proposition that automation has an impact on three stakeholder groups and their workload and/or resources. This proposition is then analysed in real-life situation using qualitative and quantitative methods.

With case study approach, it is possible to collect data using multiple different methods, including interviews, observation and using archived materials (Järvinen & Järvinen, 2000). In this case study, the chosen data collection methods were surveys, interviews and quantitative MMG (Maintenance Model Graph) analysis following the framework provided by Marques-Neto et al. (2013). Even though the results of case study cannot usually be generalized, it is possible to find similarities when studying the same phenomenon in different cases (Metsämuuronen, 2005; Yin, 2017).

3.3 Qualitative research

Qualitative research methods focus on human behavior, people and settings without reducing them to variables or numbers (Taylor, Bogdan, & DeVault, 2015). Because automation is a tool for reducing human workload and stress (Groover, 2019) and it is found to affect directly to people and their daily work (Herrick & Tyndall, 2013; Krishnan & Ravindran, 2017), this study utilised qualitative research methods to assess the experiences of the software developers. This way it was possible to ask questions that are required to understand the developers' motivations, actions, reasons and the context of their experiences (Myers, 2019). Another justification for qualitative research was the organizational approach, as Pettigrew (2013) suggests that qualitative research is most valuable when it is set into a clear context and focuses on whole processes and their dynamics.

The qualitative methods chosen for data collection in this study were face-to-face interviews and questionnaire for the end users.

3.3.1 Qualitative interviews

The software developers were approached via face-to-face interviews to find out their own experiences and feelings regarding the new automated solution. Due to the COVID-19 pandemic, the face-to-face interviews were conducted remotely via Microsoft Teams. The Teams meetings were recorded and transcribed, as is suggested in the seven stages of the interview by Brinkmann and Kvale (2015).

The purpose of the interview was to find out whether the developers had experienced any negative or positive changes in their workload or in the complexity of the tasks. The interview was designed to (1) compare developers' experiences between old and new solutions and (2) find out any changes in developer workload or task complexity.

An interview can be defined to be either structured, semi-structured or unstructured (Myers & Newman, 2007). During structured interview, the questions are prepared beforehand and the interview will consist of only the prepared questions, without further discussions. Structured interviews can be conducted as a survey, without the presence of an interviewer. A complete opposite of this approach is the unstructured interview, where the questions are not prepared beforehand. In this case the interview can be described to be a freely flowing conversation. The interviewee is allowed to tell their experiences or opinion freely and during this conversation the interviewer may ask personalized questions. Semi-structured interview allows the researcher to predetermine some of the questions but the interviewer can also ask more spontaneous questions during the interview, depending on the conversation flow. (Myers & Newman, 2007.)

For this study, the interview with the software developers was a semi-structured interview. 15 interview questions (Appendix A) were prepared beforehand and each developer was asked the same base questions. In a case where some information could be specified more, the interviewer would ask additional questions for more detailed information.

3.3.2 Customer survey with open-ended questions and Likert scale

Using questionnaires, the researcher can get various information for their study, such as behavior (frequency of use, purpose of use), attitudes (agreeing or disagreeing with certain claims) and feelings (what is perceived as 'good', what is perceived as

'average'). In addition, questionnaires enable the researcher to create demographics based on the respondent variables, such as age, location or job position. (De Vaus, 2002.)

There are two main question types for questionnaires, open and closed format. For open questions the respondents can formulate and write their own answers, whereas for close questions there are predetermined options from which the respondent must choose their answer. It is suggested to use closed format whenever the research is not aiming to find new ideas or thoughts through the questionnaire and the researcher has adequate understanding of what kind of answers to expect. Closed format answers are easier to analyse and they take less time to answer, but they do not allow the respondents to elaborate their experiences or ideas. (Armstrong, 2009.)

Even though Likert scale is often described to be a quantitative tool, as it produces numerical data that can be presented and analysed statistically, it is also a possible choice to support qualitative research (Hussein, 2009). Using Likert scale in qualitative research is suggested, if the data is expected to be either (1) difficult to measure otherwise or (2) sensitive or experienced as sensitive or private (Chimi & Russell, 2009). Likert scale can also be used as a complementary method in qualitative research, as long as the concept makes it possible to create conversation reflecting also the numerical values (Hussein, 2009).

In the questionnaire used in this study, Likert scale was used to aid the customers to rate two different solutions and their experiences on those solutions. The analysis done to these results was, however, qualitative and presented as descriptive statistics because of the low number of respondents and the qualitative concept of the research.

3.4 Quantitative research using MMG

In addition to qualitative interviews and questionnaire, this study used also a quantitative method to collect research data. Quantitative research methods are focused on numerical, measurable data that can be analysed and presented using statistical methods (Creswell & Creswell, 2017). Even though case study research often utilises qualitative methods, quantitative questionnaires can be used as well (Dubé & Paré, 2003). Quantitative method used in this case study is the Maintenance Model Graph presented by Marques-Neto, Aparecido, and Valente (2013).

It is possible to do a quantitative analysis and evaluation on the effectiveness of software maintenance by altering the traditional viewpoint. Instead of comparing maintenance to software development and focusing on lines of code, software maintenance can be seen as a service provided for the customer (Marques-Neto et al., 2013). Software maintenance, happening after the software product is developed and delivered to the customer, differs from other software lifecycle processes. It is triggered by a change request, the tasks require planning and communication and the maintenance process does not follow the basic software development processes but is rather an adaptation of development process (Marques-Neto et al., 2013).

To evaluate the maintenance service, the focus must be on the workflow and the change request status. In order to find out the service time, or time to resolution, Marques-Neto et al. (2013) proposed a model called Maintenance Model Graph (MMG). MMG is created based on the maintenance workflow in the organization, depending on the ticketing system and available statuses of each ticket type. In Figure 7 the basis of MMG is showcased. It represents an imaginary workflow of the change request ticket after it has been created in the service providers ticketing system.

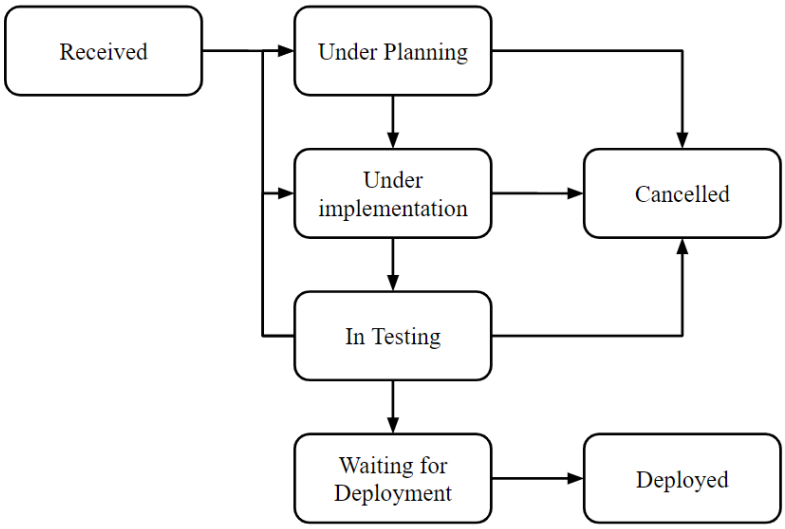

Figure 7. A simplified example of workflow used in MMG.

It must be noted that the maintenance activity may return to earlier stages throughout the process and thus the ticket status may be the same multiple times.

The time used for each change request can be calculated by following the method presented by Marques-Neto et al. (2013): The information about the (1) starting and ending timestamps of each status and (2) number of times the ticket has visited each status must be put on place in the MMG. Using this information, it is possible to calculate three important constants: QueueTime, WaitingTime and ServiceTime.

QueueTime means the time between creating the ticket and the conclusion, whether it is cancellation or deployment. WaitingTime is the time spent on waiting for different actions, for example in Figure 7 the *Received* and *Waiting for Deployment* statuses. ServiceTime is the time it took to execute the software engineering activities, in Figure 7 these tasks are *under planning*, *under implementation* and *in testing*.

From these Marques-Neto et al. (2013) have formulated the following calculation rules:

```
QueueTime = WaitingTime + ServiceTime

ServiceTime = PlanningTime +

ImplementationTime +

ValidationTime +

DeploymentTime
```

, where PlanningTime is the used to understand, to plan and to schedule a software maintenance request, ImplementationTime is the time used for programming and implementing, ValidationTime is the time used for testing and DeploymentTime is the time used for deployment and getting the final acceptance from the customer. It must be emphasized that this method does not aim to calculate the measure of effort as such, but the actual time in hours it takes to fully process a change request. Another notification relating to this evaluation is that the hours should be calculated based on service hours, for example 8-16 instead of whole days. (Marques-Neto et al., 2013.)

Maintenance Model Graph analysis also presents a way to cluster maintenance requests by using a clustering algorithm based on the CRR (Cluster Representativeness Ratio) of the tickets. The clusters are defined by the status occurrences and TTR (time-to-resolution) of the tickets. This is a very useful ability when it is required to analyse a variety of tickets that differ in their complexity levels. Whenever there is a need to compare different maintenance tickets and define their complexity levels, the clustering is a good way to find out the differences between task difficulty levels. (Marques-Neto et al., 2013.)

4. Research Implementation and Methods Used for Data Collection

This chapter presents how the research methods described in Chapter 3 were implemented in this case study and how the collected data was analysed. To support the analysis of the two stakeholder groups, three different data collection and analysis methods were utilised in this study. The groups of interest were approached in different ways, using both quantitative and qualitative methods. Qualitative interviews were conducted for software developers and the data analysis was conducted by inductive qualitative content analysis. The end users were approached with an online questionnaire combining open-ended questions and Likert scale. Quantitative data was collected and analysed from maintenance requests using MMG analysis.

4.1 Interviewing the software developers

As this study aimed to find out the attitudes and possible changes in developer workload and behaviour, a qualitative method was chosen to collect data from the developers. This choice was supported by earlier research where it has been stated that qualitative methods are good tools for human-centered research (Taylor, Bogdan, & DeVault, 2015). As the developer team working with database views is rather small and consists of only four people, face-to-face semi-structured interviews were conducted. Interviewing the developers directly made it possible to let them openly express their current feelings and experiences before and after the automation.

Having group interview was considered, as it has many positive effects on the quality of collected data (Hirsjärvi, Remes, & Sajavaara, 2009). However, in addition to the positive effects of group interviews, there are some downsides which were considered. As group interview is conducted simultaneously to multiple people, it is possible that there might be one “dominant voice”, which sets the pace and tone for the answers. It is also possible that some negative aspects are not discussed, out of the fear that someone might feel offended or embarrassed. Eventually the idea of group interview was dropped, because of the many possible threats to the validity of the interview results. (Hirsjärvi et al., 2009.)

Due to the COVID-19 pandemic, the interviews were conducted remotely in Microsoft Teams. Interviewees were first approached by email, asking about their interest towards participating in voluntary research interviews. This email included information about the research and the interviewees were also provided with a possibility to provide answers in written form, if they couldn’t attend a face-to-face interview. All four developers agreed to be interviewed face-to-face and the schedule for interviews was agreed according to their own timetables. The interviews were conducted during two days. First two interviews were conducted 15.5.2020 and the last two interviews were conducted 18.5.2020. An overview of the interviews is presented in Table 4.

Table 4. Overview of the SW developer interviews.

Interviewee	Interview platform	Interview duration	Interview date
Interviewee 1 (I#1)	Microsoft Teams	30 min.	15.5.2020
Interviewee 2 (I#2)	Microsoft Teams	25 min.	15.5.2020
Interviewee 3 (I#3)	Skype	25 min.	18.5.2020
Interviewee 4 (I#4)	Microsoft Teams	20 min.	18.5.2020

As presented in Table 4, the duration of the interviews varied between 20 to 30 minutes. The duration times are rounded to the closest five minute mark. Some interview questions (Appendix A) were written down beforehand, but in semi-structured interview the interviewer was also able to ask additional questions that might yield more information. This, in addition to the discussing nature of the interviews, caused some difference in interview durations. Also one interviewee used Skype as the interview platform because of technical issues with Microsoft Teams. All of the interviews were conducted in Finnish, voice recorded and transcribed, with English translations. Permission for voice recording was asked from each interviewee in the beginning of the interview. Interviewees have not validated or proofread the transcriptions, but the transcriptions were written directly from the audio recordings and translated to English with the aim to preserve all of the original information and meanings.

The interview transcriptions were analysed using inductive qualitative content analysis. Thematic analysis was also considered, as it shares a lot of similarities with content analysis, including the division of text into smaller units and analysing these units after they have been recognized (Sparkes, 2005). Both content analysis and thematic analysis can be either inductive or deductive, meaning that they can be used to either find new themes from the data or looking for occurrences for existing themes, presented in previous research (Vaismoradi, Turunen, & Bondas, 2013).

As software maintenance automation is studied a lot less than, for example, test automation or deployment automation, and there are not many field researches on its effects on software developers, an inductive approach was chosen. This decision was based on Hsieh and Shannon's (2005) article, where they claimed that inductive analysis can be used in cases, where previous research does not exist or is limitedly available. Instead of reflecting the interview transcriptions on existing themes, the aim of the interview analysis is to describe any changes that the software developers have experienced in their work. This means that the interview answers were approached without any existing theme sets.

Because of the low amount of interviewees, quantitative analysis is not used. The focus is on individuals and their personal experiences. However, whenever any similarities between the experiences were found, the similarities were described and interpreted in the findings.

4.2 Questionnaire for the customers

In order to address the supporting research question RQ2 (“How do the customers experience the changes that took place after automation?”), personnel from the customer organizations were approached with an online Webropol questionnaire. The questionnaire was planned and conducted in cooperation with two other researchers, Uhlgren (2020) and Värttö (2020), who were also doing research for the case organization. The questions of each researcher were carefully separated and the overall length and form of the questionnaire was discussed in internal meetings with the organization representatives. Background questions 1 and 2 were common questions used by all three researchers.

The customer questionnaire (Appendix B) included both quantitative and qualitative questions regarding the new automated solution. Following the research ethics, a consent form was also included in the questionnaire. The questionnaire was piloted internally by the head of R&D department, head of customer support and service specialists. This piloting was done to make sure the questionnaire was appropriate and understandable.

The questionnaire was sent to customer organization personnel who were known to have used the reporting database. In total the questionnaire link and the message showcasing the purpose of the research was sent to 23 possible respondents. They were also encouraged to share the link with other employees within their organization, if they knew someone who was familiar with the reporting database. To increase the credibility of the message, it was sent by the product owner of the IS’s reporting module. The questionnaire was set to be open for one week. The receivers were also approached with two additional reminder messages during the one week, as the amount of respondents was alarmingly low.

Even with reminders and permission to forward the questionnaire, the amount of respondents was perceived to be very low. Eventually the final amount of respondents was only six, which means that instead of collecting quantitative data, the responses need to be analysed in a more descriptive, qualitative way. The reasons for low answer rate were discussed and there are many potential reasons for the low number of respondents. Combining questions from three separate studies made the questionnaire a little longer than was expected or even estimated. As the original time estimation for answering the questionnaire was somewhere between 15 to 20 minutes, the actual answer times were perceived to be a bit longer, varying from 16 to 66 minutes. It is also possible that even though the trustworthiness and legitimacy of the questionnaire were considered and content form explaining the research was provided, the customers did not perceive it legitimate enough to answer. In the organizational environment especially the content of information systems are often perceived to be under very strict confidentiality and for this reason some end users may have not felt comfortable answering the questions.

4.3 Maintenance task complexity analysis using MMG

To gain a more accurate insight on the maintenance complexity of old database view solution, a modified MMG (Maintenance Model Graph) analysis was conducted. The modification made compared to the original MMG analysis (Marques-Neto et al., 2013) was that the clustering phase was left out of the analysis. In this study the clustering was not used, because the analysed tickets were classified to concern the same task done multiple times. Varying ticket complexity was not in the scope of this study, but the focus was on the overall complexity of the same repetitive task. The complexity is

measured by the time the ticket stays active in the ticketing system, measuring the different states between creation and resolution.

The case organization utilises ITIL in their processes and all maintenance requests must be written down in form of a ticket. The case organization uses a ticketing system that contains tickets for all different departments within the organization, including for example customer support, software development, product management and customer projects. As the maintenance work was done by software development, the scope was limited to concern only tickets created for development team. Even though some tickets have come through customer support, these support tickets are not calculated towards the maintenance complexity. The help of customer support personnel was used when collecting the tickets for analysis. Customer support team was approached via email message where they were asked to list possible tickets related to database views if they recalled handling such tickets.

The tickets were collected from the case organization's ticketing system between 15. – 18.5.2020. They were collected using the search query tool of the ticketing system and by limiting the search results using the following predefined limitations.

1. The ticket had to be assigned to SW developers from the reporting team
2. The ticket concerned the old reporting database solution
3. The ticket required the update process of missing DB views
4. The ticket was created between January 2019 and January 2020
5. It was possible to calculate the complete time used on the ticket

With these preconditions, 14 tickets were found. The reason for low amount is that most of the tickets created during spring 2020 have been left waiting for the deployment of the new solution and are not eligible to be analysed. In general, tickets were left out because they were not completed yet, they were not included in the one year timespan set for the tickets or they did not directly concern the old reporting DB solution. The timespan for ticket creation was set to be from January 2019 to January 2020, because the ticketing system was implemented in the case organization during fall 2018 and because the information system itself is so new that the older tickets might have been too difficult to find. Tickets created after January 2020 are not analysed, because most of them had been solved by installing the new reporting database solution or by setting them to wait for the deployment of the new solution.

The analysis of these tickets can be found from Chapter 5, under section 5.3.

5. Findings

This chapter reports the findings acquired by all three research methods used. The findings include the qualitative analysis of the experiences reported by SW developers during their interviews and experiences reported by end users via Webropol questionnaire. Findings also include MMG analysis on maintenance request tickets regarding database view updates. This quantitative analysis was used to gather more concrete information about the complexity of the old solution and to support the qualitative questionnaire and the interview findings.

5.1 Software developers and their experiences with the automation

The interviews conducted with software developers aimed to address the supporting research question RQ1: "How do the software developers experience the changes that took place after automation?" To evaluate their personal experiences, 15 interview questions (Appendix A) were formulated, partly based on previous literature. In earlier automation studies it was noted that the automation of IT operations might affect workload and quality of service (Brown & Keller, 2006; Herrick & Tyndall, 2013; Krishnan & Ravindran, 2017) and also the maintainability of the software (Brown & Hellerstein, 2005; ElMaraghy, ElMaraghy, Tomiyama, & Monostori, 2012). These factors were taken into account in this interview. Also the possibility of new findings, focused especially on software maintenance task automation was considered when designing the interview questions.

Questions 1-6 surveyed the interviewee's experience with both new and old processes to find out their familiarity with the two different solutions. Q1-Q6 also made it possible to do some comparing between old and new solutions, as three similar questions were asked about both the new and old solution. Questions 7, 12, 14 and 15 addressed the possible changes in workload after automation and also the hidden complexity of the automation. Questions 8, 9 and 13 measured the overall attitude towards automation and the experienced reasoning for implementing automation. Questions 10 and 11 focused on software maintenance and the developers' attitudes towards maintenance tasks.

5.1.1 Background: Questions 1-6

All four interviewees were familiar with the old manual solution. I#2 and I#4 described the frequency of the manual database view updates to have been about once a month to multiple different environments, whereas I#1 and I#3 found it difficult to say how often the task occurred. I#3 had done the task manually under 5 times.

"*In the beginning I did some manual changes. Later when the (change request) tickets came, they were set to wait for this new solution. So lately I haven't done them at all.*" (I#3)

"*It's difficult to say how often (I did the changes). Sometimes there were multiple requests at once and sometimes it was a little quieter on that front. They occurred repeatedly. On the later stages they were coming after version updates, when the customers were informed about, for example, a new column in the database and they*

started asking whether it existed in their database views because they couldn't see it. It's not only during version updates. At one point, during the initial deployments of production environments, when they started doing the reporting and they noticed sometimes they couldn't find certain information. So yeah, there has been… it's not only during version updates." (I#1)

Overall the task of adding database views manually was described to be easy. The degree of difficulty with the manual updates was mostly experienced to be related to different system versions and the multiple manual phases during the update process. The developers also faced some difficulties with the old naming policy of the database views and tables. Especially I#1 had experienced difficulties with table names in different database schemas and adding these tables to the database views, as the tables got easily mixed up. However, as their experience with the database structure grew, the task became easier. In the old database view solution, there were three different language versions of the views. During the interviews it also was found out that the English and Swedish versions were not supported language-wise, but the columns were added to the views "as is", in Finnish.

"*In a way the changes are easy to do, you just write the changes down and… well, at some point there were moments when I was wondering how some views were completely missing. But then I noticed that in different schemas there have been tables with the exactly same names and the view has emerged in the same reporting schema.*

Oh and I have to mention that there was that… there were these three language versions to maintain. So that's… in principle... you should have been able to get the name for the English and Swedish versions as well, and that was… at some point you wouldn't even TRY to add the translations in Swedish or something like that. You just had to accept that they are named the same as the original view… So the column was there but the translation was missing." (I#1)

Three out of the four interviewees also mentioned the multiple phases to be time-consuming. This was explained by the test runs done against different databases to make sure the new views worked and setting the new updates to multiple environments and versions. Also some frustration was experienced because of the time used in a task that was seen to be rather small.

"*Changing the view, doing only that, that was easy. To add some field into it so practically it is only to run again the - view. Put the field there. But then it had all of the other work phases that were time-consuming. First is running the, the test runs. That thing firstly takes approximately 15 minutes to go through and then the merging work in different versions. And then – you have to notice if the database structure has changed between versions, so then it is… TWO different versions of that view, which suits the main version and that suits the version it has to be merged to. It takes a lot of time to – to run the test runs*." (I#2)

"*Well, all in all it was very routine, but it had so many phases and it took time. The change itself was so small and it felt like… a small change takes that much time. A lot of time. The most challenging was to make sure to remember to do all the phases in the right way."* (I#3)

All of the interviewees were also familiar with the new automated solution. When asked about their experiences considering the degree of difficulty, the answers differed quite much. Overall the developers experienced that after setting the automation to version control, they would no longer have to do manual changes. Modifications to the script, which is also a maintenance task, would be less frequent than the original manual

process. However, each of the interviewees experienced that they will need more time and instructions to maintain the script itself. I#1 mentioned that if they had to do changes to the automation script itself, it would take some time as they are not yet familiar with the script itself. I#1 and I#4 mentioned that they have not read the instruction manual or readme-file attached to the automated solution, but they have familiarized themselves with the new solution by meetings and face-to-face training sessions. I#3 mentioned they had read the how-to documentation, but it had been out of date and the script did no longer work in the same way it was described in the documentation. After noticing this, I#3 had personally sought guidance from the people who had developed the automation.

The feedback gotten from customers was quite similar for all of the interviewees. The overview of customer feedback received for both the old and new solutions can be found in Table 5.

Table 5. Overview of the customer feedback.

Interviewee	Old solution	New solution
I#1	Mostly questions about missing views or requests for assistance in finding certain information	Queries are faster than before
I#2	Queries were slow, update requests for missing DB views were common	Queries are faster, documentation and especially ER-diagrams make it easier to understand and to create queries.
I#3	Personally has not gotten feedback. Assumes by the earlier tickets that a lot of information was missing in the old solution.	Has not seen or heard any feedback.
I#4	Feedback only through tickets, assumes that customers were not happy because of the missing views	According to what has heard, customers have mostly liked it, especially the database documentation.

As can be seen from Table 5, the feedback received from customers has been relatively similar for all of the interviewees. I#3 and I#4 have mostly interpreted the feedback based on the amount and contents of change request tickets, whereas I#1 and I#2 seemed to have more insight on wider amount of feedback. The themes around the feedback can be roughly divided into four categories: (1) up-to-date DB views, (2) query speeds, (3) documentation quality and (4) ease of use.

5.1.2 Workload: Questions 7, 12, 14 and 15

Question 7 was a direct question regarding the detected workload changes. Two interviewees, I#1 and I#2, experienced that the automation has directly decreased their workload. However, I#1 also pointed out that in addition to this automation, their developer team has also grown and some of their workload has decreased also as the result of a larger team. I#2 also brought up the automation implementation workload, as automating a process and implementing the new solution to production creates extra work.

"*Decreased, yes, but of course there has happened, like, in the beginning when I was doing, I was doing those… all of that, like, alone, but nowadays in the team there are lot of people doing those things, so in that way the situation has changed, so one can't directly compare it.*" (I#1)

"*It has decreased the workload. So there is no more that attached maintenance work and now in the beginning of course it has raised the workload some when they are installed for the first time. Establishing the schemas there and then add usage rights into mint but it is a one time work then.*" (I#2)

I#3, who had not done as many manual updates as the others, found the new solution to indirectly affect their workload. As the automation freed some time from the other developers, I#3 feels that their workload is decreased as a result. Also I#4 brought up the shared time resources, even though they felt like the new solution has not affected their workload. The automation script is not yet in the version control and thus it is not completely autonomous, but it has to be triggered by a human operator. I#4 explained that this is the reason why the effects on their workload are not yet visible.

"*It hasn't impacted much, compared to the old one, because I didn't have to do it so much, but of course indirectly it has affected, because the team doesn't have to do those changes anymore.*" (I#3)

"Well if you think about that I have not much done these prior to this… it… it might have not decreased as is, but… well if you think about it from the point of view of the whole team and those who have done them earlier it must be a lot easier now than before… Because it takes a lot less time now when you just go and run the script and then… I don't know – the situation – it is not in version control yet? But – then when it is – then, especially. It eases significantly." (I#4)

Question 12 was about the noticed changes in software maintainability after the implementation of the automation script. As the automation is not yet implemented for all the customers, this question was seen a bit difficult, as the current script version is not autonomous. I#4 mentioned this to be a problem especially during the project phase of implementing the IS, because the database is cleared and rebuilt many times.

"During the project phases when the production environment is not ready yet, and they make constant changes in the database so you will sometimes have to go and run the script again, but, well, otherwise it has increased the maintainability." (I#4)

The overall software maintainability was said to have increased by all of the developers, because the whole reporting database is now handled automatically within the same script. The interviewees felt that reducing the need to make constant small changes manually had made the system more maintainable. When asked especially about the maintainability of the automation script itself, the interviewees felt like it is too early to say. Currently the developers haven't had the need to modify the script, but this might affect the maintainability in the future.

Question 14 inquired the availability and quality of the documentation related to the automation script. As Limoncelli (2018) described, all automated processes should be documented accordingly to decrease the hidden complexity or other underlying problems of automation. Afterwards it was noted that this question was formulated a bit unclearly, as many interviewees thought the documentation pointed to the database documentation that is provided to the customers together with the automated solution. As the interview was semi-structured, the interviewer corrected the interviewees and steered the discussion towards the instruction manuals related to the script. However,

even though the question was misunderstood in the beginning, this yielded some additional information about the benefits of the automated solution. The new solution, in addition to automating the database views, includes database structure documentation in the form of ER-diagrams and metadata description. This was brought up by multiple interviewees as a positive thing.

"*Yeah I feel that there is. Enough documentation. And if I myself work with database views, or tables, the most important documentation for me is the one that tells the relations between tables. And that's well taken care of in that (automated solution).*" (I#1)

"*I think that our documentation is in a very good level. Like – now it is… sufficient. And taken a bit further already, customers can benefit from it a lot. Then we can also utilise it in-house.*" (I#2)

After asking specifically about the instruction manual of the automation script, I#1 and I#4 had not read it at all. When further asked how they were able to familiarize themselves with the new solution, both answered that face-to-face meetings and training sessions have been helpful. After these sessions they had tested the script by themselves and when they were unsure about something, they had asked for help. So, instead of reading the readme.txt file, two of the developers had only participated in training sessions and then tested the script by themselves.

"*I have just run it (the script) by myself and asked questions when I have had some questions. That way I have gained some general information about it. And in the beginning we had these meetings where we went through the new solution. So (I have learnt to handle the script) pretty much based on it.*" (I#4)

Question 14 also brought up automation artefacts that were discussed also by Brown and Keller (2006). This means that automating one task often creates so-called automation assets that can be utilised in the future automation. I#2 pointed out, that a good documentation makes it possible for the organization to build future automation on top of the existing scripts.

"*There are strict guidelines on the script itself on how it works. It is intended to broaden the script in the future. Good it is documented so it is easier to broaden it.*" (I#2)

I#3 would hope for a more accurate documentation for the maintenance of the automation script, as the current readme.txt was said to be outdated. On the other hand, I#4 pointed out that instead of a large manual, there should be a short introduction available. It should be compact enough for the developer to be able to take a look at it and quickly finding answers.

Question 15 was asked to find out, whether the solution has affected the incoming change requests. Because the new automation is not yet implemented in all of the customer environments, there are still organizations using the old solution. All four interviewees pointed out that for those organizations who have had the new solution installed, the view update requests have completely stopped.

I#1: *Yes they have decreased for those customers who have the new solution. It feels like, the ones that I receive right now, I always start thinking whether it has been discussed with them, that they would receive, or we would install the new views for them. So quite clearly, yes, yes there is change.*

Interviewer: *So you said that you start thinking whether the new solution has been discussed, do you mean the tickets are about the old solution?*

I#1: *Yes, yes, precisely, so I haven't seen tickets coming about the new one. But the old ones are still coming.*

In addition to these experiences by I#1, the other interviewees shared very similar experiences. However, I#2 pointed out that customers who have had the new solution installed in their environment are often asking more detailed information about it and also requesting user rights. These requests are being reduced by arranging training sessions for the customers and providing them the information they need.

"*Well now that we are installing the new solution, there is certain uncertainty in the customer side about what it means. So, that kind of general questions have come but nothing like, like we should add columns or something. So those have stopped completely with those customers who have already gotten the new solution. And of course we are trying to reduce the tickets even more in such a way, that we provide the customers with short training sessions about their new reporting database and its documentations, so we can block some of the tickets and reduce the amount of incoming questions. And at the same time we enhance the customer satisfaction when we train them. And of course some questions related to some functions and their use, like if they get error message during it, so it just means, when they get it, that there hasn't been rights given to it that can access to the data. So that kind of things have come, but I have understood that those occur only now when something new is being implemented. So those will be left out as well when they are handled.* " (I#2)

In general the workload was experienced to have reduced either directly or indirectly by reducing the overall workload of the team. The new solution has evoked some new questions from the customers and for this reason new tickets related to the database views are still coming. However, the repetitive manual task of view updates has been completely removed regarding the environments where the new solution is in use. Software maintainability was seen to have enhanced after the reduction of manual tasks, but it was still too early for the developers to estimate the maintainability of the new automation script. Two of the four developers had not read the instruction manual for the automation script, but they had been able to learn about it in training sessions and by testing it themselves.

5.1.3 Automation: Questions 8, 9 and 13

Question 8 was asked to find out, what the interviewees felt like was the reason for implementing the automation in the first place. The answers from all interviewees were very similar. All of the respondents pointed to "manual work" and two mentioned the task to be "arduous". I#1 described the manual task to be "completely pointless" when it was made by hand in multiple environments repetitively. One factor that lead to automation was also to save some time for more important tasks.

"*So, from my point of view and, and the reason why I was driving this thing forward too, was that we had to get rid of the manual work, so it wouldn't be like that all the time… it is COMPLETELY pointless task. If you get like, once or twice a week that kind of a request, or, investigation or something else, about why is something missing and you investigate it and fix it and stuff, so it was purely because of that.*" (I#1)

"Well supposedly one had to be constantly updating them like… manually. And that took… those tasks can take a lot of time. So if you can do it with automation – and save time, well - of course it is better." (I#4)

Question 9 continued on the same theme, asking the interviewees about their own attitudes towards automation in general. A positive attitude towards automation is, according to Limoncelli (2018), a vital part of automating any process. All of the interviewees had a positive attitude towards automation and they would increase the amount of automation whenever possible. I#3 pointed out that by automating repetitive manual tasks, it would be possible to focus on the more important tasks. When additionally asked, what they considered to be the important tasks, the answer was "developing new things". Also I#2 brought up many things in their answer, considering even that automation provides an interesting and meaningful task for the person planning and developing the automation.

"Well I think that most certainly increase (the amount of automation). So the manual work – with repetitive things, it is not meaningful from the point of view of the doer. More preferably design some kind of solution that takes away the manual work phases. That also challenges the doer to do and, and it is meaningful because it takes the stale work phases away. Like, think a little further and automate it. And most certainly all kinds of automation should be increased." (I#2)

It is known that automation process takes time and may be arduous with all the planning, testing, iterating and deploying (Bartusevics, 2017; Limoncelli, 2018). With question 13 the intention was to find out, how much the costs of automation weigh when compared to the outcome. Three of the four respondents used the phrasing that automation will "pay itself back in the long run". I#1 explained that they trust in reason, arguing that usually when something is decided to be automated, there most probably is a good reason for it. I#3 and I#4 took the new solution as an example, I#3 mentioning the hopes to increase customer satisfaction and I#4 mentioning the removed time-consuming manual work.

5.1.4 Software maintenance: Questions 10 and 11

Question 10 was based on Mookerjee's (2005) research, where it was stated that maintenance tasks are often experienced as arduous, repetitive or boring by the software developers. The answers to this question were not as strict, as the developers described the continuous maintenance tasks to be either "routine work" (I#1, I#2) or "a bit burdensome" (I#4). I#3 stated that "sometimes they are nice to do", but added that whenever there is a task that "pops up again and again", they start to wonder if there would be some way to make the said task easier. I#1 had the most critical stance on maintenance, as they described some maintenance tasks to be pointless or boring, but at the same time they described it to be a routine work, part of the software developer's daily work. I#4 pointed out the possible delays caused by maintenance tasks, as the development of new features may get disrupted whenever a constant SW maintenance tasks occur.

"If there are lots of them then – at times a bit heavy or burdensome. When you have other tasks in progress and when – when they come from the side, especially from those production environments and they have to be done immediately and the other tasks are delayed. So well… They are a bit stale if you receive lots of them." (I#4)

With I#1, the possibility to further automate some maintenance tasks was discussed alongside Q10. I#1 expressed interest towards further automation as long as the customers are also informed about the new solutions.

"It is beneficial, if one acknowledges a possibility, of course it is profitable to automate it. Of course it's good to inform the customer, so they know how it works and that this kind of automation has now done something for the user, so it doesn't stay a secret. But otherwise, yes." (I#1)

Question 11 handled the two different types of SW maintenance, corrective and enhancing. I#1 and I#3 stated that they do more enhancing maintenance, which means that they experience their work to revolve more around perfecting and adapting the system to suit possible new environments or requirements. I#2 and I#4 experienced their maintenance tasks to be shared "somewhat evenly" across corrective and enhancing maintenance. I#2 also added that the IS is still rather new and because of this, there are some bugs that need to be fixed, but believed that the system maturity will take care of the current maintenance type ratio.

"It is possibly… half and half. So both exist. But more preferably one would do that - software enhancement more. But then there is that... the product is rather new so... there are bugs – bugs have to be fixed. But when it matures up it will likely turn around the ratio number. Like there will be less bug fixing later. Because they have already been fixed." (I#2)

5.1.5 Summary

All of the developers had done manual changes to the database views before the automated solution. I#3 had done them the least, only about three times. Lately the manual changes had been left undone, as the developers were waiting for the new solution to be set for the customers who are still using the old solution. The old process was described to be rather easy. The difficulties experienced with the old solution were mostly about version branches and the multiple manual phases during the process. Only one developer mentioned the three language versions that were left with very little attention, as the column or table translations were eventually not done at all. This is, even if not numerically significant, important information about the situation with the old manual solution. Three of the four interviewees told the old solution to be time-consuming. This was also generally mentioned to be the ultimate reason for automation in the first place, as it saves time in the long run. Increased customer satisfaction was also mentioned by two interviewees, as there were no longer missing columns or tables noticed after implementing the new solution.

The overall effects of automation on SW developers are described in Table 6.

Table 6. Overall effects of automation.

	Positive effects	**Negative effects**	**Neutral effects**
Degree of difficulty	Decreased difficulty of maintaining database views Increased amount of DB documentation, which makes it easier to work with databases in general	Maintainability of the automation script is still unknown for the developers Automation script has outdated instructions manual, which has caused some issues when familiarizing oneself with it	-
Overall workload	All developers experienced that their workload had decreased, either directly or indirectly	Possible script modification may take time in the future	New solution has only one language selection (English) so the different language versions do not need to be updated
Incoming change requests	Change requests related to missing data are no longer received Reduced correcting maintenance type	New requests are coming as the customers inquire about the new solution or request user rights	Automated solution is not yet installed in all customer environments and maintenance requests related to the old solution are still ongoing
Future development	The new script can be expanded in the future development	-	-

Using content analysis and by observing the most discussed topics by the developers, four main effect areas were recognized: (1) degree of difficulty, (2) overall workload, (3) incoming change requests and (4) future development. Overall the developers experienced that the automation had affected on the difficulty of maintaining the reporting database, the overall workload, the amount of change requests and the future development plans. The most positive change was removing the need for manual updates that were often experienced to take too long. Also the phases included in the process were said to be complex and merging the changes to multiple versions was experienced to be either risky or time-consuming. New database documentation that has been implemented with the automation was also mentioned to help the developers in database related work. The effects of database documentation to work tasks in general in the case company are analysed and reported in more detail by Värttö (2020).

The experienced changes in workload varied. While two developers had experienced direct decrease in their workload, two described the decrease to be more indirect. However, it was also pointed out that as the script becomes autonomous, so that it does not require human operator to trigger it, the workload changes will be more visible for the whole team. The outdated translations are no longer an issue, because the new

solution only has database views in English. As the effects of removing Finnish and Swedish language versions from the database views is still unknown, this effect was marked as neutral. The localisation and translation related factors are more closely analysed by Uhlgren (2020).

Change requests related to the new solution were noticed to be different from the old ones. Before the automation, missing DB tables and columns had to be manually added to the DB views, but the new solution has eliminated the need for this task completely. The tickets related to new solution are mostly inquiries about some elements of the new solution or requests for user rights. I#2 mentioned that these are natural when implementing a new solution to customer environments and should be decreasing after the modifications are done.

Automating the database view updates and DB documentation has provided the case organization with assets that can be used in the future automation projects as well. I#2 revealed that there are plans to do further changes in the reporting database and the new automation script can be used as a basis for the upcoming changes.

5.2 Effects of automated solution on end users

This section showcases the perceived effects on end users. These results are based on the questionnaire presented in section 4.2.

5.2.1 Background information

The collected answers were analysed using descriptive methods, as the amount of respondents was too low for quantitative analysis. The six respondents came from slightly different backgrounds. Some of them had used the old reporting database solution provided by the case organization, but some of them are new customers who are still in the project phase of IS deployment.

Table 7 presents the general information about the questionnaire respondents and their familiarity with the reporting database solutions.

Table 7. Overview of questionnaire respondents.

Respondent abbreviation	Experience on the substance field (years)	Experience with databases (years)	Familiar with the old solution	Familiar with the new solution
R#1	3	13	No	Yes
R#2	6	2-3	Yes	Yes
R#3	12	20	No	Yes
R#4	5	3	Yes	Yes
R#5	5	30	Yes	No
R#6	12	20	No	Yes

As presented in Table 7, half of the respondents had had the old reporting DB solution in use. Only two respondents were familiar with both solutions and one respondent (R#5) was not familiar with the new solution at all. Therefore by examining the answers of R#5 it was not possible to directly address RQ2: “How do the customers experience the changes that took place after automation?” However, by also taking into account their experiences with the old solution, it was possible to compare it to the experiences of the other end users.

A numerical Likert scale was partly used in the questionnaire and it was originally planned for small scale statistical analysis. However, as the number of respondents was very low, the numerical results will be analysed qualitatively, by searching for patterns in the answers. This decreases the chance of numerical bias and provides the analysis with more value from the answers.

Before discussing the effects of the new solution, it was important to know how the end users utilise the reporting database. This revealed the importance of functional reporting database. In Table 8, the usage of DB views and DB documentation are described by respondents.

Table 8. Overview of DB view and documentation usage by end users.

Respondent	Usage of DB views	Usage of documentation
R#1	IS still in project phase, but based on earlier IS, believes that the view usage will be daily	Each time when doing queries to the DB
R#2	During project phase they are used on migration testing, in the future used for reporting and forming integration material	Twice a month
R#3	Used daily, not declared how	Used a lot in the beginning, less after familiarizing with the DB
R#4	Daily and a lot. Many business critical implementations are built on top of the DB views and for this reason it is important that the DB views are well supported and maintained.	No documentation available yet.
R#5	DB views utilised widely for operative, statistical, analytical and KPI reporting. Also ad hoc queries with specified datasets.	Extremely rarely, as the documentation is not up-to-date and is missing important information. If something is missing, a ticket is created for case organization.
R#6	Utilised as a part of their own automation, integrations use the data from DB views.	More in the beginning, less after familiarizing with the DB.

Most of the respondents said that the DB views are used a lot, for multiple purposes. Only R#1 was unsure of the frequency of DB view usage, but according to the usage of

their earlier IS, they expected use of DB views to be daily. As the customers utilise the views a lot, it is important to measure how the new solution has affected the end users and their work. In addition to ease the work of the SW developers, the automated solution was developed to provide better quality for the customers.

The use of DB documentation varies between end users. R#1 described using it each time when they are handling the reporting DB, whereas R#3 and R#6 have used documentation more in the beginning. R#5, with the old solution, described the old documentation to be inaccurate and for this reason it is unusable for them. Supporting the statements of SW developers, R#5 solves problem situations by creating a ticket for the case organization.

5.2.2 Comparison of old and new solution

To compare the old and new solution and see the impacts of automation, the questionnaire contained a section where similar claims were made concerning both the old and new solutions. To these claims, the respondents were able to answer on a Likert scale 1-5, where 1 means “Completely disagree” and 5 means “Completely Agree”. Question 5 contained claims a-h about the old solution and Question 6 contained claims a-h about the new, automated solution.

Question 5 was about the old solution and it was answered by three respondents, R#2, R#4 and R#5. In Table 9 are collected the answers given by customers, who have used the old reporting database. It should be noticed that respondent #5 has not yet used the new solution, because it is not yet implemented in their production environment.

Table 9. End user experiences with the old solution.

Claim	**R#2**	**R#4**	**R#5**
a) Content of database views is up-to-date	Somewhat disagree	Somewhat agree	Somewhat agree
b) Database views are named accordingly	Somewhat agree	Somewhat agree	Somewhat agree
c) I would need more training to use the database	Somewhat disagree	Completely disagree	Completely disagree
d) It is difficult to utilise the database views	Somewhat disagree	Somewhat disagree	Completely agree
e) The queries made to the database views are fast	Somewhat disagree	Completely disagree	Completely disagree
f) When I do a change request regarding the database views, it is handled quickly	Somewhat disagree	Completely disagree	Neither agree or disagree
g) Content of database documentation is up-to-date	Completely disagree	Completely disagree	Completely disagree
h) Database documentation is easy to understand	Somewhat agree	Neither agree or disagree	Somewhat disagree

From Table 9 it can be seen that the respondents have similar answers regarding the view naming policy (b), need for additional training (c), query speed (e) and database documentation (g). Overall the DB views were experienced to be named clearly and understandably. The respondents do not feel a need for additional training, but of course in the cases of R#2 and R#4 this may also be due to the fact that they are no longer using the old reporting database. Query speed in the old reporting DB was experienced to be slow, especially by R#4 and R#5 who are using the reporting database daily and for business critical tasks, as reported previously in Table 8. Database documentation was clearly not up-to-date, as all respondents have collectively completely disagreed with claim g. This issue was also discussed by Värttö (2020), who stated that the old database documentation was outdated, with inaccurate information and missing metadata.

Five respondents answered to the claims presented in Question 6, concerning the new reporting database solution that has been installed to their environment with the new automation. Two of these respondents, R#2 and R#4, have also used the old solution. Table 10 contains an overview of the end user experiences with the new solution.

Table 10. End user experiences with the new solution.

Claim	R#1	R#2	R#3	R#4	R#6
a) Content of database views is up-to-date	Somewhat agree	Somewhat agree	Somewhat agree	Completely agree	Somewhat disagree
b) Database views are named accordingly	Somewhat agree	Somewhat agree	Somewhat agree	Completely agree	Somewhat agree
c) I would need more training to use the database	Completely agree	Somewhat disagree	Neither agree or disagree	Completely disagree	Somewhat disagree
d) It is difficult to utilise the database views	Somewhat agree	Somewhat disagree	Somewhat disagree	Completely disagree	Somewhat disagree
e) The queries made to the database views are fast	Somewhat agree	Neither agree or disagree	Neither agree or disagree	Completely disagree	Neither agree or disagree
f) When I do a change request regarding the database views, it is handled quickly	Neither agree or disagree	Neither agree or disagree	Completely disagree	Completely disagree	Somewhat disagree
g) Content of database documentation is up-to-date	Somewhat agree	Somewhat agree	Somewhat disagree	Completely agree	Somewhat disagree
h) Database documentation is easy to understand	Somewhat agree	Somewhat agree	Somewhat disagree	Neither agree or disagree	Somewhat agree

Quite similarly to the answers regarding the old solution in Table 9, the experiences between the respondents were similar concerning the DB view names (b), but the other answers were more divided. Concerning claims c and d, the ease of use and need for training, R#1 was the only one considering it difficult to utilise the database views and they are also the only respondent experiencing the need for additional training. The four other respondents experienced the new reporting database easy to use and either disagreed or did not respond (R#3) to the need of training (claim c).

Quite interestingly the claims e and f are slightly controversial compared to the results of the interviews with SW developers. The feedback that the developers had received

had been especially positive about the query speeds, but this does not clearly show in the questionnaire results. The answers regarding query speeds are slightly better than with old solution in Table 9, but the only one agreeing on the speedy queries was R#1, whereas others either could not say or disagreed with the claim. R#4 completely disagreed to this claim, similarly as they answered in Table 9 concerning the old solution.

Also the handling of change request tickets regarding database views are experienced by R#3, R#4 and R#6 to take a long time. This is also somewhat controversial to the interview with SW developers, who had experienced that the amount of change requests has vastly decreased for those customers who had had the new solution installed. The explanation for this might be that the new solution had created a need for new kind of tickets related to user rights and inquiries about the new reporting database structure and documentation. R#4, who stated that ticket handling takes a long time, also pointed out that they have not yet received the documentation that should be delivered together with the new reporting database solution. The answers from R#1 and R#2 can be interpreted to support the developers' experiences, as it is possible the answer "neither agree or disagree" may point out that neither of these users has created a ticket related to database views of the new solution.

Database documentation was considered to be up-to-date and easy to understand by three respondents. The database views were up-to-date according to all other respondents except R#6, who somewhat disagreed with claim a. The new automation was created especially to make the views to be automatically updated, so this answer should be looked into more carefully. It is possible, as discussed in the interview with I#4 (page 39), that during project phase these database views may get erased because of the ongoing changes and rebuilding of databases. R#6 stated that they are not familiar with the new solution (Table 7) and this might mean that their environment is still under construction and the situations described by I#4 have emerged.

To compare more accurately the two solutions, it was required to have a closer look at the customers who have experience on both solutions. Combinations of the two end users who have used both are collected in Table 11.

Table 11. Changes after installing the new solution.

	R#2		R#4	
Claim	**Old solution**	**New solution**	**Old solution**	**New solution**
a) Content of database views is up-to-date	Somewhat disagree	Somewhat agree	Somewhat agree	Completely agree
b) Database views are named accordingly	Somewhat agree	Somewhat agree	Somewhat agree	Completely agree
c) I would need more training to use the database	Somewhat disagree	Somewhat disagree	Completely disagree	Completely disagree
d) It is difficult to utilise the database views	Somewhat disagree	Somewhat disagree	Somewhat disagree	Completely disagree
e) The queries made to the database views are fast	Somewhat disagree	Neither agree or disagree	Completely disagree	Completely disagree
f) When I do a change request regarding the database views, it is handled quickly	Somewhat disagree	Neither agree or disagree	Completely disagree	Completely disagree
g) Content of database documentation is up-to-date	Completely disagree	Somewhat agree	Completely disagree	Completely agree
h) Database documentation is easy to understand	Somewhat agree	Somewhat agree	Neither agree or disagree	Neither agree or disagree

From Table 11 it can be interpreted that changes have been experienced especially in the content of database views and the content of documentation. In addition R#2 has noticed assumedly some improvement in query speeds and service speed or they have not written queries or created change requests. R#4 has noticed improvement in view naming policy and ease of using the database views. The new solution has not, according to these two respondents, affected the need for additional training or documentation comprehensibility. When interpreting Table 11, it must be noted that R#4 has not yet received the complete documentation for the new database solution and for this reason the answers regarding the new documentation may not be completely valid.

At the end of the questionnaire, there were also two open questions (Q7 and Q8) to which the respondents could write their thoughts about the new solution and leave their ideas for improvement. The answers were collected to Table 12, where the comments from each respondent are combined in the middle column and on the rightmost column are listed the solutions in use for the respondent.

Table 12. End user comments regarding the two solutions.

Respondent	Comments	Solution in use
R#1	If the full documentation would already be available, work would be a lot easier Needs additional documentation especially about the functions usable in reporting database	New solution
R#2	The new solution brings lot of new opportunities New solution feels a bit more arduous but the old one was clearly lacking a lot of information and it took time to receive the updated information	New solution. Has also used the old solution.
R#3	Earlier solution was built on top of a messy DB, but overall the reporting tasks were easy to do with it. The new solution is expected to be better after the full documentation is provided.	New solution
R#4	Welcomed the new solution as an improvement.	New solution. Has also used the old solution.
R#5	Was interested in the new solution and would like to hear more about it. Described the old solution to be lacking documentation, inconsistent in view naming policy and having a bad overall performance	Old solution
R#6	Had no earlier experience on environments provided by case organization Expects a standardized solution with all of the documentation and other content immediately available	New solution

Overall it seemed that the customers were happier with the new solution, but the lacking documentation has provided challenges for them. R#5, who was not yet familiar with the new solution, took interest in it and also gave details about the shortcomings of the old solution.

There were some inconsistencies found within the questionnaire answers. R#3 answered to Q5 (“Have you used the earlier solution or are you familiar with it?”) negatively, but in Q7 (“Please describe your experiences with the new solution and if possible, compare it with the old database solution”) they gave quite detailed opinion about the old

solution and also described the old solution to have been easy to use. This may have occurred due to some obscurity in the questionnaire, which has lead to a misunderstanding. It is also possible that the customer is in Q7 comparing the new solution to some other "old solution" that has been used in their organization.

5.3 Quantitative maintenance request analysis using MMG

MMG analysis was used to analyse 14 maintenance request tickets concerning the old solution. The collected tickets had been created between January 2019 and January 2020. Only fully completed tickets were included in the analysis, as it is impossible to calculate maintenance times of incomplete requests. It should also be noted that this analysis does not calculate the work time of developers or any man-hours, but the whole time of ticket being active in the system. This affects the customer satisfaction and can also yield information about the overall complexity of the task.

Usually MMG analysis includes a workflow graph of the tickets, but because of the NDA with the case organization, the actual ticket statuses had to be anonymized in this research. Instead the ticket WaitingTime, ServiceTime and QueueTime are announced for each ticket, without revealing the different ticket states. The measure used for the time was hours and the hours were counted only for the usual working hours, 8.00 – 16.00. Any time outside these hours was left out, as well as weekends.

Case organization has a standardized workflow for the tickets. From this workflow it can be analysed whether the ticket state has been waiting (for any action or response) or under any action, such as implementation, validation, deployment or planning. The workflow had to be hidden in this study, but the calculations were based on the ticket states of the standardized workflow in case organization.

The formula for calculating ServiceTime and QueueTime is the following (Marques-Neto et al., 2013):

```
QueueTime = WaitingTime + ServiceTime

ServiceTime = PlanningTime +

ImplementationTime +

ValidationTime +

DeploymentTime
```

We can find three different time variables for each ticket. WaitingTime is calculated by adding up each hour when the ticket has been waiting for something, for example waiting to be taken under progress or waiting to be tested. ServiceTime is calculated by adding up all other work phases, such as testing or development. QueueTime, which is the time the overall handling of the ticket has taken, can be then calculated by using these two variables.

For example Ticket 1, abbreviated T#1, had been created 02.01.2019 at 09:32. It had been taken under progress 03.01.2019 at 08:13. After this, the ticket had gone through phases where it had not been waiting for anything, but instead it had been either planned, implemented, validated or deployed. This means that the WaitingTime of T#1 had been approximately 7.5 hours. The minutes were rounded to the closest 30 minutes.

Initially T#1 was marked as completely done 10.01.2019 at 12.08 and had not gone through any waiting processes after 03.01.2019. This means that the ServiceTime of T#1 can be calculated by adding up the time it has been under different work phases. Taken into account that 5th and 6th of January 2019 were Saturday and Sunday, these two days are not calculated as workdays. The variable for ServiceTime provided by this calculation was approximately 44h. As QueueTime = WaitingTime + ServiceTime, it was possible to add up the two known variables to find out the final QueueTime of T#1.

QueueTime T#1 = 7.5h + 44h = 51.5h

All 14 tickets were analysed using the above method and after these calculations it was possible to create Table 13 with all calculated time variables.

Table 13. Calculated time variables for the maintenance requests.

Ticket	WaitingTime	ServiceTime	QueueTime
T#1	7.5h	44h	51.5h
T#2	60.5h	180h	240.5h
T#3	0.5h	2.5h	3h
T#4	30h	9h	39h
T#5	24h	8h	32h
T#6	11h	2h	13h
T#7	11h	9.5h	20.5h
T#8	43h	15.5h	58.5h
T#9	26h	27,5h	53.5h
T#10	399h	11h	410h
T#11	215.5h	80.5h	296h
T#12	7.5h	43.5h	51h
T#13	1.5h	8h	9.5h
T#14	1h	216h	217h
Total	838h	657h	1495h
Average	60h	47h	107h

Maintenance request tickets T#10 and T#11 were divergent from other tickets in terms of WaitingTime variable, as can be seen from Table 13. Differences in waiting time may occur, for example, when the ticket visits waiting type status multiple times, but there are also more humane reasons that may cause the outliers. It is possible that

vacations, such as summer holidays had occurred, something was being inquired from the customer or the ticket was waiting to be merged to the next available version.

Other outliers were T#3, T#13 and T#14, where the waiting time had been abnormally low. The reason for this may also vary, as the ticket may have been taken into progress very fast or been left in the wrong state. The latter could be suspected on the case of T#14, where the service time is also an outlier, being over 200 hours. It must be noted that the ticket status was handled manually by the person working on it and this may have caused some humane errors in ticket statuses.

Unfortunately, as the workflow was defined by the case organization to be confidential, the graphical depiction of different ticket statuses and their occurrences in different tickets was not available in this study. From Table 13 it can still be seen that the average time the ticket was active in the ticketing system has been 107 hours, which translates roughly to 13 work days when dividing by 8, which is the most usual length of a work day.

As MMG analysis is not used for calculating man-hours, the results do not yield any implications related to the developers' workload. Instead, from these results it is possible to see that the outdated database views in reporting database have caused work in the case organization and customers have had to wait for the resolution multiple days, on average more than two work weeks to get the updated views. If the new automated solution worked as it was designed, the tickets related to manual updates should have stopped and the time used on these maintenance tickets could be saved.

6. Discussion and Implications

The purpose of this study was to find and report the effects of automating a single repetitive software maintenance task. The research was conducted in a Finnish IT organization as a case study. Summary of the findings and their reflection to prior research is reported in this chapter. Each research question is responded explicitly in their own chapter, and the implications are summarized to address the main research question in section 6.3.

6.1 RQ1: Experiences of the SW developers

The automation effects on software developers were analysed using a qualitative interview and inductive qualitative content analysis. The overall findings were eventually divided under four themes, according to the interview discussions and the statements acquired from the software developers. The four themes were degree of difficulty, overall workload, incoming change requests and future development. Out of these four themes, the degree of difficulty was the most discussed and it was found to be affected the most by automation. Future development was discussed the least, but it held a significant value for the case organization and the developers. For this reason the automation effects on future development was listed as an independent theme.

Degree of difficulty

The automated solution had decreased the degree of difficulty of maintaining the database views. The old solution was mentioned to be generally easy to do, but some issues were brought up by the developers. Manual changes to version control and interpreting the database schemas was experienced to be occasionally difficult, especially if the human operator was not very familiar with the reporting database.

After automation, the developers no longer experienced the need to do manual updates to the databases and version control related to database view updates. Additionally, the database documentation provided with the new solution was experienced to be helpful also for the software developers. Doing tasks that are related to database were said to be easier when there is an existing database documentation available.

The maintainability of the new automation script was not yet known. The developers had familiarized themselves with the script by having face-to-face meetings and by asking questions directly from the people who had developed the script. Some uncertainty occurred when discussing the possible modifications to the new script. This is due to the outdated documentation that was attached to the automation script. As Limoncelli (2018) stated in their article, all automation done for IT operations should be documented accurately and the documentation should be updated consistently along with any changes. This is something that should be taken into account each time when changes are made to the automated solution.

The ironies of automation that were discussed by Bainbridge (1982) and Strauch (2017) should be studied in more depth as the automation gains more maturity. The necessity of human participation is still apparent in the case handled in this study, as automation always requires monitoring and error management. In the presented case, the

automation was not yet set to run autonomously and for this reason it required a human operator. After setting the automation to be autonomous, there should be additional training and updated instructions for error management and script monitoring.

Overall workload

The workload was experienced to have decreased either directly or indirectly. The developers who were doing the manual changes the most experienced their workload to have directly decreased after the automated solution. Other team members described their workload to have decreased indirectly, after the team's overall time usage on DB views had decreased after automation. However, the size of the developer team had increased lately, and for this reason the statements related to workload decrease may not be completely accurate.

The old solution was described to be extremely time-consuming with all the manual phases it required from the human operator. Also keeping up the different language versions of the reporting database was eventually left undone completely, which partly also confirms that the workload around the database views was too high.

The future modifications for the automation script were mentioned to increase the workload momentarily. The developers were unsure about how long time it will take for them to familiarize themselves with the script. The hidden complexity of automation that was discussed by Brown and Hellerstein (2005) and Tschiersch and Brandt (1996) may increase the workload, because the developers will need to learn new skills and understand the structure and the integrations that were built in the new automation.

The developers were also waiting for the new solution to be set running autonomously. This means that instead of requiring a human operator to trigger the script, it should have been set to run by itself (Limoncelli, 2018). As suggested by Limoncelli (2018), whenever an automation requires frequent human operator, it should be considered whether it would be easier to combine the automation with an existing system or scheduler in such a way that it would trigger autonomously when needed.

Incoming change requests

For those customers who had had the new solution installed, the view update requests had stopped completely. As the new solution was not yet implemented for all customers, the change requests related to the old manual solution had not stopped completely.

The customers with the new solution were still doing tickets, but the new requests were more perfective by their nature, instead of correcting. According to Lientz and Swanson (1980) and Hatton (2007) most of the maintenance work should be perfective or adaptive. Most of the tickets related to the new solution were either inquiries about how it works, or requests for new user rights. The amount of tickets was expected to decrease after the customers have been familiarized to the new solution and the user rights have been given to the end users.

Future development

The assets gained from the new solution can be reused in the future development work. Additional localization features were already built to expand the new solution, as is presented in the study by Uhlgren (2020). The case organization was also planning an additional automation that would utilise the new solution.

This is in line with the findings of the study by Brown and Keller (2006) who added automation to change management. Brown and Keller (2006) stated that after automating one process, it will be easier to add more automation. The time used for the future automation can be decreased by utilizing the existing automation assets, in this case the automation script, the new database structure and the new documentation.

6.2 RQ2: Experiences of the end users

The customers were approached with an online Webropol questionnaire where they were asked to describe their experiences with the reporting database. The amount of respondents was low and descriptive analysis was conducted for the answers. As some customers had only used either the new or the old solution and some had used both the old and the new ones, they were asked to answer to only the questions that were related to the solution they were familiar with.

Overall it was found out that the database views maintained manually were missing a lot of data and were often outdated, creating a need to create maintenance requests to the case organization. Also the queries made in the reporting database were experienced to be slow and the database documentation was either outdated or missing.

As the new automated solution aims to keep the views up-to-date, provides new documentation in the form of ER-diagrams and metadata overviews (Värttö, 2020) and includes new database structure, the expectation was that the customers would be satisfied with the new solution. The customers who had experience on the new solution were mostly pleased with the database views and the database documentation provided with the new solution. However, some customers felt that the documentation was still lacking some function descriptions and that the new database was a bit more arduous to use.

Despite erasing the need to update the database views one-by-one, no significant differences were found in the customers' experiences towards the speed of handling their change requests. The developers mentioned in their interviews that the customers with the new solution are creating tickets where they make inquiries or require new user rights for the reporting database. This may be the reason why the customers did not experience changes in the handling of their change requests.

6.3 RQ: Overall effects on the stakeholders

The automation had affected the software developers and end users. Most of the reported changes were reported by the software developers, who experienced changes related to the difficulty of the work task, overall workload, incoming change requests and future development. The changes experienced by the customers were fewer, but overall they experienced that the new solution has enhanced their reporting database. The automation was not yet deployed to all customers and for this reason the study regarding the customer experiences did not yield a lot of information.

The developers had a positive attitude towards automation and they experienced that additional automation of maintenance tasks decreases their overall workload. The maintenance tasks were experienced to be routine work, but in cases when a certain task needed to be done weekly or monthly it was experienced to be either pointless or boring. Mookerjee (2005) discussed this phenomenon in their article, stating that it is

common for software developers to feel bored or frustrated when working on a repetitive task frequently. For this reason automation should be considered to be added in cases when a task is reported to be repetitive and time-consuming.

When implementing automation, documenting the process and the end product is important (Limoncelli, 2018). In this case, the readme-files related to the automation script were outdated and thus did not yield enough information for the SW developers. The developers had gained their knowledge by participating in meetings and asking for training sessions, but they reported a need for a short written documentation that would be kept updated.

The importance of database views was reported to be very high for the customers, as they had many business critical features built to utilise the information contained in the reporting database. The MMG analysis revealed 14 tickets related to database view updates. The tickets had been created between January 2019 and January 2020 and the time these tickets had been in the ticketing system was 1495 hours, which translates to approximately 187 workdays. Of course this is not the same as the man-hours used on the tickets, but it yielded some information about how long the customers had to wait for the updated database views. It should be also taken into account that some update requests were not reported as a tickets, but the customers had contacted the developers directly via email. These emails were not included in the MMG analysis and for this reason the actual time used on database view updates is possibly higher.

The downsides of automation, such as hidden complexity and the time automation project takes, were discussed with the developers. It is possible that in the future the developers may experience difficulties when modifying the automation script, as it may take time to comprehend the contents and integrations of the automation. The developer workload may increase momentarily when they will familiarize themselves with the script. The time used during automation project was not experienced to be too much, as the new automated solution can be used as a basis for the future automation. The developers also experienced that the time used on automation will pay itself back within time, as the original manual process was so time-consuming.

7. Conclusion

This study aimed to find out, how automating the repetitive software maintenance tasks affects the software developers and end users of the information system. According to the results, automating the manual tasks of software maintenance takes time, but it is beneficial for the software developers and the end users of the software. Especially if the maintenance task contains multiple phases or occurs frequently, for example weekly or monthly, it is suggested that automation should be considered. According to the findings of this study, the developers reported decreased workload and degree of difficulty, decrease of corrective maintenance tickets and new possibilities for additional automation. The end users reported more up-to-date data within the system and improved database documentation.

The added automation should be documented accurately and the documentation should be updated whenever any changes to the automation artefacts are done. This helps to reduce the hidden complexity of automation and also decreases the workload of the people who will be working with the automated solution. Based on the results of this study, it is also recommended that any automation that occurs frequently should be set autonomous, so that it does not require human operator to trigger it.

In addition to the effects on SW developers, the automation affected the end users of the IS. With the new automation installed, the end users experienced that the data was more current and the new solution provided them with better database documentation. However, as the solution was installed very recently, the customers had a lot of questions related to it and felt that the new solution is arduous to familiarize with. There were also some issues related to the user rights, as they were not automated, but the developers stated that these issues will get sorted soon.

7.1 Limitations

As the approach for this research was case study, it only discussed the situation within one organization and one IS. The results may vary depending on the organization and the automated task. As the automation script was not yet autonomous, but it required a human operator to trigger it, the full benefits of the automation could not be analysed in this research.

Another limitation of this study was the low amount of respondents and interviewees. There were only four developers working on the database view updates and for this reason they were the only ones chosen as interviewees. The customer questionnaire was sent to 23 end users, who were known to have used the reporting database, but only six of them answered to the questionnaire. If the automation had been installed to all customer organizations, the amount of respondents could have been higher.

The limitation in MMG analysis was that some of the maintenance requests were not reported via the ticketing system. Instead of creating a ticket, the developers were approached via email in some cases and these emails were not taken into account in the MMG analysis. This may have created a bias in the quantitative MMG analysis.

7.2 Future research

The future studies regarding the automation of software maintenance, the research could be conducted as a longitudinal study. This would yield more information about the situation before the automation and also the situation after the automation. Future studies regarding automated maintenance processes could focus on automation process that has gained maturity and has been set to be autonomous. This way it would be possible to study whether the hidden complexity and the ironies of automation presented in this study have been addressed and possibly reduced.

Future studies are recommended to also focus on the form and content of instruction manuals of automated processes. As human resources are still required to monitor the process and react to possible malfunctions, it is important that documentation regarding the automated solution is available. It would be beneficial to know what kind of documentation and instructions are needed when familiarizing people with a new solution.

References

Alonso, I. A., Verdún, J. C., & Caro, E. T. (2013). Case study of strategic IT demand management in organizations – exploratory results. *Procedia Technology*, 9, 900-909.

Armstrong, W. B. (2009). *Planning your Survey, A Brief Primer on Strategies and Approaches*. Student Research & Information, University of California-San Diego. Retrieved from http://studentresearch. ucsd. edu/_files/assessment/workshops/2009_Planning YourSurvey. pdf.

Bainbridge, L. (1982). Ironies of automation. *IFAC Proceedings Volumes*, 15(6), 129-135.

Bartusevics, A. (2017). Automation of Continuous Services: What Companies of Latvia Says about It?. *Procedia Computer Science*, 104, 81-88.

Basili, V. R. (1990). Viewing maintenance as reuse-oriented software development. *IEEE software*, 7(1), 19-25.

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

Brinkmann, S., & Kvale, S. (2015). *Interviews: Learning the Craft of Qualitative Research Interviewing*. Thousand Oaks: SAGE Publications Ltd.

Brown, A. B., & Hellerstein, J. L. (2005). Reducing the cost of it operations - is automation always the answer?. In *HotOS*. Retrieved from https://www.usenix.org/legacy/event/hotos05/final_papers/full_papers/brown/brown_html/

Brown, A. B., & Keller, A. (2006). A best practice approach for automating it management processes. In *2006 IEEE/IFIP Network Operations and Management Symposium NOMS 2006* (pp. 33-44). IEEE.

Bruckner, D., Zeilinger, H., & Dietrich, D. (2011). Cognitive automation—Survey of novel artificial general intelligence methods for the automation of human technical environments. *IEEE Transactions on Industrial Informatics*, 8(2), 206-215.

Chimi, C. J., & Russell, D. L. (2009). The Likert scale: A proposal for improvement using quasi-continuous variables. In *Information Systems Education Conference*, Washington, DC (pp. 1-10).

Creswell, J. W., & Creswell, J. D. (2017). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. SAGE Publications.

Cusick, J. J. (2020). Business Value of ITSM. Requirement or Mirage?. arXiv preprint arXiv:2001.00219.

De Vaus, D. (2002). *Surveys in Social Research (5th eds)*. University College London Press, London.

Dubé, L., & Paré, G. (2003). Rigor in information systems positivist case research: current practices, trends, and recommendations. *MIS quarterly*, 597-636.

Düllmann, T. F., Paule, C., & van Hoorn, A. (2018). Exploiting DevOps practices for dependable and secure continuous delivery pipelines. In *2018 IEEE/ACM 4th International Workshop on Rapid Continuous Software Engineering (RCoSE)* (pp. 27-30). IEEE.

Eikebrokk, T. R., & Iden, J. (2012). ITIL implementation: The role of ITIL software and project quality. In *2012 23rd International Workshop on Database and Expert Systems Applications* (pp. 60-64). IEEE.

ElMaraghy, W., ElMaraghy, H., Tomiyama, T., & Monostori, L. (2012). Complexity in engineering design and manufacturing. *CIRP annals*, 61(2), 793-814.

Galup, S. D., Dattero, R., Quan, J. J., & Conger, S. (2009). An overview of IT service management. *Communications of the ACM*, 52(5), 124-127.

Groover, M. P., (2019) Automation. In *Encyclopædia Britannica*. Retrieved from https://www.britannica.com/technology/automation.

Gunawan, H. (2019). Strategic Management for IT Services Using the Information Technology Infrastructure Library (ITIL) Framework. In *2019 International Conference on Information Management and Technology (ICIMTech)* (Vol. 1, pp. 362-366). IEEE.

Hatton, L. (2007). How accurately do engineers predict software maintenance tasks?. *Computer*, 40(2), 64-69.

Herrick, D. R., & Tyndall, J. B. (2013). Sustainable automated software deployment practices. *In Proceedings of the 41st annual ACM SIGUCCS conference on User services* (pp. 189-196).

Hirsjärvi, S., Remes, P., & Sajavaara, P. (2009). *Tutki ja kirjoita*. (15th Edition). Helsinki: Tammi.

Hsieh, H. F., & Shannon, S. E. (2005). *Three approaches to qualitative content analysis. Qualitative health research*, 15(9), 1277-1288.

Hussein, A. (2009). The use of triangulation in social sciences research: Can qualitative and quantitative methods be combined. *Journal of comparative social work*, 1(8), 1-12.

Iden, J., & Eikebrokk, T. (2013). Implementing IT Service Management: A systematic literature review. *International Journal of Information Management*, 33(3), 512-523.

Iden, J., & Eikebrokk, T. (2016). IT service management: exploring ITIL adoption over time in the Nordic countries. *In ITSM Nordic Research Workshop*.

International Organization for Standardization. (2006). *Software Engineering - Software Life Cycle Processes - Maintenance* (ISO Standard No. 14764:2006). https://www.iso.org/standard/39064.html

International Organization for Standardization. (2017). *Systems and Software Engineering: Software Life Cycle Processes*. (ISO Standard No. 12207:2017). https://www.iso.org/standard/63712.html

ISO and IEC (International Organisation for Standardisation and International Electrotechnical Commission). (2010). *Systems and Software Engineering–Life Cycle Management–Part 1: Guide for Life Cycle Management*.

ITSMF. (2007). *An Introductory Overview of ITIL® V3*.

Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2016). What is DevOps? A systematic mapping study on definitions and practices. In Proceedings of the *Scientific Workshop Proceedings of XP2016* (pp. 1-11).

Janssen, C. P., Donker, S. F., Brumby, D. P., & Kun, A. L. (2019). History and future of human-automation interaction. *International Journal of Human-Computer Studies*, 131, 99-107.

Järvinen, P., & Järvinen A. (2000). *Tutkimustyön metodeista*. Tampere: Opinpajan kirja.

Krishnan, G., & Ravindran, V. (2017). IT service management automation and its impact to IT industry. In *2017 International Conference on Computational Intelligence in Data Science (ICCIDS)* (pp. 1-4). IEEE.

Leppänen, M., Kilamo, T., & Mikkonen, T. (2015). Towards post-agile development practices through productized development infrastructure. In Proceedings of *the Second International Workshop on Rapid Continuous Software Engineering* (pp. 34-40). IEEE Press.

Lientz, B., & Swanson, E. B. (1980). *Software maintenance management: a study of the maintenance of computer application software in 487 data processing organizations*. Addison-Wesley.

Limoncelli, T. A. (2018). Documentation is automation. *Communications of the ACM*, 61(6), 48-53.

Marques-Neto, H., Aparecido, G. J., & Valente, M. T. (2013). A quantitative approach for evaluating software maintenance services. *In Proceedings of the 28th Annual ACM Symposium on Applied Computing* (pp. 1068-1073).

Mesquida, A. L., Mas, A., Amengual, E., & Calvo-Manzano, J. A. (2012). IT Service Management Process Improvement based on ISO/IEC 15504: A systematic review. *Information and Software Technology*, 54(3), 239-247.

Metsämuuronen, J. (2005). *Tutkimuksen tekemisen perusteet ihmistieteissä*. 3. Laitos, Helsinki : International Methelp.

Mookerjee, R. (2005). Maintaining enterprise software applications. *Communications of the ACM*, 48(11), 75-79.

Myers, M. D. (2019). *Qualitative research in business and management*. Sage Publications Limited.

Myers, M. D., & Newman, M. (2007). The qualitative interview in IS research: Examining the craft. *Information and organization*, 17(1), 2-26.

Office of Government Commerce. (2007). Continual service improvement / Office of Government Commerce. London.

Office of Government Commerce. (2007). Service design / Office of Government Commerce. London

Office of Government Commerce. (2007). Service operation / Office of Government Commerce. London.

Office of Government Commerce. (2007). Service strategy / Office of Government Commerce. London

Office of Government Commerce. (2007). Service transition / Office of Government Commerce. London.

Pettigrew, A. M. (2013). The Conduct of Qualitative Research in Organizational Settings. *Corporate Governance: An International Review*, 21(2), 123-126.

PostgreSQL. (n.d.). Retrieved from https://www.postgresql.org/docs/9.2/index.html

Singh, Y., Bhatia, P. K., & Sangwan, O. (2009). Predicting software maintenance using fuzzy model. *ACM SIGSOFT Software Engineering Notes*, 34(4), 1-6.

Singh, Y., & Goel, B. (2007). A step towards software preventive maintenance. *ACM SIGSOFT Software Engineering Notes*, 32(4), 10-es.

Sparkes, A. C. (2005). Narrative analysis: exploring the whats and hows of personal stories. *Qualitative research in health care*, 1(1), 191-209.

Stavrinoudis, D., Xenos, M., & Christodoulakis, G. D. (1999). Relation between software metrics and maintainability. *In Proceedings of the FESMA99 International Conference, Federation of European Software Measurement Associations, Amsterdam, The Netherlands* (pp. 465-476).

Strauch, B. (2017). Ironies of automation: Still unresolved after all these years. *IEEE Transactions on Human-Machine Systems*, 48(5), 419-433.

Strauss, A., & Corbin, J. (1998). *Basics of qualitative research techniques*. Thousand Oaks, CA: Sage publications.

Taylor, S. J., Bogdan, R., & DeVault, M. (2015). *Introduction to qualitative research methods: A guidebook and resource*. John Wiley & Sons.

Thummalapenta, S., Sinha, S., Singhania, N., & Chandra, S. (2012). Automating test automation. In *2012 34th International Conference on Software Engineering (ICSE)* (pp. 881-891). IEEE.

Tschiersch, I., & Brandt, D. (1996). Chaos and Automation. *IFAC Proceedings Volumes*, 29(1), 778-783.

Uhlgren, J. (2020). *Database Structure Extension – a Case Study using Design Science Research* (Unpublished master's thesis). University of Oulu.

Vaismoradi, M., Turunen, H., & Bondas, T. (2013). Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study. *Nursing & health sciences*, 15(3), 398-405.

Värttö, A. (2020). *Effects of Automatically Updated Database Documentation on the Work Tasks of IS professionals and End-users: A Case Study* (Unpublished master's thesis). University of Oulu.

Yin, R. K. (2017). *Case Study Research and Applications: Design and Methods*. SAGE Publications.

Appendix A. Interview questions for SW developers

Translated from Finnish

1. *Did you do any manual maintenance tasks related to the old reporting database solution? How often did you do them?*

2. *Were the changes to the old reporting database easy to do? What do you feel was the most challenging about them?*

3. *Did you get customer feedback from the old solution? What was the feedback like?*

4. *Are you familiar with the new automated solution?*

5. *Are the changes to the new reporting database easy to do? What do you feel is the most challenging about them?*

6. *Have you received customer feedback from the new solution? What is the feedback like?*

7. *Do you feel that the new solution has affected your workload? If it has, please describe how. (Brown & Keller, 2006; Herrick & Tyndall, 2013; Krishnan & Ravindran, 2017)*

8. *What factors do you think affected to the overall decision to start automating the database view updates? (Bartusevics, 2017)*

9. *What is your general opinion on automation? Do you think it should be reduced or increased? (Limoncelli, 2018; Bartusevics, 2017)*

10. *How would you describe software maintenance tasks? (Mookerjee, 2005)*

11. *According to your experiences, do you feel you use more time on fixing problems/bugs or enhancing the software? (Lientz and Swanson, 1980)*

12. *Has the automation script affected the maintainability of the software? If it has, describe how. (Brown & Hellerstein, 2005; ElMaraghy, ElMaraghy, Tomiyama, & Monostori, 2012)*

13. *Do you feel like the time used on automating tasks is too high when compared to the benefits of automation? (Bartusevics, 2017; Limoncelli, 2018)*

14. *Is there enough documentation available for you regarding the new automated solution? What kind of documentation would you hope for? (Limoncelli, 2018)*

15. *After the automation, have you noticed changes in the amount or content of maintenance requests related to database views? If you have, please*

describe the changes. (Brown & Keller, 2006; Herrick & Tyndall, 2013; Krishnan & Ravindran, 2017)

Appendix B. Customer questionnaire

Translated from Finnish

Background questions

1. *Estimate your work experience in substance field (in years)*
2. *Estimate, how long you have worked with databases (For example creating queries or interpreting the DB structure)*

General questions

3. *Please describe, how often you use the database views in your work*
4. *Please describe, how often you use the database documentation in your work*
5. *Have you used the earlier solution or are you familiar with it? If yes, answer to these following statements according to your own experience by choosing a value between 1 and 5. Please notice that these statements concern only the earlier solution.*

   (1: Completely disagree, 2: Somewhat disagree, 3: I can't say, 4: Somewhat agree, 5: Completely agree)

   a) *Content of database views is up-to-date*
   b) *Database views are named accordingly*
   c) *I would need more training to use the database*
   d) *It is difficult to utilise the database views*
   e) *The queries made to the database views are fast*
   f) *When I do a change request regarding the database views, it is handled quickly*
   g) *Content of database documentation is up-to-date*
   h) *Database documentation is easy to understand*

6. *Have you used the new solution or are you familiar with it? If yes, answer to these following statements according to your own experience by choosing a value between 1 and 5. Please notice that these statements concern only the new solution.*

   (1: Completely disagree, 2: Somewhat disagree, 3: I can't say, 4: Somewhat agree, 5: Completely agree)

a) *Content of database views is up-to-date*

b) *Database views are named accordingly*

c) *I would need more training to use the database*

d) *It is difficult to utilise the database views*

e) *The queries made to the database views are fast*

f) *When I do a change request regarding the database views, it is handled quickly*

g) *Content of database documentation is up-to-date*

h) *Database documentation is easy to understand*

7. *Please describe your experiences with the new solution and if possible, compare it with the old database solution.*

8. *You can give general feedback and possible suggestions for improvement.*