Tampere University

Ella Holmala

GUIDING SOFTWARE QUALITY ASSURANCE ACTIVITIES THROUGH DATA ANALYTICS

Case M-Files

Master's Thesis
Faculty of Management & Business
Professor Hannu Kärkkäinen
November 2022

ABSTRACT

Ella Holmala: Guiding software quality assurance activities through data analytics
Master's Thesis
Tampere University
Information and Knowledge Management
November 2022

Nowadays, utilizing data in decision-making has become an important topic in organizations. Data is constantly gathered from organizations' internal and external sources, but the full potential of data has not been utilized to maximum extent. The software development context involves a lot of decisions regarding resourcing and prioritization, that are difficult to solve based on intuition. The purpose of this thesis is to find ways for software development quality assurance team to guide testing efforts based on data, instead of intuition.

This case study research was conducted for a company that develops an intelligent information management system for customers globally. More specifically, the research was conducted for the quality assurance team within the product development organization of the case company. The challenge in quality assurance is the constantly growing software product that has increasing demand in testing resources.

The results of this thesis reveal that the QA team in the case company has so far performed well in ensuring quality of the product, but some improvement areas were identified. The research identified relevant information needs and data sources, that could support the optimization of testing activities. This thesis provides a framework including suggestions how to guide and develop quality assurance activities.

Keywords: quality assurance, data analytics, test intelligence

TIIVISTELMÄ

Ella Holmala: Ohjelmiston laadunvarmistuksen toimintojen ohjaaminen data-analytiikan avulla
Diplomityö
Tampereen yliopisto
Tietojohtaminen
Marraskuu 2022

Nykypäivänä, datan hyödyntäminen päätöksenteossa on tullut yhä tärkeämmäksi aiheeksi organisaatioissa. Dataa kerätään organisaation sisäisistä ja ulkoisista lähteistä jatkuvasti, mutta datan potentiaalia ei olla vielä hyödynnetty maksimaalisesti. Ohjelmistokehityksen piiriin liittyy paljon resursointiin ja priorisointiin liittyviä päätöksiä, joita on hankala ratkoa intuitiopohjaisesti. Työn tarkoituksena on selvittää tapoja, miten ohjelmistotestauksen testausresursseja voidaan allokoida dataohjautuvasti intuition sijaan.

Tutkimus toteutettiin tapaustutkimuksena, jonka kohdeyritys kehittää älykästä tiedostonhallintaohjelmistoa. Työ toteutetaan kohdeyrityksen tuotekehityksen laadunvarmistusosastolle toisin sanoen ohjelmistotestauksesta vastaavalle osastolle. Laadunvarmistuksessa haasteena on kasvava/skaalautuva tuote, joka vastaavasti vaatii kasvavia testausresursseja. Lisähaasteena on myös tietämättömyys siitä, mitkä tuotteen alueet ovat tärkeitä asiakkaan kannalta ja miten testausresursseja kannattaisi järjestää. Ohjelmistotestauksessa on tärkeä ymmärtää asiakkaiden käyttäjiä ja millaisia tarpeita tuotteen käyttäminen täyttää, jotta voidaan arvioida vastaako kehitettävä tuote asiakkaan käyttäjien tarpeita. Toisin sanoen, testaajien on hyvä ymmärtää mitkä asiat luovat arvoa käyttäjille. Työn tavoitteena on tunnistaa, miten testausresursseja voidaan optimoida dataohjautuvasti. Lisäksi tavoitteena on ymmärtää, mitä tietoa laadunvarmistuksessa tarvitaan onnistuakseen tehtävissään.

Työssä saatiin selville, että kohdeorganisaation laadunvarmistusosasto on suoriutunut toiminnoissaan hyvin tuotteen laadun kannalta, mutta toiminnasta löytyi kehitettäviä asioita. Työssä tunnistettiin relevantteja tietotarpeita sekä tietolähteitä, jotka voivat tukea testausresurssien optimointia. Työ tarjoaa viitekehyksen kehitysehdotuksista, mitkä voivat auttaa laadunvarmistuksen toimintojen ohjaamisessa ja kehittämisessä.

Avainsanat: laadunvarmistus, data-analytiikka, testausälykkyys

PREFACE

During the past year, I've had the great opportunity to write this thesis from the lens of information and knowledge management, and this project has been a great learning experience. This process has opened my eyes that the practical need for information and knowledge management is legitimate. There is a lot to do in the field, which is exciting to look forward to. Even defining the scope for this thesis was a real challenge due to many interesting topics!

First, I would like to thank Antti Leinonen, Tero Piirainen and Minna Vallius for giving this interesting topic for me to study. Especially thank you Antti for being patient with my work, and giving the time and resources needed to finish this project. Thank you to every interview participant taking the time and helping me gather information needed for this thesis. Thank you also to my colleagues for proof-reading my work. I would also like to thank my university advisers Hannu Kärkkäinen and Nina Helander for revising my work.

Thank you to my family for all the support you have provided during my studies, you have been there through the ups and downs. To my parents, I truly admire your work ethic and I look up to both of you. I hope to become as hardworking in my career like both of you have been. To my brothers, thank you for being supportive throughout my life, you both have been great examples to look up to.

Lastly, I would like to thank all my friends and Tietojohtajakilta Man@ger for the amazing memories during the past years, it has been quite a ride!

In Tampere, 14.11.2022

Ella Holmala

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

API	Application Programming Interface
CI	Continuous Integration
CoP	Community of Practice
CD	Continuous Delivery
CTP	Customer Touchpoint
DevOps	Development and Operations
DoD	Definition of Done
NPS	Net Promoter Score
QA	Quality Assurance
R&D	Research and Development
SaaS	Software as a Service
TTD	Test-Driven Development
UAT	User Acceptance Testing
UI	User Interface
UX	User Experience
VCS	Version Control System
XP	Extreme Programming

1. INTRODUCTION

The purpose of this thesis is to study the topics of software quality assurance, data analytics, and information management from the software product development perspective. This thesis is carried out as a qualitative interview research for a software company in Finland. More specifically, the study focuses on the quality assurance (QA) team within the product development organization, that is responsible for ensuring high product quality for customers. In practice, QA involves testing the product to guarantee the quality is on a proper level. The aim of this study is to recognize the information gaps and areas, where data analytics can be applied to optimize activities in quality assurance. The results will help the QA team to recognize the current opportunities and take the necessary actions to optimize their activities to become more data-driven. The introduction presents background information of the chosen topic based on literature and the case company. Later, the research objectives and questions are defined with consideration of the context.

1.1 Background and motivation

Ensuring high quality in software development is a complex task. In software product development, QA is responsible for ensuring customers do not run into issues while using the product. For example, QA tests each development increment to ensure changes work as expected. Before the software is released, there is a need to guarantee that the product works based on expectations or requirements. Not only should the product work from a technical perspective, but the product should serve a purpose that customers find valuable. Thus, it is vital to understand who the end-users are, what they value and how they behave while using the product.

Nowadays, Agile software development methodology is common, where software releases are done in shorter cycles on a regular basis, in contrast to the traditional waterfall methodology with longer release cycles. (Sahoo & Pattnaik 2020) For example, short release cycles can mean monthly releases, while longer cycles can be yearly or less often. For QA, shorter release cycles mean higher release test frequency. In addition to higher frequency, systems are becoming more complex and the demand for testing efforts keep increasing.

Sahoo & Pattnaik (2020) also explained how agile method allows better involvement of customers and stakeholders in the development process. Several stakeholders can affect the outcome or be affected by the outcome of created software. Therefore, it is important to

identify the key stakeholders and search for ways to connect with them to improve stakeholder interest and product involvement. (Sahoo & Pattnaik 2020)

In the digital world, a lot of data is being created every day. Alsaqqa et al. (2020) defined the term big data as follows "massive data files that have a large, diverse and complex structure and are therefore difficult to store, analyze and visualize for other processes and results". Big data is usually referred to the three V's: Velocity, Volume and Variety. Volume relates to the large data amounts generated from many sources in one period of time. Velocity relates to the generation or analysis speed of data. Variety refers to the sources and type of data that are either structured or unstructured. Structured data, for example, refers to data that can be stored in a database. Unstructured data, like images, videos, and audio, refers to data that do not have a structure and need more processing. (Alsaqqa et al. 2020) Although organizations collect data in large amounts daily, data is not always utilized to its full potential. While data is being gathered in vast amounts, data analytics offers an opportunity to make sense of the data.

Organizations with business units in different silos typically have siloed organizational data and information. For example, sales might have their own information systems with saved data of customers and sales transactions. This information might also be valuable to other business units, such as production and other units. Ensuring information and data flows between silos requires understanding of business processes, information system interfaces, architectures, infrastructures, and data types. (Laihonen et al. 2013)

This thesis focuses on the quality assurance team of the Finnish software company, M-Files Oy. M-Files develops an intelligent information management software solution for businesses around the globe. The growth in M-Files has been fast in the recent years. Not only has the company itself grown, but also the size of the product (M-Files 2021a). Serving an increasing customer base requires M-Files to improve scalability. Due to the growing platform, there is a need for quality assurance to allocate resources intelligently to ensure high quality. Thus, identifying what QA needs to improve data-driven activities and recognizing data analytics capabilities is essential to make the transformation. Since QA activities are influenced by other internal stakeholders, there is a need to research relevant activities affecting QA.

The motivation for this thesis is driven by the goal to achieve desirable customer satisfaction with the developed product. From an information and knowledge management perspective, it is interesting to find out how QA and customers can be brought closer with better information and data flow. Utilizing data analytics to enhance activities is fascinating to study, as data analytics is certainly a relevant topic nowadays.

1.2 Research objectives and research questions

For this thesis, the main research question is “In what ways can QA activities become customer-centric and data-driven?”. To get a more thorough understanding about the context, some sub-questions were also formed:

“What type of activities does QA consist of?”

"What challenges relate to performing QA activities?"

“What type of internal stakeholder activities relevantly affect QA activities?”

“What customer information is needed for QA to improve activities?”

“What type of data analytics can and should be established?”

Currently, the case company has existing data analytics capabilities, but they are not utilized now from QA point-of-view. For that reason, this thesis will focus on investigating what QA wants and needs to know, and where to apply data analytics to help in optimization. With the investigation and analysis of the topics, the aim is to identify key challenges in the product development organization affecting QA activities, recognize the areas where data analytics can be applied and provide a proposal to the quality assurance team to optimize their activities. Suggestions are also provided on the product organization level to stakeholder activities that relevantly affect QA. A more detailed explanation of the case company will be presented in chapter 4.

2. SOFTWARE QUALITY ASSURANCE

This chapter will dive into the role of quality assurance in software product development. Moreover, this chapter will reveal how literature defines the typical activities in software development and QA, relevant stakeholders, and general challenges. Quality assurance in continuous product development has many similarities compared to software projects. In software projects, delivery is typically only for a certain customer. In contrast, continuous project development has often several customers with different needs. Software engineering literature mainly focuses on software projects. In this thesis, the literature findings will be applied to the continuous development context.

2.1 Definition of quality

The meaning of quality is nontrivial because the concept of quality is multidimensional, including both objective and subjective dimensions (Mistrik et al. 2016, chapter 13). Mistrik et al. focus on two main perspectives of quality:

1. conformance to specification and

2. customer satisfaction. (Mistrik et al. 2016, chapter 13)

The conformance to specification view sees the definition of quality as measurable characteristics that satisfy a specification. Customer satisfaction view on quality describes whether customer expectations, explicit or not, are met. (Mistrik et al. 2016, chapter 13) The definition of quality by Feigenbaum states how the customers actual experience with the product or service is measured against requirements that can be "*stated or unstated, conscious or merely sensed, technically operational or entirely subjective*" (Feigenbaum 1983; Mistrik et al. 2016, chapter 13). Quality is also defined by ISO Standard 9001 as "*the degree to which a set of characteristics fulfils requirements*". (ISO 2000; Mistrik et al. 2016, chapter 13)

Nader-Rezvani (2019) illustrated the key stakeholders affecting holistic quality in product development.

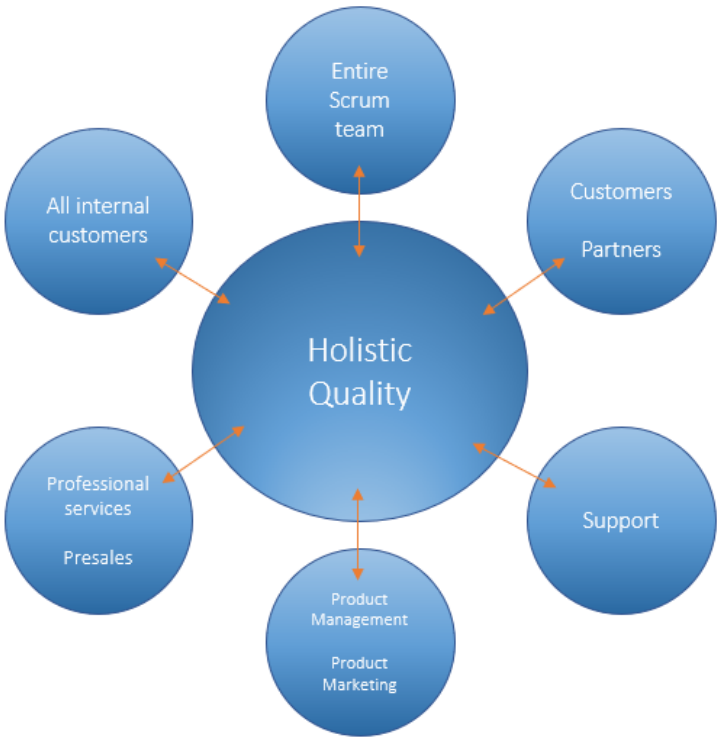

Figure 1: *The key functions in most product development teams (modified Nader-Rezvani 2019)*

As Figure 1 illustrates, holistic quality is affected by many stakeholders, such as the scrum team, customer support, product management and marketing, services team and even pre-sales. In an organization, it is important to have everybody informed and made sure everyone is aware of the quality improvement plan. By such understanding, each action from idea conception to product delivery, to customer deployment, up to sales and support, are all aligned to achieve quality objectives of the organization. Building quality culture needs the following key pieces: open communication, common information, and a sense of shared accountability. (Nader-Rezvani 2019)

2.2 Quality assurance in software development

Quality assurance is a crucial activity in software development where it typically refers to software testing. According to Galin (2018), software quality assurance aims to minimize cost of ensuring a software product's quality by introducing a variety of infrastructure activities and other activities during development and maintenance processes. In other words, the objective is to prevent the reasons for causing errors and detecting them as early as possible. Moreover, the quality of the software product can be brought to an acceptable level. The activities are performed in all stages of development to detect errors, remove the causes of the errors, and correct them in early stages of development. Ultimately, quality assurance reduces the number of products that are not qualified. The set of activities both define and

assess the adequacy of software development processes, providing evidence that establishes confidence that the processes are appropriate and produce suitable quality for their intended processes. Some objectives are:

- Ensuring conformance to functional technical requirements with acceptable levels of confidence.
- Ensuring conformance to requirements relating to schedules and budgeting with acceptable levels of confidence.
- Initiating and managing activities that improve and increase efficiency of development and QA activities. (Galin 2018)

The traditional V-Model illustrates the types of testing and how testing is an ongoing activity throughout the software lifecycle. (Mili & Tchier 2015)

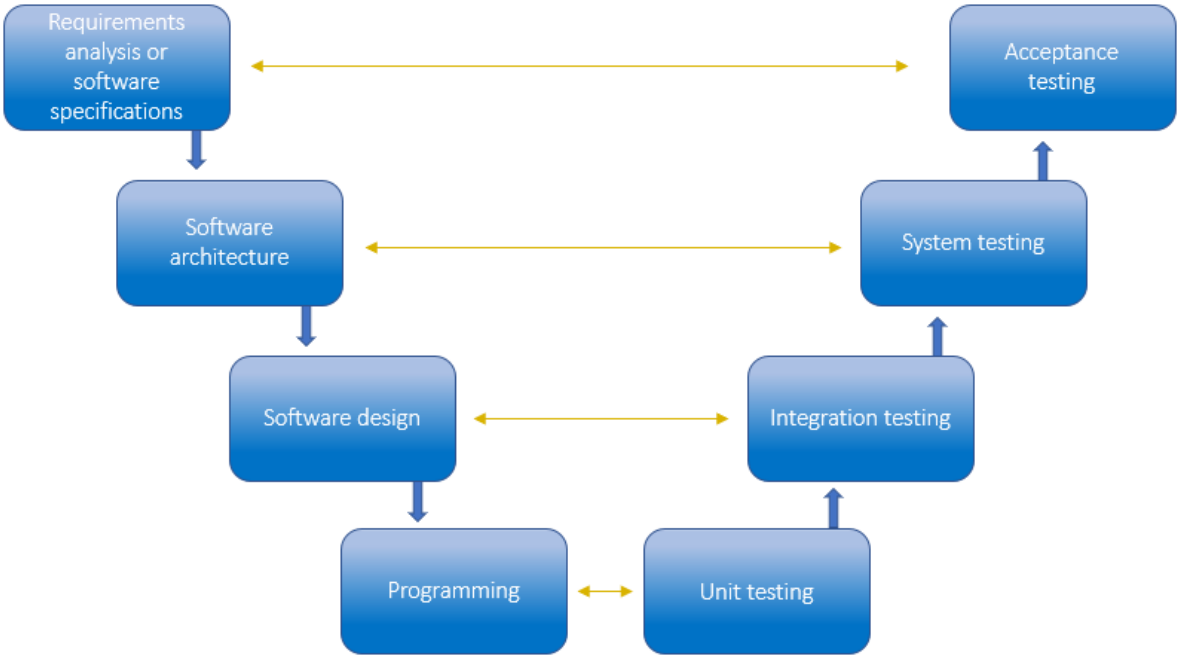

Figure 2: *V-Model of software testing (modified Mili & Tchier 2015)*

In Figure 2, phases connected with a horizontal arrow are related to each other. Activities on the left side of the branch prepare the corresponding phase on the right, while activities on the right side of the branch test the validity of the former. Unit testing refers to testing individual modules against module specifications. Integration testing includes tests of specific attributes of the software products. More specifically, integration tests test the capability of system components interaction based on the system design. System testing is a service for the development team and the goal is to find as many faults as possible before the last phase of testing. Lastly, acceptance testing aims to show the ultimate behavior is according to initial specifications. (Mili & Tchier 2015)

During the software life cycle, three popular quality assurance activities are testing, reviewing and process auditing. Based on research findings, reviewing is one of the most effective quality assurance activities and the best industry practice, since it aims to prevent defects and save time used for later fixing. Usually, review refers to requirements review, formal design reviews and peer reviews (inspection and walkthrough). Although reviews are found to be the most effective, it should not replace testing, as they are overlapping activities. (Li et al. 2010)

Software testing is one area of quality assurance. The goal of testing is to identify software failures and evaluate the level of quality to result in user satisfaction (Homès 2012, p.7). According to Batarseh and Gonzales (2018), testing includes two activities: validation and verification. The relationship between requirements and the product are illustrated in Figure 3.

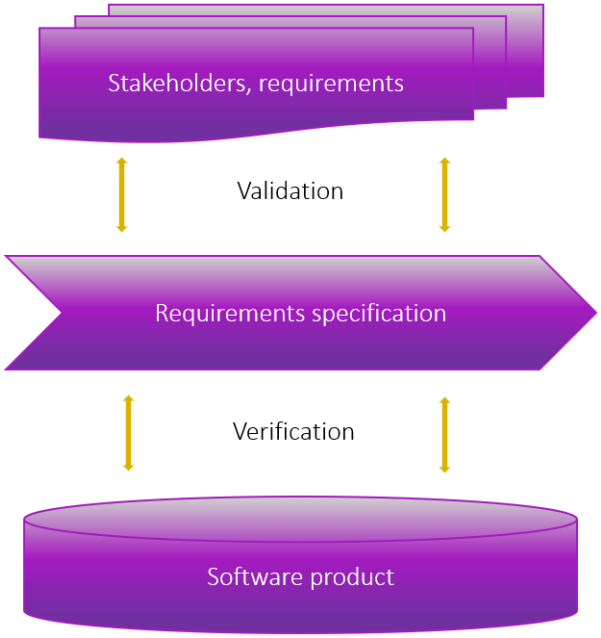

Figure 3: *Verification and validation (modified Mili & Tchier 2015)*

Validation is about confirming development meets the requirements and verification is about making sure the system matches the functional specification. (Batarseh & Gonzales 2018) Furthermore, verification answers the question "*have we produced what is specified?*", whereas validation answers the question "*have we build the correct product?*" (Homès 2012, p.7).

To identify failures in software, testers need to perform certain testing activities based on a testing process (Homès 2012, p.14). Figure 4 depicts the fundamental testing process.

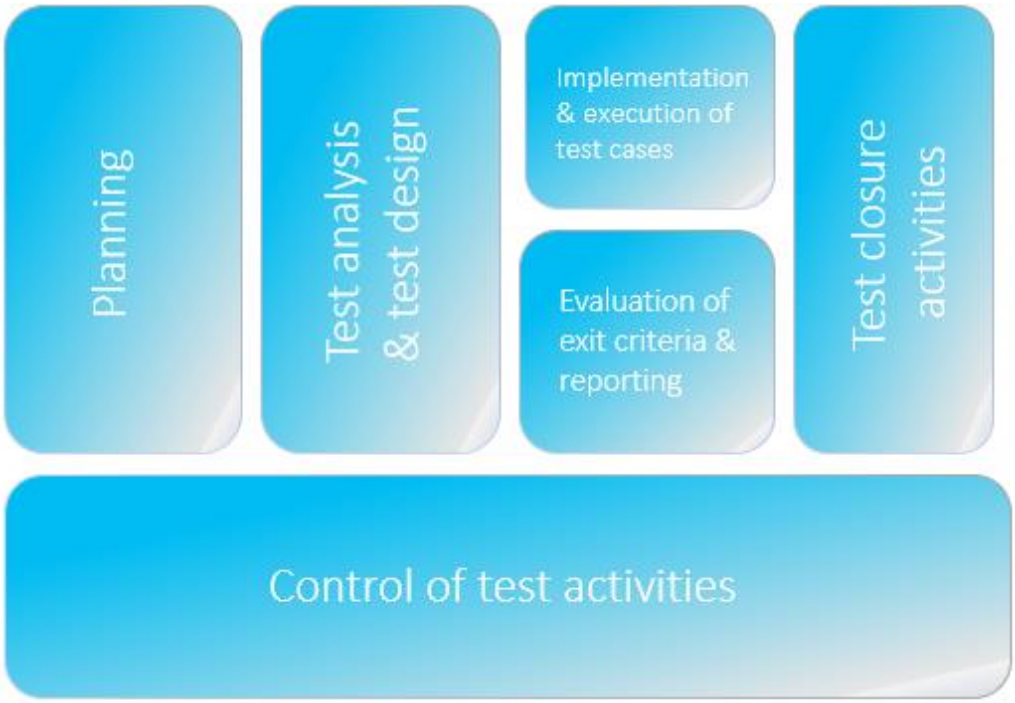

Figure 4: *Test activities in the fundamental test process (modified Homès 2012, p.14)*

Testing usually includes the following activities: planning, defining goals, identifying test conditions, creating test data, start and exit criteria, test environments and controlling all these activities. Test planning includes the definition of test goals and the activities to reach these goals. During the test analysis and design step, test conditions for the software are created. The previously defined goals help create test design documents, test procedures and test conditions. Test implementation involves using the test conditions to convert them into test cases and test procedures with test data and specific expected results. To anticipate test execution, precise information is needed on the test environment, test data and the sequence of test cases. The test execution is performed in the set test environment and the actual test result if compared to the initial, expected results. During the test execution, the exit criteria are also analyzed to see whether the test object fulfils the test objectives defined in the planning phase. Test activity results are then reported to relevant stakeholders. When the software or system is considered ready for delivery or project is considered complete, the test activities are closed. (Homès 2012, p.14-21)

Executing manual testing activities on large software systems can be time consuming. According to Goericke (2020), fast growing demand for testing requires companies to invest in test automation and continuous integration (CI) (Goericke 2020, p.2). More and more tests are being automated. In agile development, test automation is important especially in regression tests, i.e., testing existing functionality (Baumgartner et al. 2021, p.17). The benefit of automation is the fast feedback loops. (Goericke 2020, p.34).

Continuous integration refers to a practice where code changes are committed to a common repository many times a day. Continuous delivery (CD) is an interrelated practice where the automation ensures every build (deriving from the common repository) available is stable and verified. Moreover, CD ensures that a software build could be pushed to production at any time. (Bobrovskis & Jurenoks, 2018) Figure 5 illustrates how CD relates to the DevOps (development and operations) production and delivery process.

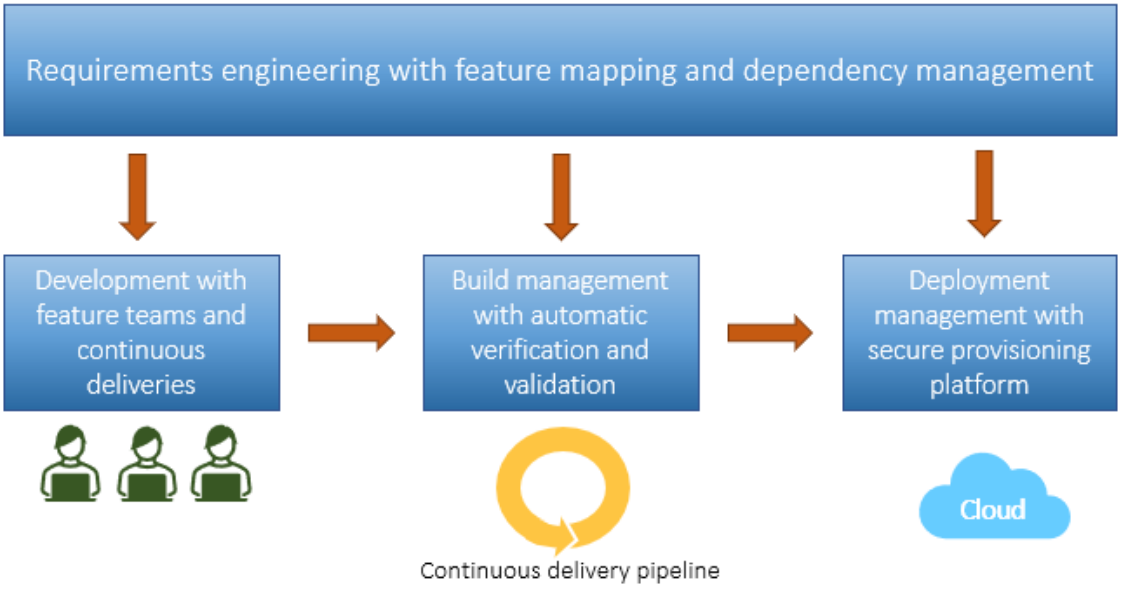

Figure 5: *Typical DevOps production and delivery process (modified Ebert et al. 2016)*

DevOps combines development and operations using automated development, deployment and infrastructure monitoring. In DevOps, the culture between development, quality assurance and operations are changed to be collaborative. Through DevOps, continuous delivery with small upgrades is possible in organizations. (Ebert et al. 2016).

2.3 Agile development and stakeholders

In the software industry, there are several types of development practices used. For example, Scrum, Extreme Programming (XP) and Kanban are de facto standards in agile development. The most popular practice is Scrum, where the aim is to deliver functioning software early and frequently. The Scrum practice also focuses on communication, speed and developer collaboration in the process. In Scrum, there are roles such as the Product Owner, Scrum Master and Scrum Team. The purpose of the Product Owner is to represent ideas and needs in between stakeholders, consisting of the customer as well. The Product Owner creates *user stories* that represent user's needs and the business value in sentence format with a distinct structure. These user stories are managed in a list which is referred to as the product backlog. The Scrum Master is in charge of the process and is considered as a coach for the Scrum Team. The Scrum Team is a team of individuals who work independently and

are self-organized. The Scrum Master works as a facilitator for the Scrum team to remove obstacles during the process. The team has a daily Scrum meeting, where each member updates their progress of current workings and the plan for that day. (Larusdottir et al. 2016).

The Agile Manifesto, which was released in 2001, involves 12 main principles of agile software development. According to the Agile Manifesto, software development is not only recognized as a technical activity since it needs input from other stakeholders who are not from the technical field. The principles include topics of e.g., satisfying customers, teamwork between developers and business people, as well as creating projects with individuals who are motivated. (Larusdottir et al. 2016)

In software product development, quality assurance and development have a close relationship. For example, QA activities, like design reviews and software tests, are integrated in software development activities. (Galin 2018) Testing efforts interest many stakeholders, therefore reporting activities can be used to keep each stakeholder updated. Not only does reporting concern stakeholders, but also testers themselves to evaluate their progress and efficiency. Development is interested in the code quality and if any defects need to be fixed. QA managers are interested if there is a need to determine process improvement activities. Customers and end users, or even marketing, also need to be informed if the software is ready for release. The evaluation of outstanding expenses and efficiency of activities by upper management requires information from testing activities. All stakeholders should be informed either via progress reports, statistics or graphs, to give adequate information for decision-making. (Homès 2012, p.21) Although these are studied based on software projects, this can be applied to continuous product development context.

In software projects, the evolution and progress of the project can be shown through trends on graphs. Summarized information can be visualized with designed dashboards. Even time-over-time diagrams allow to detect possible delays of projects. (Homès 2012, p.243) However, this is more useful to project-like working. In continuous product development, where product releases are delivered frequently, each product release can be considered as a short project.

Within a product development organization, there are many individuals who work in different areas. Coming from different perspectives, individuals may understand the same aspects of a product in another way. The issue is even worse when work is transferred from team to another sequentially. Instead, cross-functional teams are found to have more effective information flow when solving problems. (Barney 2014)

2.4 Challenges in quality assurance

According to Baumgartner et al. (2021), one decisive factor in agile development is time. Within a given timeframe, testers need to ensure that the quality is as high as possible. When

the timeframe is shortened, often comes in the expense of the quality aspect – resulting in technical debt. (Baumgartner et al. 2021, p.14)

Quality and deadline are related to the term "*time to market*". In fast-moving times, the time between product development and product launch to the market is becoming an increasingly important factor. The idea is to establish new products that are both of interest to the end customer but also to keep the development and delivery short. In software projects, another challenge is to meet requirements with a certain budget. (Baumgartner et al. 2021, p.18).

Shuib & Hassan (2021) explained how adopting a QA mechanism in Scrum method has some challenges. For example, if customers are not available during the review process, the validation will be late. Another issue is the challenge of getting feedback effectively since gathering and managing feedback can be quite tedious, especially if the communication medium is not effective enough. (Shuib & Hassan 2021) These issues mainly relate to software projects, where customers are more directly involved.

One challenge is the large platform support matrix (PSM), which refers to not deprecating older platforms, operating systems, browser versions and databases. When no action of deprecation is performed, customers continue to expect new releases to be backward compatible with various versions. Certifying many different configurations is not scalable which causes quality challenges. To address the challenge, streamlining platform support matrix can be done by deprecating old platforms or versions. To help streamline the PSM matrix, there are three areas of information needed to be captured:

- The most common configurations customers use
- Configurations that have created most support calls
- The trend within the industry on supporting different combinations and configurations. (Nader-Rezvani 2019)

Nader-Rezvani (2019) calls it "*Supported Versus Certified*" - only selected specific configurations are considered "*certified"*, by which new software is validated during the release cycle. All others are "*supported*", where if customers face problems, support can be contacted for help. This approach allows performing business and cost analysis when key reference architectures can be focused on. Over time, customers could be encouraged, by customer-facing departments, to migrate their infrastructure to the key reference architecture which are fully certified internally. The deprecation of platforms that are no longer popular among most customers is then allowed by product management. The concept of *supported versus certified* can be defined by analyzing support data, the input from presales on the new market trends and the input of services on common challenges with specific configurations. (Nader-Rezvani 2019)

Practitioners consider test management and test automation to be the most challenging activities in testing. In test management, the effectiveness and efficiency of testing is difficult to assess. Finding suitable testing approaches for a given context and estimating the return on investment (ROI) is challenging. (Garousi et al. 2020) Goericke (2020) also addressed the challenge of automating testing where teams face the “legacy of inherited code” that has been written for over a decade and is in use by customers. Automating such systems require refactoring, which is a difficult situation if there are no existing unit tests. Therefore, from an engineering perspective test automation might make sense, but from ROI viewpoint, it might not. (Goericke 2020, p.114)

Modern software development approaches have changed how software is developed today. The main difference is the speed at which products are released and the higher frequency of releases. One main challenge is to upkeep development speed without having to sacrifice quality. To attain ‘software quality at speed’, test automation is crucial. However, organizations still lack test automation maturity, i.e. test automation practices are not mature enough. According to ISTQB’s survey, 65% of 200 software organizations in the world have made efforts to improve test automation maturity, while half of them claimed the attempts to be effective. (Wang et al. 2021)

Test automation maturity can be considered from three different dimensions:

- Test automation process
- Test automation technology maturity
- People capability.

Test automation process maturity refers to having an explicitly defined, managed, measured, controlled and effective test automation process. Test automation technology maturity refers to test automation technology that reaches a pre-defined level of sophistication to meet the needs of software development. Such level of sophistication can allow increased test scale, test coverage, test efficiency and test effectiveness. The people capability maturity is measured by the ability to leverage competencies of workforce in test automation. (Wang et al. 2021)

2.5 Agile testing quadrants

In releases, Agile teams need to manage the scope to ensure business deadlines are met while quality is preserved. The first priority is high-value user stories. The focus should be on delivering value, and release planning is the time when examples of use cases for features are investigated and the value they generate. (Crispin & Gregory 2009, p.340-341) From testing perspective, the agile testing quadrants help make sure all different types of tests

needed are considered to deliver value. (Crispin & Gregory 2009, p. 97) Each quadrant in Figure 6 demonstrates the various reasons for testing.

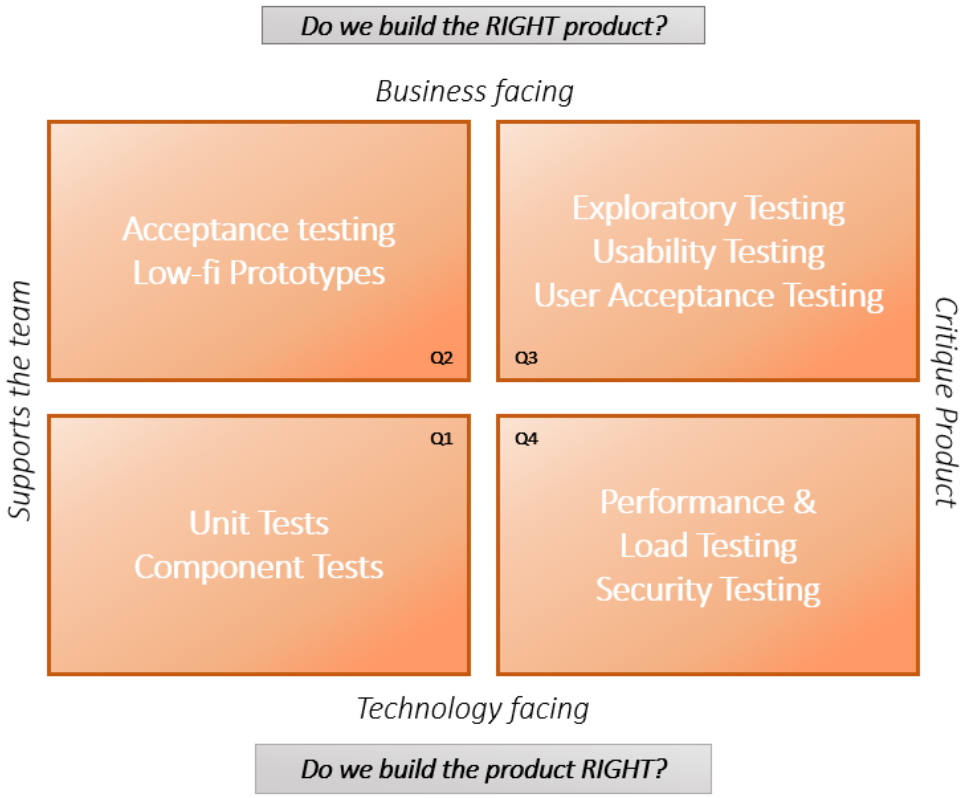

Figure 6: *Agile testing quadrants (modified Crispin & Gregory 2009, p.97; Goericke 2020, p.113)*

The quadrants on the vertical axis are divided into tests that support the team and tests that critique the product. The quadrants on the horizontal axis are divided into tests that are either business-facing or technology-facing. (Crispin & Gregory 2009, p.97)

Quadrant 1 represents test-driven development (TDD), which helps programmers design code in a good way. Unit tests are used to verify functionality of a minor part of the system, like an object or a method. On the other hand, component tests are used to verify a larger sector of the system, for example a group of classes that provide a service. (Crispin & Gregory 2009, p.99)

Tests in the second quadrant support the development team's work on a high level. Such business-facing tests define not only the external quality, but also the features that customers want. These tests are developed from examples that the customer team provides. Tests in the business-facing area are run at a functional level and they each verify a business satisfaction condition. The tests are written in understandable business domain language for business experts. Moreover, these tests are used by business experts to define the external quality of the product and they are involved in writing the tests. The second quadrant aims to

illustrate and confirm the system behavior is as intended at a higher level. (Crispin & Gregory 2009, p.99)

The third and fourth quadrants focus on critiquing the product. Quadrant 3 includes the classification of business-facing tests that focus on exercising the working software to look if it doesn't meet expectations or doesn't stand up to competition. This involves manual testing which only humans can do. The idea is to emulate how real users would work the application, when doing the business-facing tests to critique the product. It requires using brains, senses, and intuition to see whether the business value required by customers is delivered by the development team. (Crispin & Gregory 2009, p.101)

Customers and users often perform such tests. User acceptance testing (UAT) is a way to allow customers the chance to try new features and get feedback on what features they might want and what new stories could be created. In usability testing, focus groups may be interviewed to study the use of applications and interview to collect their reactions. For instance, the usability testing can include the navigation patterns from page to page or tabbing patterns. Having the knowledge of how users use the system is an asset when testing usability. (Crispin & Gregory 2009, p.102)

The fourth quadrant includes the technology-facing tests that critique the product by the following characteristics: performance, robustness, and security. These types of tests may require the use of specialized tools and extra expertise. The product critiquing, technology-facing tests should be considered at each step of the development cycle and leaving it until the very end should be avoided. (Crispin & Gregory 2009, p.102)

The Agile Testing Quadrants gives a tool to categorize tests conveniently. They give guidance to make sure all angles of product quality are covered both in the testing and developing process. (Crispin & Gregory 2009, p.108) Business-oriented testing makes sure the product is built right from a user and market perspective, while technology-oriented testing makes sure the product is built right. As the different types of testing require different expertise, it is the team's responsibility to embrace the different aspects of quality and engage in activities that contribute to the various types of testing. (Goericke 2020, p.113)

3. QUALITY ASSURANCE DRIVEN BY CUSTOMER VALUE

This chapter covers the topics of customer value, information and knowledge management, data analytics, business intelligence and test intelligence.

3.1 Customer value creation and customer experience

Customer value has various definitions in literature. Customer value can be perceived from two perspectives – value for the customer and value for the firm. The types of value can be categorized into four dimensions. Functional or instrumental value refers to the degree of which a product (either good or service) has desired characteristics, is useful and or has the wanted function. Experiential or hedonic value refers to experiences, feelings and emotions that a product creates. Symbolic or expressive value is concerned with the association of psychological meaning to a product. (Smith & Colgate 2007) Although customer value creation is a marketing concept, it is central for other organizational areas to also understand the factors contributing to customer value.

Key sources of customer value are information, products, interactions, environment, and ownership/possession transfer. The variety of “value chain” processes and activities within and between organizations create these sources of value. Each value source provides different kind of value, such as cost/sacrifice value, functional/instrumental value, experiential/hedonic value, and symbolic/expressive value. For example, cost/sacrifice value can be generated through information when consumers can make more informed and faster decisions. Products can create e.g., functional/instrumental value, like safety relates to Volvo automobiles. Value created by the purchasing environment can bring symbolic/expressive value, i.e., holiday ornaments that appeal to cultural traditions. Lastly, processes relating to ownership/possession transfer, such as timely delivery, can create functional/instrumental value. (Smith & Colgate 2007) Thus, customer value accounts for much more than monetary costs and benefits of product or service consumption. Smith and Colgate (2007) developed a framework for customer value creation strategies. The tool helps describe a generic marketing strategy, improve the specifications of product concepts, identify opportunities for value creation and develop customer value measures. (Smith & Colgate 2007)

In the software industry, value-based software engineering has received attention in the past recent years. Companies in the industry have attempted to define ways to integrate value-based decision making in product development and evolution life cycle to sustain competitive advantage and growth. Value-based decision making can help in the development of high-quality products within a timeframe and budget by taking the correct strategic decisions that

recognize the benefit, value and risk factors all through the life cycle. (Marciuska et al. 2013) Software development can become more value-based by conducting more thorough analysis of the benefits to be gained by the software, eliciting value propositions that stakeholders have and conducting business case analyses on the software projects. One Norwegian development project had used "*benefit points*" relative to an epic (a set of user stories), similar to story points, with "known" value to customer organization. Such figures can help make decisions regarding the priority in a product backlog. (Dingsøyr & Lassenius 2016).

How to minimize waste in software development efforts that do not deliver value for customers or users is a critical question in software engineering. Many companies rely on intuition and experience when decisions about product development are made, which can lead to development and business risks. One way to avoid risks is to frequently identify, prioritize and validate critical product assumptions. Experiments can help in testing product assumptions. Transforming assumptions into hypotheses and the scientific method is applied to either approve or refute the hypotheses. Product decisions can therefore be influenced by the results of experiments. The experiment-driven approach is especially prevalent in Software-as-a-service (SaaS) companies, in the environment of cloud computing. The approach can be seen as a subset of data-driven decision-making approaches because the experimentation provides a method to back up decisions with verifiable data. (Lindgren & Münch 2016)

Lindgren and Münch (2016) also found that many companies had acknowledged the potential in product usage data and had plans to advance procedures in this regard. However, in the business-to-business (B2B) domain, there are challenges regarding experimentation, where customers often are not users of the product. Compared to business-to-consumer (B2C) business, the value for users is not necessarily the same as the value for customers in B2B. (Lindgren & Münch 2016)

There is a statement that only 20% of software components create 80% of business value. The challenge lies in estimating which features improve customer perceived value at the time of requirements analysis and prioritization of them for the upcoming release. Another part of the challenge is to follow the perceived value of features that already exist and are in use in a product. This includes recognizing and erasing low value generating features for the next product release. (Marciuska et al. 2013)

The dimensions of software business value can be differentiated into three dimensions: product value, customer's perceived value and relationship value. The product value refers to the price of the product, which might change depending on other competitive products. Customer's perceived value refers to the benefit that is gained by the customer from the product, which can be shaped by customer expectations, past experiences, or even cultural back-

ground. Lastly, relationship value is created through the establishment of good social connections and an effortless flow of business communications between a company and it's clients. (Marciuska et al. 2013)

In knowledge-creation, customers play a vital role and customer knowledge is recognized as an important resource for companies. The recognition assures that there is a need for processes that facilitate knowledge creation, transfer and leverage both on an individual and collective level. Knowledge and value co-creation with customers, suppliers and business partners is increasingly more important for organizations. Value co-creation relates to the opportunity to gain competitive advantage by the development of unique competences with correct organizational resources and technological capabilities. The aim is to satisfy the demand for personalized products, services, and experiences. To achieve and sustain competitive advantage, value creation should be required in software engineering. Furthermore, it is important in software engineering to understand customer value and take it into account. (Aho & Uden 2013).

3.2 Information and knowledge management in organizations

Information and knowledge management aims to improve the performance and flow of working in organizations. Organizations and individuals are constantly looking for new ways to acquire and utilize information. Understanding information-based value creation logic is one key information and knowledge management challenge, and foundation for management development. The idea is how to create value with information. Value is created when information is utilized intelligently in both in operative work as well as activity development. (Laihonen et al. 2013, p. 7-12)

3.2.1 Organizational learning and information management cycle

Pursuing goals in an intelligent organization with changing external environment requires salient adaptation of behavior based on knowledge about its external and internal settings. To clarify, intelligent organizations are learning organizations, which can create, acquire and transfer knowledge, and even change their behavior to reflect new knowledge and insights. Therefore, learning starts when new knowledge and ideas are applied to change the organization’s goals (Choo 2002, p. 12)

Effective organizational learning requires going further than only detecting and correcting errors from the past. Organizational learning occurs when its members respond to changes happening in external and internal environments by detecting and correcting errors between outcomes and expectations. To bring outcomes and expectations back into line, correction

of errors is made through modifying organizational strategies, assumptions or norms. Organizational learning also includes unlearning, moreover, not restricting learning to only existing practices but also rediscovering new goals and re-examining existing norms and assumptions. (Choo 2002, p. 13)

Organizational learning can be divided into two types: single-loop and double-loop learning. Single-loop learning refers to learning, where modification of organizational action is enough to correct the error without challenging existing norms. Double-loop learning, on the other hand, happens when error correction requires the modification of norms, which means having to modify strategies and assumptions associated to those norms. Organizations are often good at single-loop learning since they measure performance based on set objectives and changing operational procedures help correct deviations. Organizations seldom challenge their own norms, goals, or policies in relation to their changing environments. (Choo 2002, p.14)

The main goal of information management is to capture information resources and capabilities that allow organizations to learn from and adapt to changes in their environment. (Choo 2002, p.24) Figure 7 depicts organizational learning in more detail by Choo's information management model.

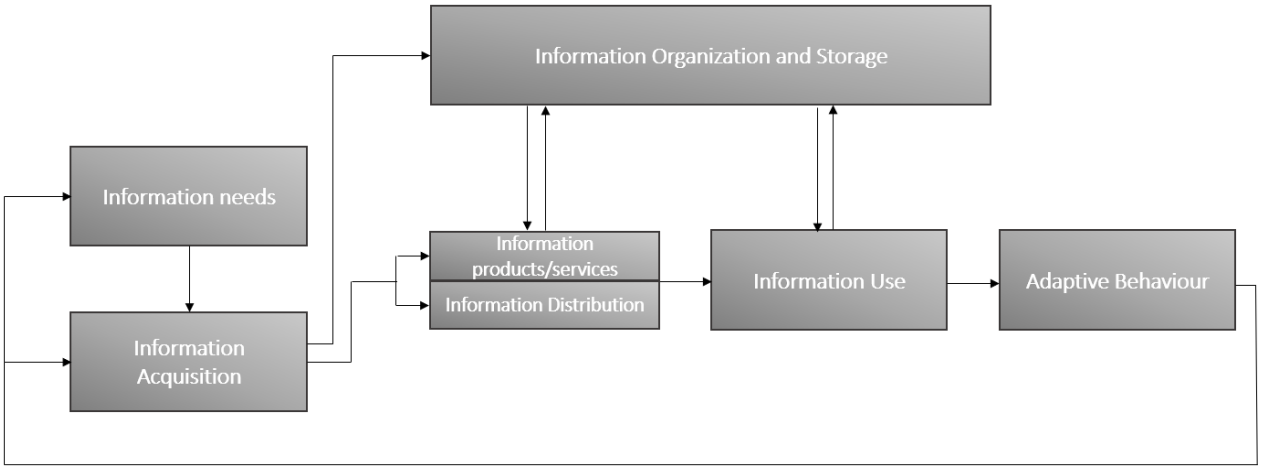

Figure 7: *Information management cycle (modified Choo, 2002)*

The model illustrates how information management is a continuous process. The process starts with identification of information needs. After information needs are defined, the information is acquired and stored. Storing information gives the opportunity to develop products or services, which are then distributed and, ultimately, used. (Choo 2002)

One of the most neglected processes of information management is needs analysis, although the quality of information is quite dependent on how well needs have been communicated. Furthermore, the use of information is crucial, because continuous improvement of other information management processes requires an understanding how information is used or not used to make decisions, solve problems, or interpret situations. (Choo 2002, p.25)

Information needs emerge from problems or ambiguities encountered in organizational situations and experiences. To uncover information needs, some questions need to be asked, such as: “What do you want to know?”, “Why do you need to know it?”, “What do you know already?” and “What do you anticipate finding?”. Information needs are contingent, dynamic, and multifaceted. Thus, a rich representation of the information-use environment can help complete the specification at a sufficient level. (Choo 2002, p.26)

3.2.2 Organizational silos

Organizational structures are prone to create organizational silos that can cause information and knowledge to flow in vertical silos instead of horizontal. Silos are not always physical since they can exist in the minds and thoughts of personnel who have a similar impression of reality. The barries caused by silos can create an "us-them" mentality, thus, breaking the boundary is challenging. Silos in organizations focus on the fulfilment of the function and achieving their own objectives, instead of achieving a process outcome. (Forsten-Astikainen et al. 2017)

To gain wider insight, silos can be broken down to gain efficiency and improve performance. Informal communities, also known as communities of practice (CoP), can be created based on the base of a shared concern, problem or passion, the silo syndrome can be mitigated. The internal communities are not bound by organizational affiliations and are mostly self-organizing, but can benefit from support received by an organization. The participants in such communities can tackle tasks collaboratively by transferring knowledge, competences and experiences. However, if CoPs are created inside the silos, the distinction between functions may be more clear-cut. CoPs that are created across organizational and functional boundaries may mitigate the unfavorable silo effects. Managers can help identify and support the communities, in addition exploit the expertise and knowledge. (Forsten-Astikainen et al. 2017)

3.2.3 Business intelligence

Making decisions solely based on intuition, personal experience and “gut feeling” has been typical for commercial enterprises in history. Such decisions can be classified as biased and unreliable. Instead, data can be utilized to provide guidance in decision making. One core idea in data-driven decision making is the deduction of decisions based on key data sets. (Matthies and Hesse 2019) The process of business intelligence (BI) comprises the journey of data: from the source of collection to developing it into knowledge, which is taken as actionable insight to make a decision (Najdawi and Patkuri 2021). According to Laihonen et al. (2013), business intelligence refers to organizations acquiring, analyzing, sharing, and utilizing business information that is significant for their business. The purpose of business intelligence is to acquire relevant information from different sources, then categorize and store it

for later usage. The main purpose is to analyze and combine information that were not connected beforehand to create insights and share information to decisionmakers that need such information. (Laihonen et al. 2013, p. 45)

Figure 8 below depicts the business intelligence process.

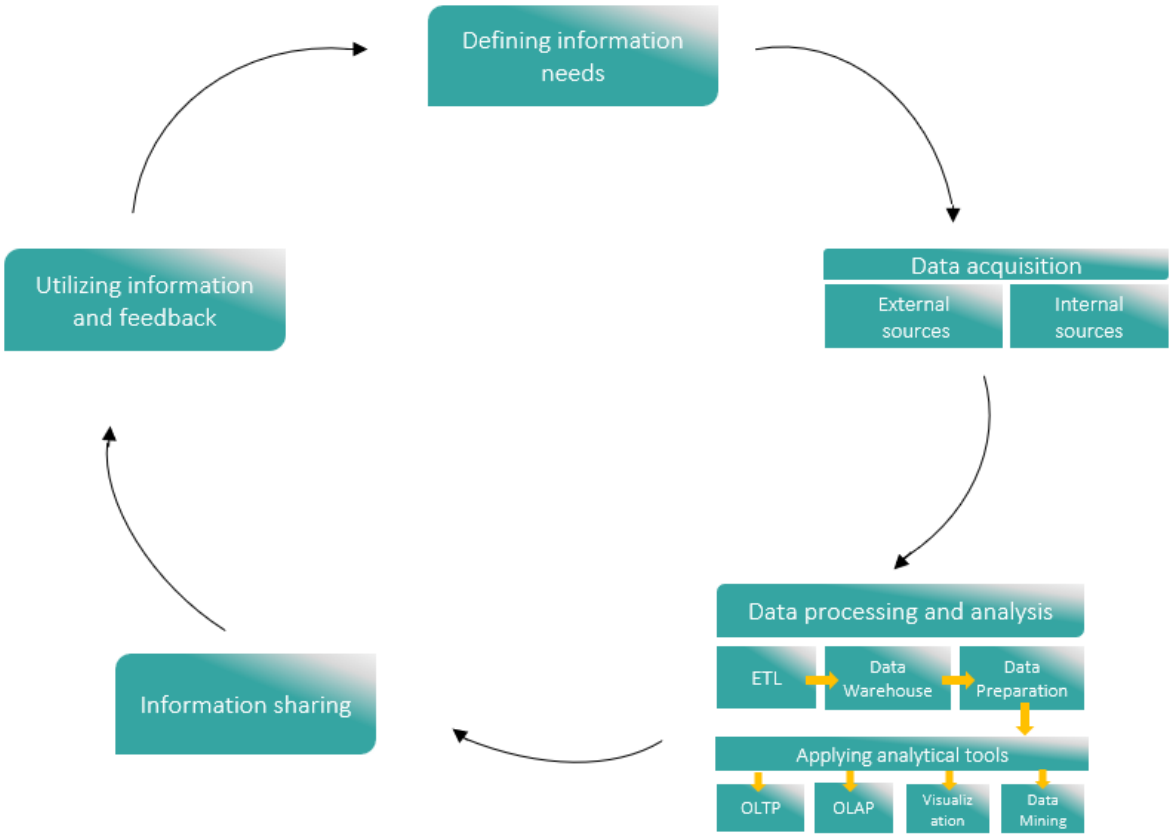

Figure 8: *Business intelligence process (modified Laihonen et al. 2013, p.46, Najdawi & Patkuri 2021)*

The business intelligence process starts with defining information needs for decision making. The intent is to define what information is needed, when is it needed and how it could be utilized. When information needs are defined well, information acquisition can be targeted better and redundant information can be avoided. Many factors affect information needs, such as the organization's industry and strategy, changes in business environment and uncertainty in decision-making. Organization-level information needs are formed by individual decision makers' needs. The constantly changing business environment of an organization also changes the information needs, therefore information needs are defined continuously throughout the process. (Laihonen et al. 2013, p.47)

The second phase is about acquiring information from different sources. Typically, explicit information sources are widely used, such as database reports or consultant analyses, because acquiring them requires less work. However, less accessible sources and humane sources are also seen valuable. From large masses, finding reliable and correct information

can be difficult. Not only is acquiring information difficult, but also the ethical and legal perspective should be taken into consideration. (Laihonen et al. 2013, p.47)

Acquired information usually needs to be processed in order for it to be properly utilized in decision making. Processing involves clearing, reviewing, and categorizing information to answer needs as well as possible. Information can be stored in information systems, from where it can be found, shared, and utilized. Some information also is transferred as intangible capital by people who use the information. The acquired information is analyzed and connected to earlier information to give it a purpose in the organization's context. Some examples of tools that can be utilized for this purpose are scenario planning, statistical analyses, and information visualization applications. (Laihonen et al. 2013, p.48)

The meaning and benefit of information use is better understood when information is refined to become an information product. For example, market region specific monthly reports or competitor analyses are common information products. One precondition for information utilization is to receive information at the right time and in a usable format. Information can be distributed through different channels, like information systems or e-mail, but the channel does not define the value of information. The last phase determines the value of the process. If the information product is not utilized, it will not generate value. Information can either confirm current perception of a situation or bring up a new perspective. To create value to the organization, it must affect the organizations activities and should support the organization's processes, problematic situations, and daily routines. (Laihonen et al. 2013, p.48)

3.2.4 Decision-making in the software industry

Decision types in software business can be divided into three domains: strategic, tactical, and operational decisions. Strategic decisions relate to strategic planning, which defines the direction and goals of the company in the long haul. Strategic decisions concern the whole business, not necessarily individual business units or teams. In Agile development, strategic decisions are often assigned to Product Owners that need knowledge of the product in detail, and they are mainly concerned with software release plans and product road maps. As development teams have deep technical knowledge, they should also be included into defining the strategy. Tactical decisions relate to the management of projects and the translation of strategic decisions into actionable plans for different organization areas, meaning the structurization of teams and resources to achieve goals given. Operational planning relates to decisions that are on the most concrete level. Such decisions may define milestones, success conditions and what portions of projects and proposals are dealt with. In agile development, operational decisions involve the implementation of features and making sure tasks are performed efficiently. (Matthies and Hesse 2019)

Software engineering requires creativity, problem-solving capabilities and empathy for users and stakeholders for whom the software is developed. Therefore, software engineering is ultimately a human task. Important decisions in projects should neither rely only on intuition or data. Rather, project data should be collected, analyzed, and then interpreted (or possibly even ignored if irrelevant) by team members and practitioners. Having supporting data allows to developers, product managers, sales departments and other roles related to the context make more *informed* decisions in software development. In agile development, the approach of making shared decisions includes stakeholders and development teams. Prioritizing features and work items falls into the hands of the development team and stakeholders, which have different goals and backgrounds. (Matthies and Hesse 2019)

Lean software development framework has seven principles that can be incorporated into agile development. The seven principles are:

- Eliminate waste
- Amplify learning
- Decide as late as possible
- Deliver as fast as possible
- Empower the team
- Build integrity in (build quality in)
- See the whole

In Lean thinking, increasing productivity and efficiency is key. Anything that is not adding value to customer is thought to be waste. However, waste needs to be recognized. For example, features that are not used, documentation that is not read, errors that have occurred and incomplete work are included. Both Lean and Agile have common goals: improving software quality, improving the productivity of development process, implementing changes in requirements quickly and customer satisfaction. (Baumgartner et al. 2021, p.35-36)

3.3 Data analytics capabilities

Making sense of data is essential in today's organizations. Namvar et al. (2021) explained that there is an increasing demand for the right skills that make data understandable. Business analytics, or simply analytics, is described as the use of data to make more informed and fact-based decisions in business. Organizational data is generated both from internal and external sources. (Namvar et al. 2021)

There are different types of data analytics defined in literature. Raghupathi & Raghupathi (2021) explained that in the past, descriptive analytics has been the most applied analytics

type in businesses. Descriptive analytics describes data without additional, complex calculations. The aim is to understand decisions made in the past and currently by data. Therefore, business decisions can be made based on data, i.e., be more data-driven. Descriptive analytics converts data into useful information for analyzing business decisions and outcomes by categorizing, characterizing, aggregating, and classifying data. Visualization is deployed to a great extent in descriptive analytics. Data warehouses help disparate data to be merged and produce dashboards, customer views and repositories. (Raghupathi & Raghupathi 2021)

Predictive analytics is a more advanced type of analytics where the use of information, instead of data, is emphasized. Future performance is predicted, for example, by exploring past business data, by detecting relationships or patterns in the data and extrapolating the patterns to produce an informed forecast. The benefit of predictive analytics is having the ability to anticipate risk and identify relationships that possibly are not apparent with descriptive analytics. (Raghupathi & Raghupathi 2021)

Prescriptive analytics utilizes not only data and information, but also business knowledge. The aim is to follow up recommended courses of actions for previously predicted outcomes. Prescriptive analytics typically comes into play when too many alternatives and choices are involved, so that the two other analytics types cannot be considered. The question prescriptive analytics addresses is "*what should be"*. (Raghupathi & Raghupathi 2021) The future of decision-making has been shaped by big data, predictive analytics and forecasting techniques. (Najdawi & Patkuri 2021)

The fourth analytics type, discovery analytics, refers to the utilization of wisdom or meta-knowledge in the identification of new products or services. Discovery analytics is seen to be the next frontier in analytics, helping businesses identify new markets, products and strategies. (Raghupathi & Raghupathi 2021) The data analytics types are summarized in Figure 9.

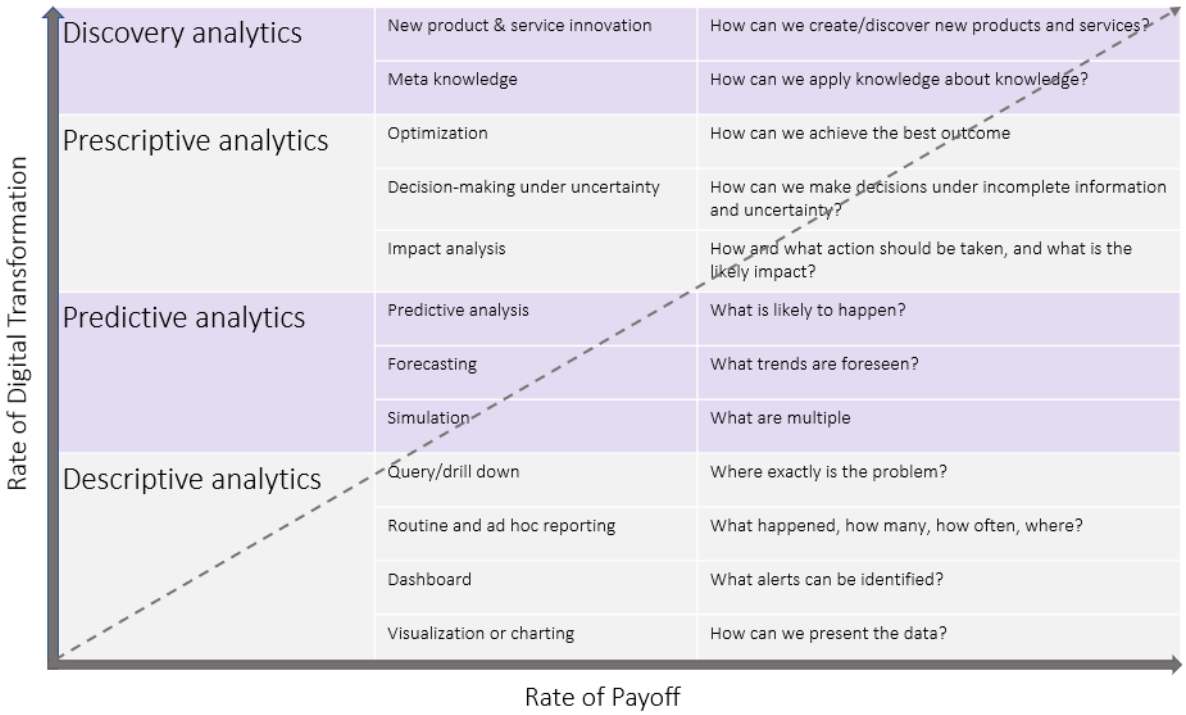

Figure 9: *Data analytics types (modified Raghupathi & Raghupathi 2021)*

There are three pillars of analytics that inform business modelling: visualization, statistical modeling, and machine learning. For descriptive analytics, visualization is the most important pillar. Different variables, dimensions, correlations, and insights can be visualized with data. A variety of charts can reveal characteristics or describe data. Altogether different charts can tell a story about the selected data, the problem analyzed, the insight that is found and the potential implications for practices. Visualization is enabled by different tools, like BI tools. (Raghupathi & Raghupathi 2021)

Statistical modelling relates to mathematical modeling and representation of the data, where the method works with a set of assumptions. Machine learning, a subset of artificial intelligence, is about applying mathematical modeling to data. Machine learning involves three steps in modeling: training, validation, and testing of the data. The method is typically appropriate for unstructured data like text, audio and video. (Raghupathi & Raghupathi 2021)

3.3.1 Data analytics in software development

Data analytics can also be applied to agile software development. According to Biesialska et al. (2020), big data analytics can be applied to the whole agile software development lifecycle. Organizations tend to adopt big data analytics in different degrees depending on their level of technological maturity. Often additional functionalities are integrated to existing tools or solutions. Biesialska et al. (2020) found in their paper that in agile software development, software testing was one of the major applications that applied analytics, covering all four analytics types. Software quality assurance was found to be one focus area in data-driven

agile software development, regarding defect/bug fixing, testing and code maintenance. (Biesialska et al. 2020)

The aim of analytics is to support decision makers in extracting important information and insights from datasets that otherwise is not be visible. Moreover, the goal is to move from gathering information to gaining insight, although it is not easy. Questions of information that tools can already provide, e.g., how many bugs are open, differ from questions of insight which give managers a better understanding of the project's dynamics and what to base decisions on. Insight requires knowledge of the domain while it also needs the ability to identify patterns, which include multiple indicators. For example, if a bug is open, most likely some other activities are consuming resources needed for the bug fixing. (Buse & Zimmermann 2010) Although Buse and Zimmermann had researched software development in the project context, this can also be applied to continuous product development as well since e.g., new features are developed in a project-like manner.

Based on literature, there is a disconnection for project managers to make decisions with the information provided by current tools. When tools are either unavailable, too difficult to use, hard to interpret or do not give useful information, managers have to rely on past experience or intuition for making critical decisions. (Buse & Zimmermann 2010)

According to literature about the mobile industry, delivering agile software ensures better time to market with frequent application releases. Customer feedback coming from different channels ensures quick adaptation. Test optimization can be achieved through capturing insights from different data channels. Application monitoring tools are one data channel that provide insights on application usage, crashes, and errors, but also device usage and crashes. Defects found from production provide some insight on where to allocate testing efforts. However, data insights on feature usage, popularity and crashes can give guidance in test optimization. (Singi et al. 2017)

System usage typically refers to different aspects like total time, frequency and number of features used and so on. Mostly such usage measuring techniques concentrate on the comprehensive system usage, not necessarily on a single feature level. There are two approaches to monitor the system usage of certain features: extending the code of software that is responsible for monitoring or designing an application that catch triggered events from the usage of the system. These approaches do include some challenges as increasing complexity of software if adding such extensions and also possible violations of privacy laws when collecting data in a hidden manner. (Marciuska et al. 2013)

3.4 Collecting customer feedback in R&D

Collecting customer feedback and customer data efficiently is difficult. It is called an 'open loop' problem where it is hard for product management to access customer feedback that

could help, for example, in the process of prioritizing features. Although significant data is collected, there is little knowledge how their products are used and what features are appreciated and used by customers. One way to gather such information is to perform prototype testing, where feedback is collected to support developers continuously. Another way is to do A/B testing – a way to optimize a feature that already exists, a new feature or when building a new product. (Fabijan et al. 2015)

In product management, it is typical to make product decisions and requirement prioritization based on opinions and intuition than validated customer feedback data. As a result, resources are spent on development that does not fulfil customer needs or expectations. In some cases, the feedback loop from customers is quite slow and complex. Product management works as an interface between customers and development. In addition, getting feedback in a timely manner also has its own challenges. Companies do not have mechanisms to systematically collect, analyze and include feedback from customers during the development process. Verification is also done post-deployment. To improve decisions made in product management and development investments, companies need to improve data-driven practices. (Sauvola et al. 2015)

According to Ahmad et al. (2021), there are benefits in getting a better understanding of information needs among software practitioners. Such benefits are maintaining competitiveness, becoming aware of issues that hinder releasing in time and being able to build a visualization tool to address software practitioner's information needs. (Ahmad et al. 2021)

The intuition of software development companies can be off even up to 90% regarding customer preferences. To avoid such problems, getting data on how the product is used can help the prioritization process to become more accurate since it focuses on what customers do, not what they say. Controlled experimenting helps evaluate ideas with customers reliably, thus helping prioritize product development activities correctly. (Fabijan et al. 2017)

3.5 Customer-centric software development

Figure 10 presents a customer touchpoint (CTP) model by Sauvola et al. (2015), which was formulated during a multi case study of five Finnish software companies. (Sauvola et al. 2015)

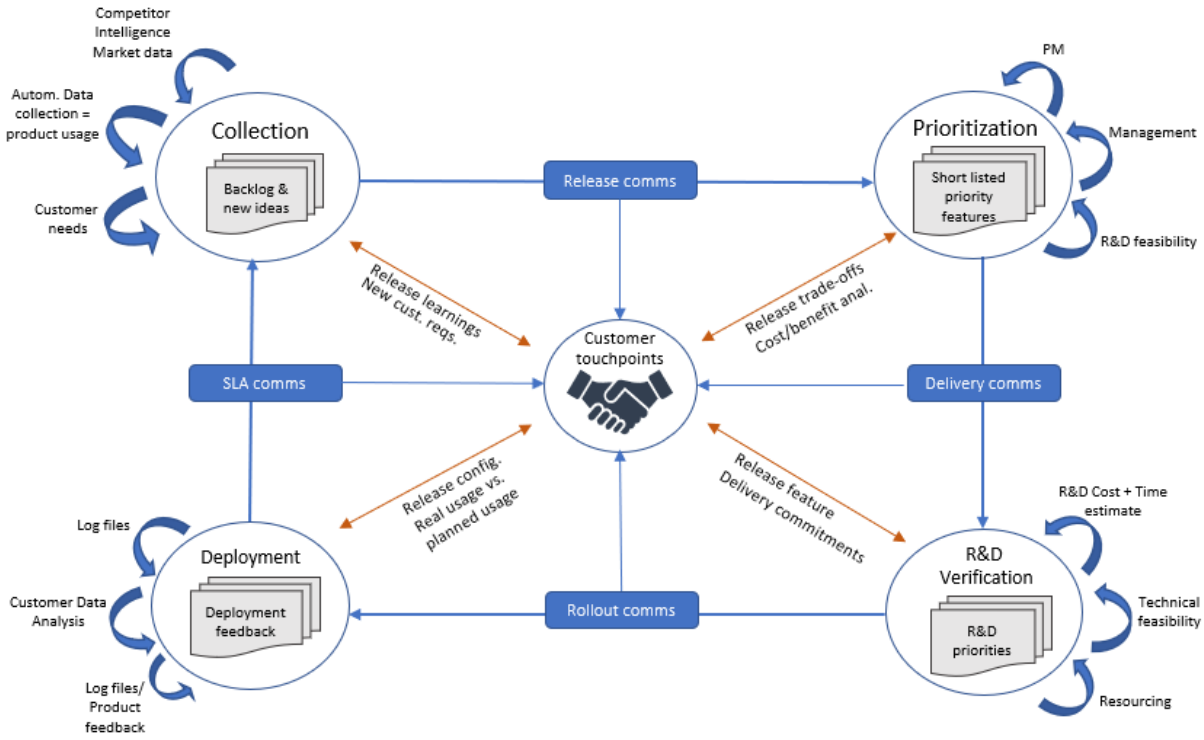

Figure 10: *Customer touchpoint model (modified Sauvola et al. 2015)*

The model in Figure 10 gives an overall understanding on the most important customer touchpoints and collection of qualitative and quantitative data from customers, markets, and product usage. The four activities in this model are: collection, prioritization, R&D verification, and deployment. Customer touchpoints inside activities refer to points where information is exchanged between companies and customers. (Sauvola et al. 2015)

3.6 Test intelligence

The development of software systems has changed on a fundamental level, from releasing once or twice a year to releasing within months, weeks or even days. Feature-driven releases require testers to test in each feature branch, and on the overall system level. The solution for the fast-growing testing demand has been to invest in test automation and CI. However, the automation running on large systems can take several hours or possibly days. The time between creating the mistake and finding the mistake in the code increases. Although test automation improves the test execution to be more efficient, it necessarily does not make testing effective. To make testing more *effective*, testing efforts should be aligned with code changes. (Goericke 2020, p.1-2)

Testers need to continuously make decisions regarding which tests to run, find out what else is needed to test and whether the test suite contains redundant tests. Searching for such information manually can be difficult, but some information can be provided through the software development process. Such approach is like Business Intelligence which aims to gain

insights on business processes by analyzing readily available data. Like Business Intelligence, the term Test Intelligence can be used, which aims to answer questions concerning both tests and code changes under test. Much of the data is available in the development environment. (Goericke 2020, p.2-3) Figure 11 depicts what data sources are relevant for testers and what questions testers are interested in:

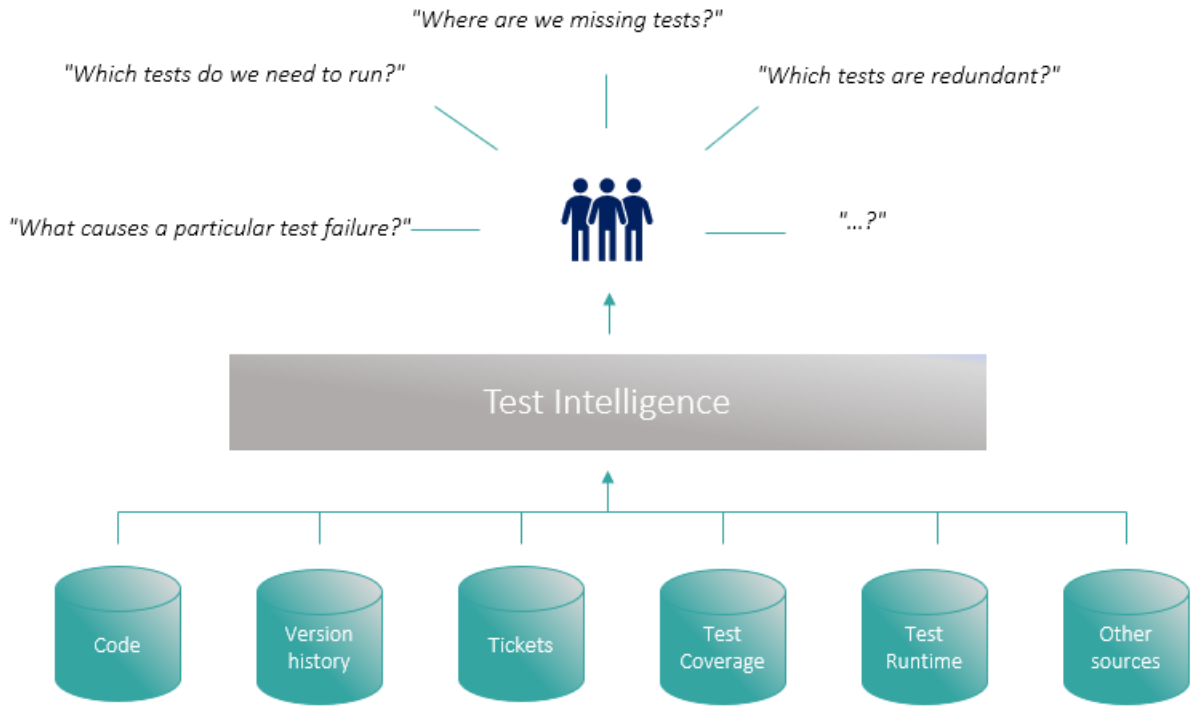

Figure 11: *Test intelligence (modified Goericke 2020, p. 3)*

In software development, version control systems (VCS) are used to keep chronological history of code changes. VCS guide developers as they can keep track of code history which helps in coordinating changes. In addition to VCS, ticket systems are used to follow the track of change requests like bug reports., feature requests or maintenance tasks. These systems help manage the requests as tickets which have unique IDs, assignees, statuses, and other metadata. Typically, developers annotate the ticket ID in the VCS to justify the reason for making changes. (Goericke 2020, p. 3-4)

The execution of application code can be tracked through profilers. Profilers assist in understanding which methods, statements or even branches are executed during program execution. Profilers can track coverage at different granularity. However, the more granular the profiling the higher the performance penalty. When profiler's coverage target is a test, it is called test coverage. Test coverage recording is a largely used practice for automated tests and CI. The coverage of a test suite can be visualized on a treemap. For example, highlighting executed methods in green and non-executed methods in gray can give an overview what has been run. (Goericke 2020, p. 5)

3.6.1 Change-driven testing

Change-driven testing refers to a technique where testing efforts are aligned to changes. When changes are increasing, executing all tests after every change has become infeasible, and also impractical to manually check if all changes are tested sufficiently. The idea behind the method is that existing tests fail only when new mistakes are introduced, which are created through changes. (Goericke 2020 p.2-5) Figure 12 illustrates the idea behind change-driven testing.

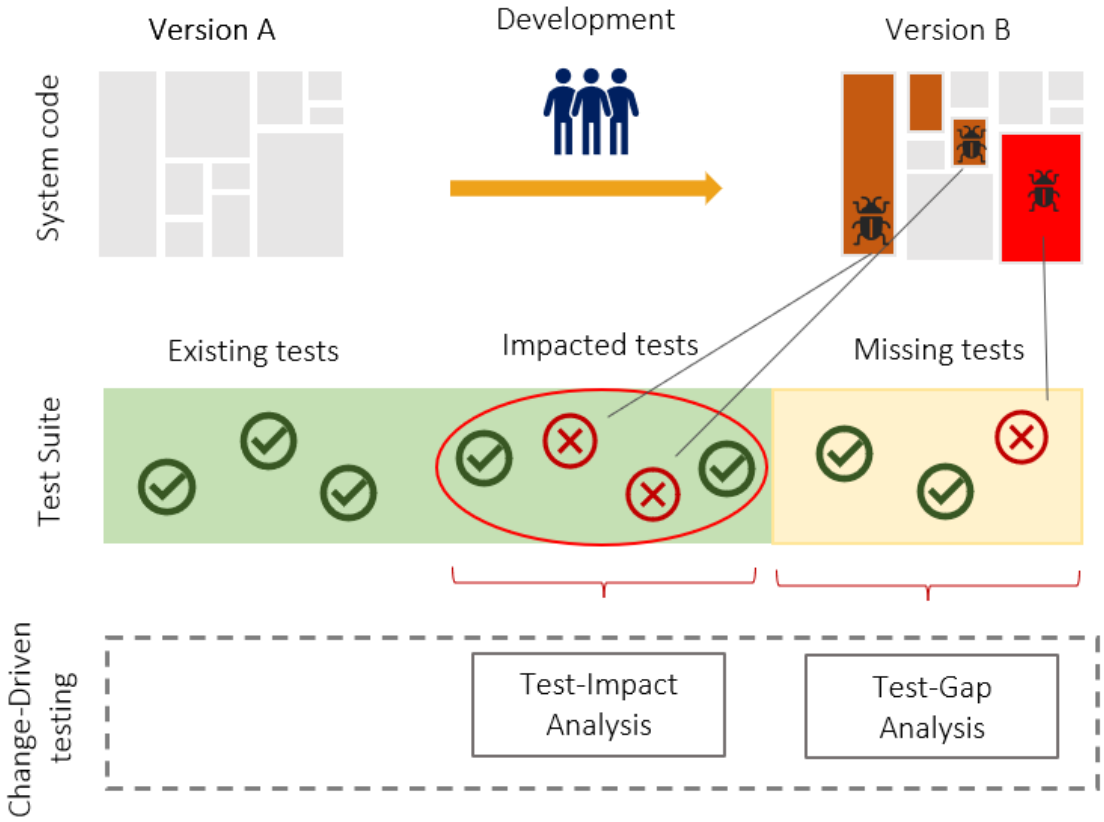

Figure 12: *Change-driven testing (modified Goericke 2020, p.6)*

To test the changes, there are two phases of analysis:

1. Test efficiency can be achieved by testing only the changes made and excluding all other system areas. The impacted tests can be identified automatically through Test-Impact Analysis (TIA).

2. Test effectiveness can be achieved by ensuring all changes are tested. To guide testing towards that goal and verify whether it was achieved, the changes that lack tests can be automatically identified through Test-Gap Analysis (TGA). (Goericke 2020 p.6)

Figure 13 illustrates the Test-Impact Analysis process.

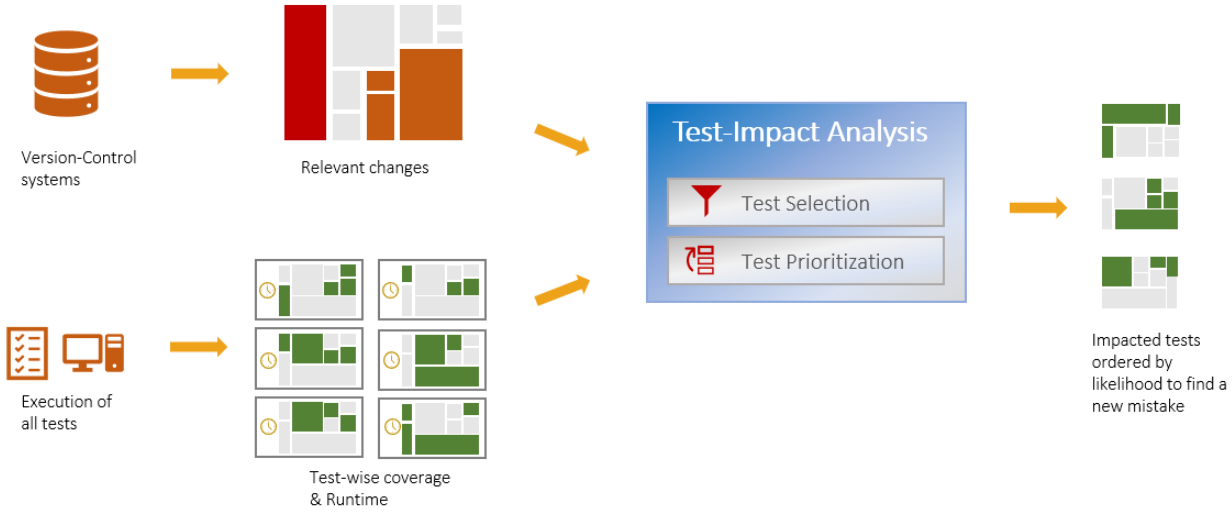

Figure 13: *Test-Impact Analysis process (modified Goericke 2020, p. 7)*

Test-Impact Analysis combines relevant code changes, test runtimes and test-wise coverage to compute a subset of the test suite, where mistakes are identified as early as possible. The analysis has two phases, test selection and test prioritization. Test selection only selects the impacted tests that execute any changed methods. In the test prioritization phase, the impacted tests are ordered in a way that all changes are covered as fast and early as possible. One way to order tests is to select the tests based on which covers the most additional changes of code per execution time. One study of 12 software systems had found 99.3% of randomly put mistakes by selecting impacted tests with TIA. Another study including 100 systems had found TIA identifying on average over 90% of mistakes in a test suite compared to the 2% of execution time in the test suite. TIA is seen useful in small and local changes, where the test selection involves only a small set of tests. (Goericke 2020 p.6-7) Figure 14 illustrates the Test-Gap Analysis process.

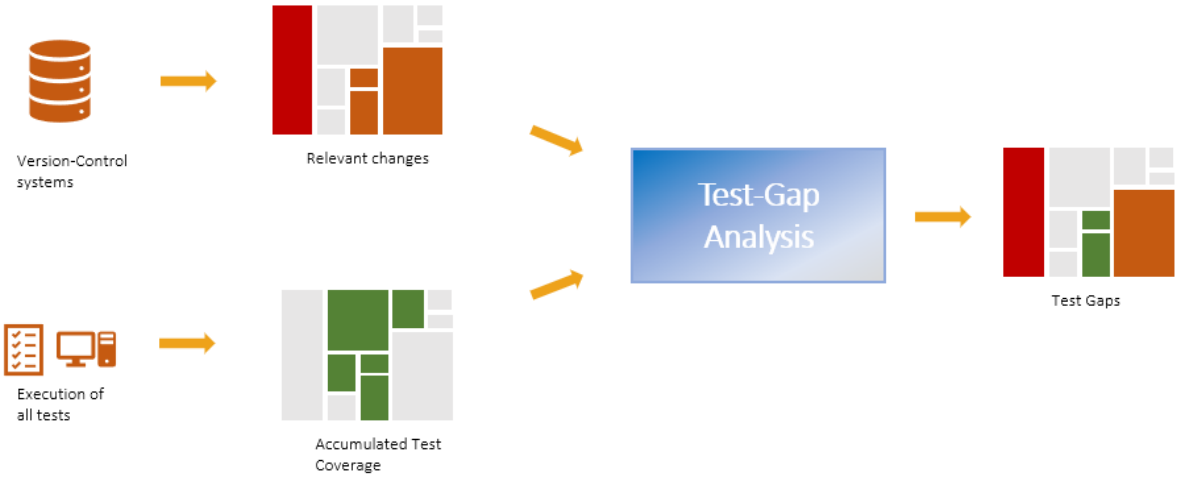

Figure 14: *Test-Gap Analysis process (modified Goericke 2020, p.8)*

Test-Gap Analysis matches the relevant changes of code with the aggregated test coverage to pinpoint changes that are not covered by tests. The changes are called test gaps. In TGA, the test gaps are determined by the chronological order of changes and test runs. Moreover, new changes invalidate the preceding test coverage of changed code and open new test gaps. Testing done afterwards will record new coverage, thus respective test gaps from earlier changes are closed. In one case study of a large industrial software system, TGA found 55% of untested code changes in two consecutive releases, where 70% reported field bugs were traced to untested code. One study of another industrial software system identified 110 test gaps through TGA, where 35% of the gaps were worthy of testing. In addition, 49% of the gaps were the result of cleanup work that were not part of the ticket description. Hence, the changes remained untested although the changes had been tested based on the ticket. (Goericke 2020 p.7-8)

3.6.2 Change-driven testing limitations

The change-driven testing analyses has certain limitations to be aware of. One limitation is the changes on the configuration or data level, which remain hidden as the changes are not reflected in the code. Another limitation of TIA is when indirect calls are used. The coverage of test cases is assumed to be stable in TIA. The coverage of automated tests may differ between runs, for instance, because of garbage collection, and manual test steps may be executed differently with each execution. Thus, it is recommended to execute the whole test suite occasionally to make sure nothing was left unnoticed. (Goericke 2020 p.8-9)

Another limitation is that TGA does not examine the depth a change is tested. TGA only considers if the methods, that include the change, are executed in a test. Like always, testing cannot confirm the absence of mistakes in code, which also applies to this testing approach. (Goericke 2020 p.8-9)

4. RESEARCH METHODOLOGY

This chapter describes how the research is conducted. In addition, the case company and the context are presented. Moreover, this thesis uses a case study research strategy to analyze QA activities, information management practices and data analytics opportunities in the case company, supported by theory on these topics. The characteristics for the research design are described.

4.1 Context of a software company: case M-Files

This study examines the case company M-Files Oy, which is a Finnish company offering intelligent information management solutions for different industries. In 2020, M-Files completed the transition to a SaaS- business and delivery model. The primary SaaS- solution, M-Files Online, is a hybrid intelligent information management platform where both cloud and on-premises solutions can be implemented, all with one license. (M-Files 2021) Figure 15 illustrates the organization structure and highlights the organization areas involved in this thesis.

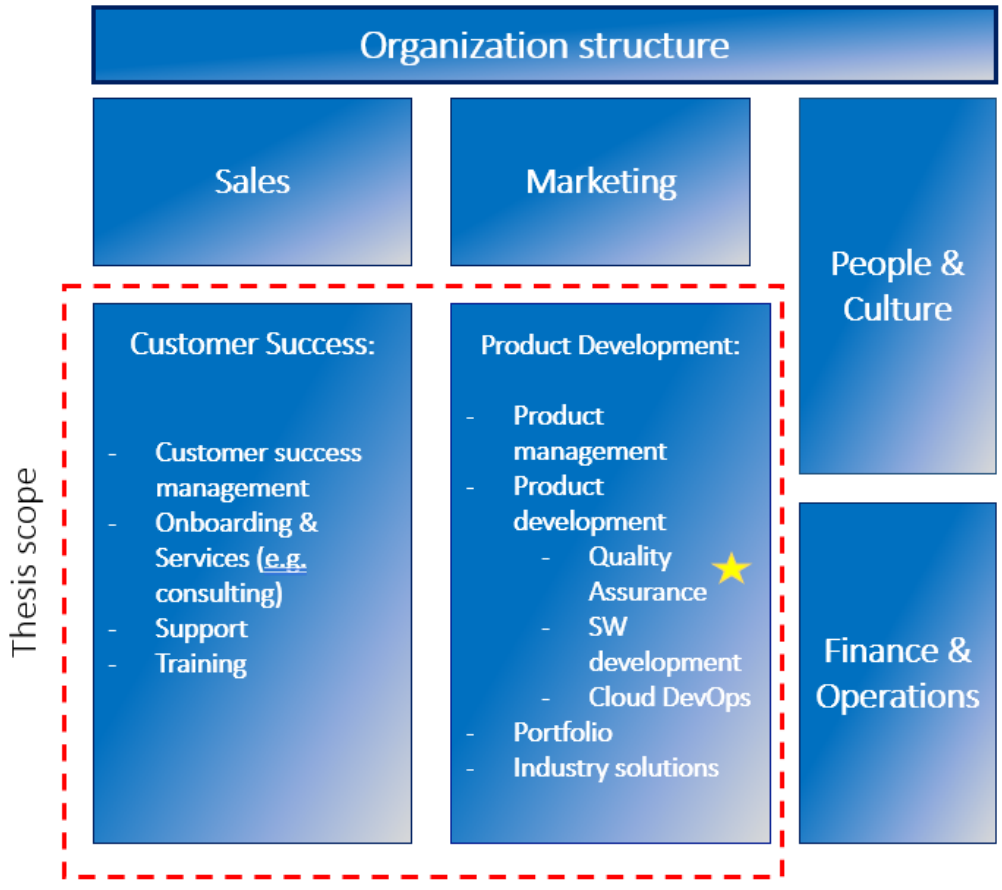

Figure 15: *Case company organization structure*

The case company is researched by interviewing employees within the product development organization and customer success organization. This thesis focuses on the product development organization, especially the QA team, of the company. The company's mission is to improve how customers do business in the digital, work-from anywhere world. To align product development with the company's mission, the product organization's strategy has shifted focus to the development of current processes, ways of working and mindset towards more customer oriented, data-driven and outcome focused product development.

M-Files product development is an ongoing process, where the product is developed continuously by adding new features and fixing possible bugs on a regular basis. With the help of continuous integration and DevOps, the M-Files product is nowadays released to customers every month, thus new features and patches are available more consistently. The workload of development and testing is done within short, biweekly sprint cycles to ensure and manage progress in development. The development of M-Files features is split into smaller user stories that each developer works on during sprints.

Quality assurance has four phases at M-Files that are seen in Figure 16.

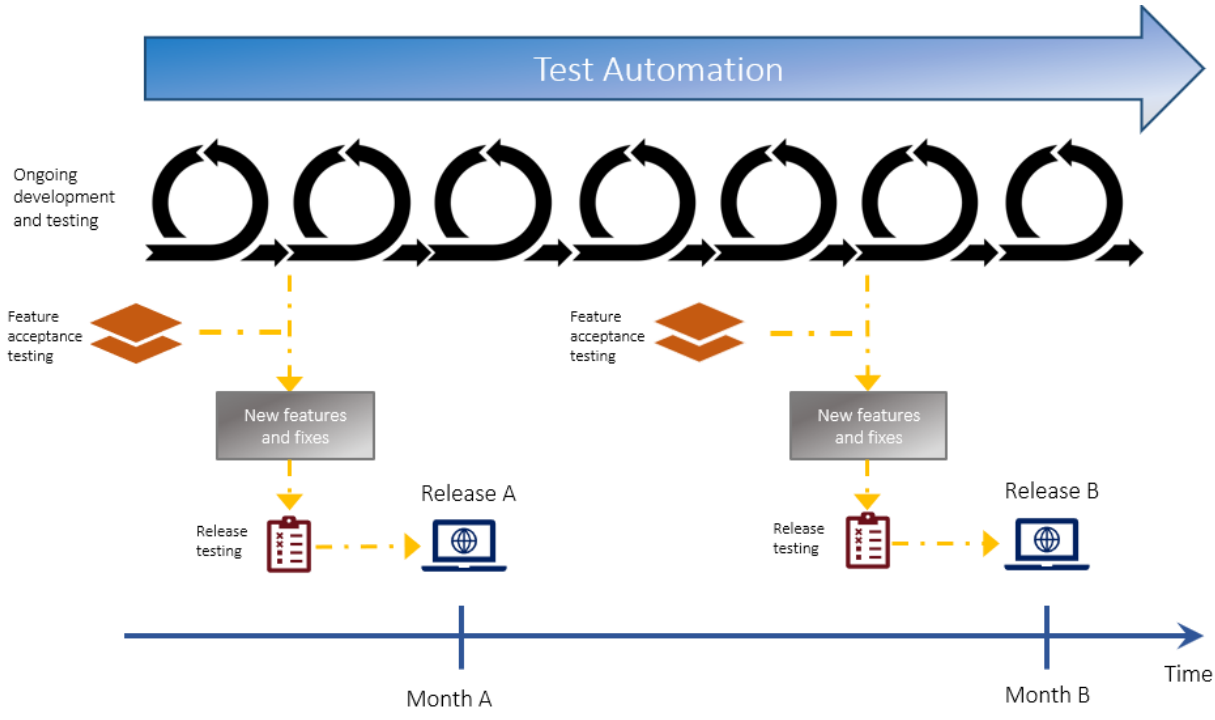

Figure 16: *Main QA activities in case company*

Quality assurance is responsible for testing developed user stories and providing feedback on the increment. At the end of each sprint, ready user stories are reviewed based on Definition of Done (DoD) checklist which is one major quality assurance control point. When a set of user stories form a ready feature, QA also test the whole feature, which is called feature acceptance testing. If feature acceptance testing confirms the feature is ready from quality

perspective, the new feature can be included in the main product. Before each monthly release, QA performs release acceptance testing, that includes system testing of the core product and new features. QA also performs test automation, which includes API testing and functional UI testing. Test automation phases include automated developer tests, automated integration tests and automated regression tests. Furthermore, testing activities include non-functional testing such as performance testing and security testing. QA also has some additional testing tasks, such as M-Files add-on compatibility testing, that refer to checking if extensions with other service providers work together with M-Files product.

At the time of the research, the quality assurance team had approximately 30 employees. Every function in the organization follows an Entrepreneurial Operating System (EOS) framework, including QA. The framework helps a business to set goals on different levels of the company, thus having everyone working towards the same goals. (Entrepreneurial Operating System 2022) Testers have two main tools to use, one includes test requests with other testing materials like documentation and information that help with building test environments. The other tool is an issue reporting tool, where bug reports are created, and implementation improvement suggestions can be added.

4.2 Research strategy: Case study research

This thesis follows a qualitative research design. Qualitative research design refers to a data collection technique where data is non-numerical (Saunders et al. 2019 p.175). Qualitative research is typically linked to an interpretive philosophy, where researchers must understand the subjective and socially constructed meanings indicated about the target phenomenon. Moreover, qualitative research is about studying the meanings and relationships of the participants. Meanings are often extracted from words and images, instead of numbers. Typically, qualitative research is related to an inductive approach to theory development, where theory is formed based on the collected and analyzed data. (Saunders et al. 2019 p.179) Other characteristics of qualitative research involve the researcher not being seen as independent from the research subjects. During the research subject selection, the sampling technique is often non-probability based. (Saunders et al. 2019 p.180)

This thesis follows a single case study research strategy. Yin (2018) defined a case study as follows: "*an in-depth inquiry into a topic or phenomenon within its real-life setting*". For example, a case may relate to an individual, a group, an organization. The chosen case also needs to have determined boundaries to form a defined case study. Case study research aims to investigate the dynamics of the topic within its context. The benefit of case study research strategy is the ability to generate insights from in-depth study in the real-life context. Such insights may help create rich and empirical descriptions, and the advancement of theory. (Saunders et al. 2019 p. 196-197)

Case study research strategy has also been under criticism due to seen views of not being able to generate generalizable and reliable findings and contributions to knowledge. However, the quality of research design can be considered through different criterion, e.g. validity and reliability. (Saunders et al. 2019 p.197)

4.3 Semi-structured interviews

To get a thorough understanding about current QA activities in the case company, it was decided to use semi-structured interviews to extract knowledge from employees. The data collection technique of semi-structured interviews has a combination of both closed- and open-ended questions, which are often followed by *why* and *how* questions (Adams 2015). When planning the semi-structured interviews, the work of Adams in "Conducting Semi-Structured Interviews" was utilized. Adams (2015) recommended using semi-structured interviews in a situation, where you need to ask probing questions and want to know independent opinions and thoughts of each individual in some group. (Adams 2015) Saunders et al. (2019) explained that semi-structured interviews not only help the "what" and "how" questions, but also "why". (Saunders et al. 2019, p.442) Understanding the reasons how QA organizes their activities, and why they are organized like they are currently, is important for this thesis. Therefore, semi-structured interviews were found to be the most optimal way to collect data.

Semi-structured interviews are designed based on the research questions set. The semi-structured interviews have two main themes: current state of QA activities and current understanding of customers from QA point-of-view. Based on these themes, a set of questions were created beforehand. The first theme aimed to recognize what type of activities QA has currently and how they are organized. In addition, the aim was to understand what activities need to be improved and what type of information needs relate to them. The second theme maps out the current understanding of customers from the QA perspective. Moreover, the idea is to find out how well QA understands who their customers are, what needs do they have and how the M-Files product is used by customers. Through these two themes, the QA can identify what information is needed and what actions should be taken to become more customer centric and data-driven.

Interviews were held as one-to-one interviews and one-to-many interviews, i.e., group interviews. According to Saunders et al. (2019), group interviews have the advantage of allowing broad point of views to emerge, and interviewees can respond to the differing views. Group interviews are also more dynamic, where ideas can be generated and evaluated. Thus, concepts are easier to be explored and explained. In addition, group interviews provide an efficient way to interview a large group of individuals (Saunders et al. 2019, p. 470) In each

group interview, there are two participants simultaneously. Defining information needs requires domain knowledge and experience, therefore the exploration and bouncing of thoughts back and forth was desired. As the focus of this thesis is the QA team, it was decided to split interviews into pairs to attain efficiency but also gain enough material that is within relevant extent for a thesis. The interviews are mainly held as face-to-face interviews but also through video call due to ongoing pandemic times. Face-to-face interviews have an advantage of building rapport while simultaneously alleviating concerns with data transfer with the researcher. (Saunders et al. 2019, p.442).

4.3.1 Interview design

The idea is to design the interviews to start from general questions, gradually proceeding to specific questions. The goals of the interviews are:

- Understand what challenges QA has currently
- What internal stakeholders are responsible for and how their activities affect QA
- How customer-facing, internal stakeholders view QA performance and overall product quality
- What ways can customer facing stakeholders and QA improve customer-centric activities

Gaining a sufficient understanding of the context requires gathering empirical evidence about QA, who QA works with, and which internal QA stakeholders work in the customer interface. In addition, gathering information about analytics capabilities in the case company is also relevant. The first interview target group is the QA team that consists of over 20 team members and four managers. The second target group are internal stakeholders that work together with QA team. The sampling for the semi-structured interviews is selected based on the researcher's understanding of which employees have knowledge about the topic. The sample intentionally includes employees who have longer experience working in the company but also newer employees, to get as wide perspective as possible.

As seen from the organization chart in Figure 15, QA has internal stakeholders both within the product development organization and the customer success organization. From the product development organization, the aim is to gather empirical evidence by interviewing developers, product management, and the leadership of product development are the. Based on the researcher's understanding, product management drives *what* is developed in product development organization; thus, it is relevant for QA to know where needs for development derives from, to be able to properly evaluate the level of development quality. Also, one product manager is more involved with analytics development at the case company. QA tests directly developers work, which is why developer understanding is needed. Leadership of

product development can open the higher-level expectations and objectives set for product quality. In the product organization, some developers have more direct contact with customers than other developers typically do. The interview sample from the product development organization is selected based on these reasonings.

From the customer success organization, the intention is to interview stakeholders who work with customers more directly. Within customer success organization, customer success management team works with existing customers, and consultants work with customers during the delivery phase. Although the interview sample from customer success is narrower, many stakeholders from the product development organization also work together with customers, therefore the customer interface view is represented sufficiently enough.

Interview questions are customized based on role. Figure 17 illustrates the main interview themes.

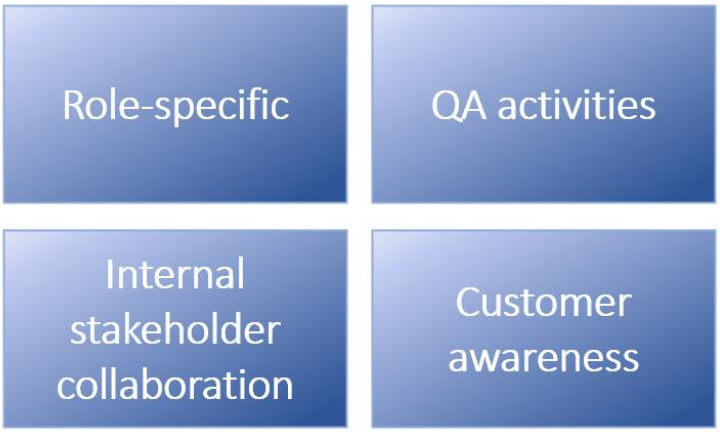

Figure 17: *Interview themes*

This research involves 11 interviews and 14 participants in total. Table 1 below concludes all participants taking part of the empirical evidence.

Identifier	Role	Time at case company (yrs)	QA Relationship	Interview duration
I1.	QA Engineer	0-5	In QA	55min
I2.	QA Engineer	0-5	In QA	45min
I3a.	QA Engineer	5-10	In QA	51min
I3b.	QA Engineer	5-10	In QA	51min
I4a.	QA Manager	5-10	In QA	1h 2min
I4b.	QA Manager	10-15	In QA	1h 2min
I5a.	QA Manager	5-10	In QA	1h 24min
I5b.	QA Manager	5-10	In QA	1h 24min
I6.	Product Manager	0-5	Customer interface	37min
I7.	Software Development Manager	10-15	Product development	59min
I8.	Customer Success Manager	10-15	Customer interface	39min
I9.	Product Manager	0-5	Customer interface	43min
I10.	Management of Product Development	0-5	Product development / Customer interface	29min
I11.	Consulting Manager	5-10	Customer interface	39min

Table 1: *Interview participants with details of current role*.

Each identifier in Table 1 represents one interview participant. Five interviews are targeted to the QA team and 7 to internal stakeholders. Some of the QA interviews are conducted as pair interviews (e.g., I3a and I3b) due to large team size and time constraints. As mentioned in previous chapter, group interviews were seen to help generate new thoughts, which fit well to this context. Some feedback was also gathered from the first interview, where pair interviews were seen to be a better option during answering questions. The rest of the interviews involved participants from the customer-facing functions. The participants are in roles working together with customers in different degrees. The work experience participants had at the case company varied a lot, longest being 12 years and shortest 8 months. The interviews included participants from different positions as well, from trainee positions up to management positions. The average duration of the interviews were 49 minutes. All participants had some type of connection to the QA team, either working together directly or indirectly.

The participants were contacted either through e-mail or directly asking face-to-face. The e-mails consisted of a brief introduction about the research topic and what the interview questions will relate to. Permission to record the session was also asked, so the researcher could revert and make additions to notes. Each interview started with a brief introduction about the topic and researcher before moving on to the actual, role-specific questions. The interview mainly followed a set of pre-defined questions, but the use of questions was determined on

the spot. In some interviews, a couple probing questions were asked to complement the answer. Part of the questions which did not bring value in terms of the research scope were removed afterwards.

4.4 Qualitative analysis

The qualitative data collected through interviews are analyzed with a thematic analysis method. Saunders et al. (2019) defined thematic analysis as a method that searches for themes or patterns that are found in the collected dataset. The data is coded to identify themes that relate to the defined research question. (Braun & Clarke 2006, s. 87; Saunders et al. 2019, p.651) For instance, a unit of data might refer to several words, a sentence, or a line of transcript (Saunders et al. 2019, p. 653).

Thematic analysis procedure is outlined by Saunders et al. (2019) into four steps:

1. Familiarize the data
2. Code the data
3. Search for themes and recognize relationships
4. Refine themes and test propositions. (2019, p. 652).

Familiarization of the data may start by transcribing interviews. With transcription, the researcher can generate summaries or self-memos that can help during analysis. After the researcher is familiar with the material, the coding phase can be started. In this context, coding is about finding similar meanings through categorizing of data. Each unit of data are added with labels that symbolizes meaning of that specific data point. In qualitative research, material is typically broad, and content is rather complex. Coding helps the researcher to comprehend the material and the meanings within the data. The collected data can be managed by splitting data items into fragments and regrouping data units with similarities collectively. Through codes, the management of the data can be done by rearranging and retrieving data found under relevant codes. The amount of coding depends on your research approach. With an inductive approach, typically all material is coded to find all possible meanings within the data. Codes can be generated through terms heard and recorded during the interviews. Conversely, labels can be developed from the empirical material or even through existing literature. (Saunders et al. 2019, p. 652-655)

After coding is finished, the material is examined and the search for themes starts by looking for patterns and relationships in the list of codes. A theme can be interpreted as a wide category which includes many codes linked to each other, and it is gives an idea that is essential for the original research question. The constructed themes can be divided into two, some become main themes, and some become secondary-level themes. The last step of the pro-

cedure is to refine the themes and relationships found. To end up with a well-structured analytical framework, it is important to have a coherent set of themes. (Saunders et al. 2019, p. 657-658)

In this thesis, the interview answers were transcribed and coded on a high level. The transcription excluded expletive words and only necessary extracts were included as codable data. With the help of the codes, the general themes were formed.

4.5 Research limitations

As this thesis aims to study how QA activities can be optimized with data analytics, the concept of data analytics is rather extensive, therefore we need to define limitations for this work. Data analytics capabilities will be researched on a general level and the technicalities will be ruled out of this thesis.

Interviews are limited to 11 interviews due to limited time and appropriate extent for a Master's thesis. The interviewees are relevant stakeholders for the QA team chosen based on the researcher's own evaluation. The researcher is a relatively new employee working in the product development sector therefore recognizing all relevant employees from the whole organization may prevent the desired outcome.

Another typical research limitation is conducting the research as an internal researcher. Familiarity can create problems, as you may cut out basic questions because respondents might feel you should know the answers already. In addition, the timeframe may also be a practical problem. (Saunders et al. 2019 p. 220) In this research, the timeframe became an issue in terms of having all wanted participants in the sample. However, 11 interviews are quite extensive for the extent of a Master's thesis, therefore that limitation is not considered to be significant.

5. EMPIRICAL RESULTS

In this chapter, the aim is to analyze interview results and highlight key findings from the case company. The interviews were conducted during at the end of year 2021. Each topic from this chapter follows the thematic structure based on Figure 17.

5.1 Current state of quality assurance activities

I1 explained that testing within the whole company is divided into two layers. The first layer is the testing done by product development organization and the second is done by consultants in the customer success organization. Testing in the product development organization focuses on the core product, and the testing is performed by QA team. Consultants focus on testing customer-specific configurations made on top of the core product. (I1) The QA team has several responsibility areas, such as manual user story testing, test automation, release acceptance testing, feature acceptance testing, performance testing and security testing (I1, I2, I3a, I3b).

QA management has an objective to lead the QA organization to make sure the product is high quality, and ensure testing is a key part of the development process (I5a). QA management is responsible for organizing these activities with the right priority and the development within the team (I4a, I4b). Working as a facilitator and removing obstacles was also considered to be an objective for their management role (I5b). Developing processes, working according to the annual budget and recruiting were also recognized as objectives for management role (I5a).

QA employees found that people and their knowledge was the most important resource within the team (I2, I3a, I3b, I5a, I5b). The product at the case company is vast and complex product with a lot of features, thus knowledge that people have about the product was found to be important. (I3a) The education within the company and team is seen to be good, and the induction for new employees is quite extensive. (I3b, I5a) Technical skills were also seen as an asset in QA (I5a, I5b). Basic knowledge of software engineering and development is accumulated from employee's background, but practical skills are typically developed when working in the team (I5a).

According to I31 and I3b, the processes in QA were considered good, and even having an existing process in general was found to be impressive. Based on the interviewees' knowledge, the development of processes was impressive to them because in other companies there might not be processes for QA activities. (I3a, I3b) I1 found that the user story testing process was quite clear, but it could be improved to be more efficient. (I1)

In QA, there are a couple of metrics followed. QA follows how many testable user stories are in queue, thus one threshold is to make sure there is less than a certain amount of user stories in queue. On the product development level, there was a goal to have less than a certain number of bugs at the end of each month. (I1) QA does not however contribute to that metric in a "positive" way - the intention for QA is to find bugs, not to minimize them. I1 continued that for release acceptance testing and feature acceptance testing processes, the metric was to "just be in time". A quality promise was seen to be hard to do as a quality engineer and there was no common way to say something is high quality. (I1)

5.1.1 Success and expectations in QA

From tester point-of-view (I2), the success and performance of QA activities was perceived hard to track. It was recognized that one possible way for tracking success is calculating the number of issues that were found before customers find them. However, success was defined by I2 as follows: "*success is that customer doesn't find many issues, i.e., minimizing issues in release"*. (I2) For I3a, success was to release high quality features. High quality is accomplished when the developed feature is something the user wants and that works for the user, not just something product development thought would be good. The outcome should be that the user finds the feature useful, and the feature works correctly. (I3a) I3b added that the released feature should not have bugs. When testing a feature, if an implementation works well but is not user friendly, it would be important for QA to affect the implementation. (I3b)

From QA management point-of view, I5a explained that success is measured by few key measures, like the success of testing processes and success of key activities. Such key activities are user story testing, release acceptance testing and feature acceptance testing, which are followed constantly. Test automation success rate was also followed up all the time. In addition to technical measures, there were also soft measures to measure employee satisfaction, based on the EOS framework. Some goals were set and measured quarterly, and the goals were followed on a weekly basis. (I5a)

5.1.2 Current state of product development stakeholder activities

I9 explained that product management is responsible for making sure development is doing the right things that create value for customers, and currently also takes care of the operative activities. In product management, new product features are designed and then communicated with leadership, marketing, architecture, and other stakeholders. Deprecating old product features is also one task of product management. Product management activities involve strategic thinking to understand what should be done in relation to market. With competitor

analysis, product management can understand the market and the competitive situation, to later determine what product related decisions should be done in the big picture. (I9)

User stories are typically created by product management together with other stakeholders such as user experience (UX) team and architects. I9 explained that in an optimal situation, user stories are based on design which are split into implementable user stories. In practice, it was found that the user stories were not created in that order, instead, it typically has happened in the opposite order. There are also architectural and UX design stories that specify the more detailed stories. The creation process of user stories should start from having a concept, then a more specified implementation design, which is later broken down into implementable user stories. (I9)

5.2 Challenges affecting quality assurance

5.2.1 Challenges in quality assurance

In the case company, I2 said the developed product had a vast number of features. During testing (I2), QA personnel had to deal with many different features and teams. One difficulty had always been that QA did not have awareness of each feature. In addition to that, human communication was not always efficient. For example, development might progress although previous user story is still under testing. Such development changes might have affected ongoing testing activities. (I2) This human communication refers to communication between developer and tester.

In QA, the queue of user story testing tasks was seen to be a struggle. A long queue of user stories means developers need to wait for the testing to be completed and, in the end, customers end up waiting for the feature to be released. (I3a) In addition to user story testing, the activities of feature acceptance testing and release acceptance testing were found to be time consuming. Release acceptance testing was done on a monthly basis, and it consumed a lot of resources. Another challenge was writing test cases for all kinds of test scenarios which took time. (I3b) I5a also explained from management perspective, there was constantly more work than QA can do. With a considerable size of a team, the QA work should be organized better. Balancing between available people and work has been a challenge. Also, finding new candidates to join the team was found to be difficult. (I5a)

One resource challenge was the gap between test automation and manual testing. There was no guidance on what should be automated and what should not, currently the attempt was to try and do everything. (I5b) According to (I3a), test automation was not an integrated part of feature development. When doing CI delivery, there should be some test automation for a feature before the release. In CI/CD, test automation was seen to be important. (I3a) I1

found test automation a never-ending demand. When the product expands, test automation is the only way to keep larger features in check regarding regression. (I1)

Regarding resources, I5a found the development of new technologies and tools a risk since the development was not necessarily followed actively. There might be some advanced and more sophisticated solutions in the outside world that could be helpful for QA, but those opportunities were not being scanned due to lack of time and resources. I5a guessed such solutions could, for example, help in security testing, performance testing or even artificial intelligence testing. (I5a)

Involving the usability perspective in testing was an improvement area noted from the interviews. I4b found the technical verifications of testing to be the current strength. The mindset has been engineer-based: for instance, error messages contain error stacks that are not readable for users with less IT skills. Usability perspective has got less emphasis and that was one area that could be improved. (I4b) I5a agreed that QA has accepted the current user interface and was not quite strong in giving feedback on usability topics. If QA had a stronger position in user experience, QA could give better feedback to development. I5a also noted that feedback could be given to product management, since they are mainly responsible for defining concepts for the product. (I5a)

5.2.2 Challenges in product development level

On a product development level, I2 saw communication to be insufficient. Feature teams were found to be quite separated, although one team might have been developing a feature that affects another feature (I2). Another challenge, according to I4a and I4b, was the weak representation of QA in the feature planning phase. Typically, user stories and specifications were already made before QA was involved. Both participants agreed that QA involvement should be earlier. (I4a, I4b)

One challenge identified related to the time consumed when builds (pre-released versions of software) are created (I3a, I3b). I3b considered stable builds to take quite long to build, fastest build time is approximately 45 minutes. In contrast, building the new web client takes only 10 minutes, thus testing is quicker. I3a considered it to be a good goal to have the version built and tested for example within one hour. (I3a) Currently, if a tester finds a bug, and developer needs a new build that includes the fix for the bug, the testing can continue earliest the following day (I3b). However, it was acknowledged that it might require a lot of development to achieve it (I3a).

I3b identified that the new developers' lack of knowledge of the M-Files product is one challenge. Testing requests are written by developers, which include the description of the user story, and often the descriptions might only include one scenario to be tested. With more

knowledge of the product, the more you know what else should be checked, like compatibility, security and other aspects. (I3b)

In product management, there was a need for supporting data sources. I9 explained that decisions made in product management were typically based on intuition. The lack of user feedback and concrete product usage data was recognized to be a current challenge. If usage data were to be combined with market research information, it would be more likely to make better decisions regarding how value is created further to end-users and to customers. Thus, the design phase would need the two data sources to be more available and easier to use. Another recognized benefit of product usage data would be to understand different user roles. When user roles are known, there could be better understanding how different roles use the product. Depending on what different roles use and what the environment is like, perhaps testing could become more accurate and testing could focus on what is used the most. (I9)

Another challenge was having the responsibility of operative work which was not necessarily optimal responsibility for product management. From product manager's perspective (I9), managing the operative work was seen as excessive work since product management should typically focus on value creation activities instead of operative work (I9). Thus, the operative work impeded value creation activities, which were seen as crucial activities for product management.

In product development, the biggest general challenge was the lack of sufficient knowledge of the subsegments in the professional services customer segment. It was recognized that customer-centricity cannot be achieved if product development does not understand segments of subsegments, such as tax auditing. The lack of knowledge of how customers use the product, or the platform, in their daily jobs was seen as a factor preventing customer-centricity. (I10)

5.3 Customer interface stakeholder views

This section focuses on the internal stakeholders who were more involved in the customer interface. Since QA was not present in the customer interface, the stakeholders' views who work directly with customers, were needed to understand the customer viewpoint. Figure 18 illustrates more clearly which stakeholders are involved in this sample.

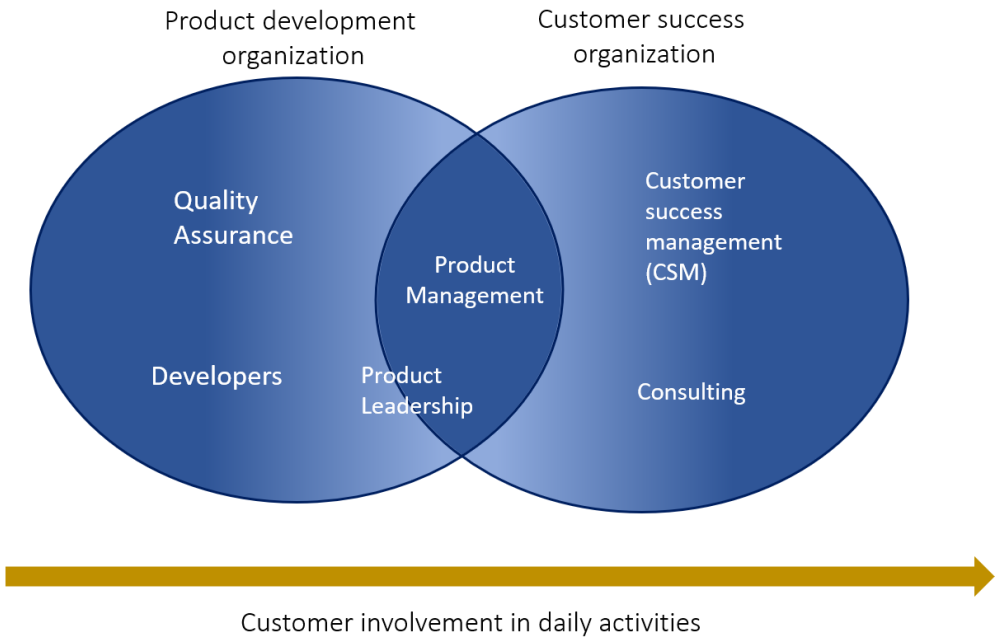

Figure 18: *Stakeholders*

As seen from Figure 18, customer success management and consultants were the closest stakeholders to customers. To clarify, customer success management refers to a distinct team within the customer success organization, responsible for maintaining current customer relationships. Product management works both in the customer interface, but also in the product development organization. Thus, product management was seen as a key stakeholder in the collaboration between the two organizations. This sample also included a developer, I7, who works with critical customer issues, therefore was seen to be qualified to answer customer-related questions.

5.3.1 Views on product quality

I6 and I7 found the quality of the product to be on a good enough level but there was still room for improvement (I6, I7). I6 added, when bugs are reported by customers, they were fixed rather quickly compared to the industry in general. (I6) However, I7 saw that some implementations would require more thorough review before publishing to customers. I7 thought that in the past, some implementations have been published that should have needed more effort in development to achieve production quality. (I7) I6 found it hard to evaluate the quality level of the product since there are no hard facts available. However, I6 recognized that the Net Promoter Score (NPS), a customer satisfaction measure used by customer success organization, could be one source of information to support the evaluation (I6). NPS measures how likely customers are to recommend the service provider on a scale from 0-10 (I8). On the other hand, I11 saw the M-Files product quality to be quite high. Sometimes some feedback was received where new features had issues, or some regression had

happened after the monthly release, but they were usually individual cases. I11 saw the general view to be that M-Files is a Finnish, high-quality product. (I11)

From the consulting point-of-view (I11), the QA team has taken good care of the product quality. The quality was seen to be at a good level since little feedback has been received from customers concerning the core product. For example, one delivery of a feature had recently been quite late because the feature required a lot of testing due to complexity with extra functionalities and various use cases. The feature had been tested for a while and different issues had been solved. However, I11 considered that quality had been taken seriously at the case company and quality was a priority. (I11)

From the customer success management point-of-view (I8), product quality was perceived differently by different customers. Customers that have had more simple solutions with no advanced customization have been happy with the quality of the product. On the other hand, larger customers that have had more complex solutions with more customizations have had more risk for some areas of the product to break after updates which have raised some questions about quality. I8 had no general answer since the perception of quality depended on the customer. (I8)

The leadership of product development (I10) saw the maintenance of high product quality as a difficult, but not impossible, task. The platform has had a multitude and multiple layers of configurability which made it difficult to keep a high standard of quality. Quality objectives and expectations were more heavily defined from functional requirements perspective. The area of customer-centric testing was something that the product development organization had not dwelled into, yet. When QA did not understand customers, the testing also tended to lean on the functional side. (I10)

I10 was unsure whether the product quality had been compared to other competitors. The product was seen to be high quality but maintaining quality has been a difficult task due to the multitudes of the product. From the quality perspective, competing with giant competitors has been difficult since the available resources have been on a different level compared to competitors. However, the ratio of developers and testers has been quite high: the amount of quality engineers was high compared to developers and architects. (I10)

5.3.2 Customer expectations

According to customer success management (I8), what customers value has depended on the customer. Some customers have loved the product and only have thought the product to create value to them. Some customers have had a truly great relationship with employees who work in the services side, such as consultants and project managers. (I8) *Services side* refers to the customer success organization employees who help in the delivery phase of the M-Files product. I8 also explained that some customers were still on a more active part of

their customer journey and there was constant development. The M-Files product itself has received quite good feedback, for example based on NPS results. Sometimes negative feedback has been received, but mostly positive feedback has related to the product. (I8)

According to I11, the M-Files product fulfilled customer expectations in various levels. The basic and core functionality that M-Files offers with metadata-based document searching is what customers have truly valued due to easier information management in their business activities. However, the new functionality and extensions to other systems have not matched the level of easiness based on customer's expectation and how it has been presented to customers. For example, connecting M-Files to network drives, Sharepoint, Teams and Google Workspace was not considered easy. In addition, artificial intelligence functionality was found to be quite limited, and customers have had larger expectations compared to what the functionality had been in reality. (I11)

To improve the ability to meet customer expectations better, it was seen to be beneficial for product development to understand what is built for customers, what use cases are implemented for them and what it requires for services side employees to implement them. I8 saw that product development was not that flexible and agile in a way, thus it was difficult to get certain features in the product. The product roadmap was quite fixed, and a business decision had been made to focus on a certain customer segment and build the product towards the direction that serves them. However, a part of customers, who did not belong to that customer segment, felt the product was not getting features that they would need and appreciate. (I8) I11 agreed that there are many feature improvements and requests coming at a fast pace that all simply could not be implemented. (I11)

5.3.3 Customer feedback on the product

Customer success management (I8) has received feedback regarding the difficulty of using the product. I8 considered it to be tough feedback since there is no knowledge of whether the product is difficult to use or if the metadata structure configured in a difficult way. (I8) Like previously mentioned by I11, I8 also mentioned that customers felt the user interface (UI) of M-Files to be a bit outdated. The desktop client, that users use most heavily, was quite outdated. Customers are also becoming younger, and expectations are much different compared to more mature users. The current approach was seen to be quite engineering oriented. (I8)

On the consulting side (I11), feedback was mostly received of the development work that is done on the consulting side. Consultants customize the M-Files product for the needs of the customer and that development layer might leave bugs and errors that more likely appear in the customer interface. Some feedback also tended to relate to customers wanting the product to help with their specific needs, and often the feedback also included comparison to

older, previous solutions. Regarding the core product, consulting has encountered some feedback relating to the web and mobile experiences. In addition, the desktop client was found to be outdated. Many customers might constantly be on the go, and especially during the pandemic, work-life has changed to a work-from-anywhere type of way of working. Thus, a more coherent experience between the mobile, web and desktop client was seen to be one area to polish and develop forward. (I11)

According to I8, customer experience was not actively measured at the time of the interviews. Previously, customer satisfaction surveys were sent to customers with low response rates. The only way measurement was done continuously was through a simple NPS question that was sent twice a year. Customer experience measurement was also seen to be something customer success does daily by working together with customers. (I8) I6 said customer experience was measured indirectly through analytics data of used features. There was still room for improving the measurement of customer experience. I6 also recognized that NPS questions were sent to customers by the customer success organization, but I6 did not have more information about that. (I6)

5.4 Customer awareness in QA

Based on the interviews, it was certainly clear that awareness of customers in QA was scarce (I3a, I3b, I4a, I4b, I5a, I5b). According to I2, some assumptions and guesses of customers could be made on a high-level based on company-wide presentations. Customer reports of issues could be found from the issue reporting tool, and those issues might have included some descriptions about the environment's customers operate in. (I2) However, more detailed information was not available. For example, I2 said compliance requirements was one area that lacks details. Customer environment details also often came as a surprise, since customer reports from the past have included environment information where they have had surprisingly old versions of, e.g., SharePoint and customers have not wanted to update the versions. (I2)

I1 had some knowledge on what kind of roles the end-user represent. According to I1, customers have admin users who use M-Files Admin tool. The goal is to train 1-2 people from the customer company to use the M-Files Admin tool, so they can administrate their company's M-Files system. Other users use the M-Files client, and most users are client users. Typically, the users have different kind of backgrounds, for example on a technical perspective, some users can barely write an e-mail while some can read source code. (I1)

The current level of awareness of customer requirements in QA was scarce. I4a said QA did not have any information on requirements. Related information had flown through feature descriptions by product management. (I4a) Customers had differing requirements and some individual requirements were heard through meetings together with product management or

feature teams' meetings. I4b also participated in issue prioritization meetings, where requirements sometimes might come forth. (I4b)

There were some different views on the statement whether QA validates customer requirements or not. According to I3b and I4a, QA did not validate customer requirements (I3b, I4a). I4a said QA did currently focus on the functionality perspective, but the customer viewpoint was not receiving as much attention. However, it was pointed out that QA would listen if they had more knowledge about the customer viewpoint. (I4a) On the other hand, I4b considered that the information does flow to QA, and often during testing, you get a hunch on what should be tried. Customer requirements were verified, but not always. Handling customer requirements in Agile was seen to be problematic because there is no ready list of requirements available compared to the Waterfall methodology. However, the consensus was that QA could have more awareness of customer requirements. (I4b) From tester point-of-view, validation was done in some sense, otherwise QA would never receive user stories to test. The tester felt that he necessarily does not need to know about it, but perhaps could do a bit more to contribute. (I1) The assumption was that product management does some analysis, but it would be great to know the use cases to make sure everything actually works as it should. When doing specifications, the use cases could be thought more closely, which could be an input for QA. (I4a) I2 saw QA to be strong in covering functional validation, and the edge case functionalities (I2).

The awareness of customer feedback varied between QA members. Some QA testers considered they did not know much about feedback of released features (I1, I2, I3a). It was assumed that if something crashed in the system or the response would be negative, then testers possibly would get feedback (I1, I2). I2 saw that it would be nice to know how customers have taken into use new features. M-Files Analytics tracks the use of the system but seeing how many support tickets were generated as a result of a feature would be interesting. Overall, I2 would have appreciated to know more about customer feedback. (I2)

QA testers had regularly come across customer issue reports, which the testers saw as the main source of customer feedback (I3a, I3b). However, the testers felt that there was no feedback actively available of how customers had taken new features into use and how the customers had received the new features. Information about the success and pain points of implementation projects were available through broad presentations within the company, but no information about recent releases and how they were received by customers. (I3a) I3b said they would appreciate to see presentations about such topics by "sales or consultant people", i.e,. by the customer success organization. (I3b).

I4a found that customer feedback has not reached QA. It was recognized that customer support gets plenty of feedback and some feedback reaches product management, but not QA.

There was no systematic way to transfer information and information was exchanged randomly at the time. For QA, I4a would appreciate feedback for example to help evaluate whether QA does "*too good*" testing. Testing can be done infinitely and there was no awareness in the team whether QA does extra work. Since there was lack of understanding of use cases, I4a felt that all kinds of combinations were tested too often. Key questions were identified: What is the sufficient level of testing? What features are used, what are not? By acquiring such information to answer those questions, test automation resources could be optimized, which could give capacity to test something else. (I4a) I4b found that some feedback did reach QA but was not communicated to the team. Some feedback was purposely filtered through management, however, there was still room to improve the level of communication. (I4b) I5a emphasized that most customers and end users are happy with M-Files product and excellent feedback has been received from customers. Most of the feedback QA has gotten from customers were negative since the feedback was received from customer support. Customer support has been the contact channel for customers when they have faced problems with features or new releases. QA has received those kind of bug reports and often they have been high priority tasks where a fix was needed to be delivered. Customer support has been the channel QA gets feedback from, and QA then acted on that feedback. (I5a)

5.5 Current analytics capabilities

M-Files Analytics has gathered telemetry data of the product usage. Telemetry data has been generated by the system, not by customer's input. According to I9, the benefit of M-Files Analytics has been to get an overview of the product utilization rate and what areas of the product are being used the most. By that information, better questions can be asked that help in the design and development of a better product to customers. However, the use of analytics has been in early stages. I9 finds that the product has not yet been designed to be better with the help of analytics, but there has been progress happening. (I9)

The adoption of M-Files Analytics has been slowed down due to technical difficulties. Data has not been reliable enough to users. The reporting and delivery of data has not been transparent or self-explanatory. The adoption of analytics has lacked technical enablers, in addition to legal and information security roadblocks. Those challenges had been resolved and the delivery of data through the reports were being tackled at the time. (I9)

According to I9, the main target segment for the analytics data has been the customer success organization who wants to see the product engagement and see whether the value matches what has been offered or promised. If customers don't use the feature, those cases could be investigated and see whether there could be done something to create the value. I9 found that other functions, such as marketing, could want to see which customers were likely to take a new feature into use (provided that a permission had been provided to do

marketing research), and research how value was created and how it could be achieved with other customers. The training and enablement organization were also seen to be interested in knowing the adoption rates by new customers to ensure the adoption has started as wanted by the help of trainings. (I9)

The target group for sharing the analytics data was mainly for managers and upper management. On the other hand, the intention was to have it available for everyone so anyone could make defining queries to data or challenge plans. Still, the main focus was to aid managers, not necessarily those working on the tactical level. I9 found that there were two ways to share the analytics data across functions. First, key metrics could be created and visualized. They would be included in a constantly followed, updated and easily readable report. Another way was to give easy tools for users, i.e., each employee, to find the needed analytics data by making queries by themselves. That would allow each user to independently explore the data. However, I9 said that other ways could be possible if other ideas would come up. (I9)

The analytics data was also collected from the product internally. If there were processes or things that were used in processes, those areas could have been examined if the processes worked or not. For example, investigation could have been done if different vaults were used as they were expected, and if not, those problem areas could have been caught and seen how the process could have been developed to better serve its purpose. From a technical perspective, it was seen that the data could be integrated with development tools if there was a need for it. From QA perspective, I9 saw that there could be a constant report for QA where, for instance, certain feature flag related data could be observed. (I9)

At the case company, predictive analytics adoption was already explored for the benefit of customer success segment through a project that was done together with Oulu University. The project was about creating a behavior model of customer churn (customers unsubscribing). The idea was to identify key usage parameter trends that most likely forecast that a customer is going to churn and to identify churning customers as early as possible from usage data. That had been modeled through machine learning methods. Through that model, it could have been reflected on available customer data and some alerts could have been created if customers were turning towards that type of behavior. (I9)

5.6 Recognized solutions to current challenges

During the interviews, many ideas came up on how to solve current challenges. For solving the challenge of long queue of user stories, I3a suggested to develop a way to detect which user stories are clearly new and which change old functionality that is already covered by test automation. Another option was to have developers partake in testing activities. I3a saw that testing less could be one option in general. (I3a) To avoid idle time during manual testing activities, having a demo for the next user story could be held before the current user story

is tested. To improve efficiency during testing, one recognized way was to write only main test cases instead of creating test cases for all tested scenarios. For release acceptance testing, I3b would have appreciated having automation especially for old features and the remaining manual testing efforts could focus on new features. (I3b) For test automation guidance, I5b would have appreciated intelligent filtering of what areas of the product should be automated and what should not, i.e., being smarter with test automation efforts (I5b).

On a product organization level, documentation about implementations of new features could help prevent the challenge of not knowing about other features. To avoid issues in communication, I2 said there could be improvements in processes. When feature is being designed and implemented, there should be evaluation if it affects other teams. I2 found that this practice would help keep everyone in the loop. (I2) I7 agreed that there is a need to understand how implementations affect the product overall. I7 considered that sometimes implementations have not been thought through well enough. The product is vast and there is a need to understand how the implementations affect the product overall, for example in the cloud environment. In the past, there have been cases where implementation have not been evaluated or tested well enough, for example from the performance perspective. I7 hoped for QA to raise more actively questions of implementations and challenge whether the implementation is something wanted for the product (I7).

As mentioned in previous sections, product management is one key stakeholder in between the product organization and customer success organization. Since product management currently handles both value creation activities and operational activities, the responsibility load was seen to be rather excessive. I9 felt that the product management load could be reduced by having a Scrum Master in development teams to manage operative work instead of product managers. (I9)

5.6.1 Information needs in QA

Understanding customer environments better was evidently the most important information need recognized in QA (I1, I2, I3a, I4a, I4b, I5a, I5b). With better understanding, testers feel there is an opportunity to simulate the customer environments in QA, which could give more confidence for releasing. In QA, there was also a need to know the amounts of data customers have (I1, I4b) and where is the data stored. (I1) Although there seemed to be somewhat knowledge of customer data amounts to some extent as *"it always comes as a surprise although it should not be a surprise*", there is a need to know what the amount is in reality (I4b). Simulating customer environments more easily and efficiently was seen to be an improvement area for QA, but it is not an easy task (I1).

In QA testing, I2 said there are a lot of different variables in terms of what should be tested, such as the different client options. In release acceptance testing, the various areas to be

tested was a concern. For example, if Windows 8 is not used by customers, that could be eliminated from tested operating system versions. Getting data about how important a feature is for the customer was considered beneficial. Having such knowledge could help evaluate how much testing should be done. (I2)

Diving deeper, understanding even the customer's business and their needs was recognized as beneficial knowledge: "*Why are they purchasing M-Files? What are their needs?*". Knowledge about the different users was also raised as a need. For instance, understanding what the kind of (M-Files) client skills they have. Further, other needs such as GDPR requirement related details would be beneficial. Other preferences were found interesting, for instance, choosing either cloud or on-premises environments: what are the factors behind such decisions and knowing which preference they have would be beneficial in testing. (I1)

5.6.2 Improving customer-centric QA

One target customer segment at the case company was described as the professional services segment. Professional services entail all companies that involve knowledge-intensive work in their daily activities. According to I4a, having knowledge of customer profiles and domains from the professional services segment (e.g., health care) could be one solution to improve customer-centric QA activities. Some examples of complex environments were desired. The professional services segment is a large segment and might entail all various of needs. Information on how the customer uses the product, how they experience the product was seen to be interesting information. I4a found that current employees have used the product themselves long enough not to notice problems regarding the product. (I4a)

To improve customer-centricity in QA, I6 found that the NPS score, and the M-Files analytics data are the beneficial information sources. With analytics data, I6 saw that there should be better visibility for everybody, and plans for actions were ongoing on how the information was going to be shared, like discussions of what kind of reports should be created. The data was already in the backend, but the problem was with creating dashboards. Recently a new hire had been dedicated to developing dashboards relevant for product development. (I6)

Actions to improve information sharing were already ongoing through meetings where other organizational functions came to the meeting to hear what "lessons learned" situations had been recently. Another way recognized to improve customer centricity was to do job rotation, where for example developers could visit the customer success side of the organization and observe their daily tasks. I8 also elaborated on a high-level what kind of end-users exist. Super users are more advanced users who can advise other end users to for example user advanced features. Business administrators use the administrator tool, and they can make changes to structures and the solution itself. Champions are users who bring a positive spirit to the company to help with change while adopting M-Files. (I8)

I11 found that the collaboration between consulting and product development organization had somewhat difficulties. I11 saw that product management function was quite narrow, having too few employees to handle it. Consultants had found the collaboration with product development, and product management, to not be on a good level. I11 personally found it easier to be in touch with other departments due to their own management position, but other consultants felt they do not get sufficient information or enough collaboration. (I11)

Having consultants in the same Microsoft Teams channels together with product development employees was seen to improve information flow between the organizational functions. However, consultants found it difficult to get information through official communication channels, when consultants approach product development through support cases, and support approaches product development. However, I11 recognized this mechanism was to protect the product development resources and understood the reasons why the process had been like that. (I11)

To improve collaboration between functions, I11 found that workshops should be held together between consulting and product development functions. In the past, QA and consultants have had meetings together where the usage of new features was taught and the challenges regarding the features were presented. Information sharing meetings where key product development employees would share information towards consultants would be beneficial, and vice versa. Currently, the information flow was found not to be on a great level. (I11)

Understanding where value comes from to customers was recognized as a major information need, not only for QA but for product development. I10 suggested that QA shouldn't merely focus on functional, quantitative tests but also see the qualitative side of testing. Data could be added as a source of truth, like in the usage between versions, and analyze reasons if some changes in behavior have happened. Ultimately, I10 found that it comes down to trust, security, intuitiveness, ease of use, accessibility, and functionality. The more there is understanding of the end-user and how they use the product, the more those cases could be focused on in testing. How the product is used, what features are used and what things users are doing were seen to be relevant questions to know. When the use cases are known and the customer segment representation is understood, I10 thinks those use cases should be heavily tested. (I10) I7 saw that the usefulness of features should be evaluated together with customers. The evaluation could specify both reasons why a feature is useful, and it is not. I7 thinks measuring should be done through customers. (I7)

One solution for QA is to be a part of the validation process. I10 suggested that QA could go and meet the customers and their end-users for example once in a quarter. When meeting end-users, questions could be asked regarding the product. Meeting end-users could result in new observations of how the product is used and testers could recognize own biases regarding assumptions of users. I10 feels the shift should turn from how product development

employees use the product, to how end-users use the product and what the quality and feel of the use cases are. (I10)

Many stakeholders recognized challenges and even solutions to improve customer-centricity in quality assurance activities. The main findings are collected in Table 2.

Affected stakeholder	Painpoint	Root cause	Recognized Solution
QA, Product Management	Testing involves different variables concerning what to test	No sufficient knowledge on what product areas and environments are relevant	- Deprecate non-valuable product areas - Prioritization based on customer priority and feature usage.
Product management	Product management activities involve non-value creation related activities	Product management resources are scarce	Adding operative roles to support PM
QA	Scarce awareness of customer information cause testers not to consider non-functional aspects as thoroughly	- Information sharing practices are not consistent due to organizational silos - Analytics capabilities are under development - No data available on customer environments	- Information sharing sessions/job rotation with stakeholders - Involve QA in the validation process by meeting end-users - Educate QA what creates value for customers and end-users in target customer segments - Continue the development of analytics - Add analytics and share the data on environments - Help testers recognize non-functional aspects in testing
Stakeholders in product development organization	Not having sufficient documentation on features and not being aware of all product features	Finding such information is difficult or such information is not documented explicitly	Work on improving and documentation practices and encourage documenting to become more consistent

Table 2: *Recognized challenges and solutions*

Addressing the challenges with recognized solutions require co-operation between different stakeholders. The analysis of current challenges and presenting other additional solutions will be addressed in the following chapter.

6. DISCUSSION

6.1 Comparing findings with literature

Next, the empirical findings are compared to literature findings to form the final research results. Literature findings relate to the literature research on chapter 2 and 3, and empirical findings refer to content on chapter 5. This chapter will include recommendations for the quality assurance team, and product development organization in general.

6.1.1 Enhancing quality assurance activities

Success in QA is that customers do not confront issues after monthly releases. In other words, success is to release high quality features; the features are something users want and that work functionality-wise. The development outcome should be something users find useful and that features work correctly. Based on interviews, the overall quality of the product was seen to be quite high and customer feedback regarding the M-Files product is mostly positive feedback. However, some improvement areas regarding QA activities and other stakeholder activities were found to improve the quality level even further.

As mentioned in chapter two, there are two perspectives to quality, conformance to specification and customer satisfaction (Mistrik et al 2016). Verification and validation are two important activities in software testing. Verification answers the question "*have we produced what is specified?*" and validation answers the question "*have we built the correct product?*". (Homès 2012, p.7) Thus, validation makes sure requirements are met with the development, and verification checks the system matches the functional specification. (Batarseh & Gonzales 2018) Based on interviews, the QA team is strong in verifying implementations, i.e., checking development is done based on the functional specification. The leadership of product organization emphasized that QA should be a part of the validation process. From the development perspective, it was seen that QA could sometimes challenge implementations and make sure the implementation answers customer needs. So far, QA has assumed and trusted that validation happens outside of QA. Literature finds that validation, "*have we built the correct product*", is a part of testing activities. Being a part of the validation can help QA to evaluate whether the wanted quality level, and if customer satisfaction, has been reached. Thus, there is a need for QA in the case company to achieve a stronger presence in the validation activity.

Customer facing stakeholders expressed that some new functionality and extensions were not as easy to use as they were thought to be. Customers find that some areas of the product are outdated, like the user interface of the desktop client. Testers also recognized that it would be important for QA to be able to affect the implementation is not user friendly although

the functionality works. There is a need for QA to be able to give feedback directly to product management, preferably in a proactive manner.

6.1.2 Information needs and sharing

Intelligent organizations can create, acquire, transfer knowledge, and adapt its behavior based on new knowledge and insights. Learning begins when new ideas and knowledge are applied to change the organization's goal**.** Organizations seldom challenge their norms, goals or policies in relation to changing environments (Choo 2002, p.12-14). The case company has a need to break information silos between functions on the organization level, and also within teams in the product organization. Based on interviews, the awareness of customers was limited in QA. Testers currently rely on guessing, for example, what system environments customers use and what type of end user roles exist. QA is aware that there is little knowledge about end-users, although there is end-user research done within the company. Actions how to improve the awareness of customers have not been known before. Information silos can be avoided by sharing knowledge, for instance with the UX team, who are involved in end-user research, to educate QA personnel. During internal stakeholder interviews, it was suggested to try job rotations together with functions that work with customers, so QA could see what challenges and thoughts emerge at the customer side. The leadership of product found it a beneficial option for QA to meet directly with customers, moreover, be involved in the validation process.

Product management works as a link between customers and development (Sauvola et al. 2015). Product management is responsible for defining what should be developed and that the development answers the needs of customers. At the case company, the product management function was found to be narrow and overloaded with current tasks. Product managers currently deal with excessive tasks, that necessarily do not relate to value creation activities. The performance of product management is crucial for QA because the quality of the product is also dependent how customer requirements are understood and answered.

According to Lean thinking, anything that does not add value for customers should be considered waste. Waste also needs be recognized. Waste is, for instance, features that are not used and documentation that is not read. (Baumgartner et al. 2021, p.35-36) Based on interviews, there is no shared knowledge on what product areas are important for customers. As Fabijan et al. (2017) et al. explained, having data about how customers use the product can help prioritize development activities to become more accurate. Product usage focuses on what customers do, and not on what the customers say. (Fabijan et al. 2017)

6.1.3 Analytics capabilities and needs

In businesses, descriptive analytics is the most common type of analytics applied. Data is described without any additional calculations and the aim is to understand decisions made with historical and current data. In other words, descriptive analytics helps make data-driven decisions. (Raghupathi & Raghupathi 2021) Data is generated from both external and internal sources in organizations (Namvar et al. 2021). Visualization is commonly used in descriptive analytics. (Raghupathi & Raghupathi 2021) At the case company, the main type of analytics performed is descriptive analytics. The analytics data can be visualized by creating reports with key metrics. The wider utilization of analytics is under progress within the organization. Value of information is created when the information is utilized in operative work and activity development (Laihonen et al. 2013, p. 7-12). To extract the value of information, QA needs to *use* the information in their activities.

Literature introduced change-driven testing as a way to guide testing efforts to achieve test intelligence. The idea of change-driven testing is to align testing efforts with relevant code changes. (Goericke 2020 p.2-8) QA management expressed concerns regarding the scalability of the product due to the high and growing demand of testing efforts. To tackle scalability, change-driven testing could guide testers to focus test efforts only on relevant areas of the product.

6.2 Proposal for case company

There are two questions testers need to think about: *what to test* and *how to test.* Testing efforts can be optimized by the adoption of test intelligence to guide what areas of the product should be tested. The second perspective requires effort from product management to reduce non-valuable features and create information channels to communicate the importance and maturity of features. To evaluate the quality of implementations, QA also needs to know what elements are valuable from the customer and end-user perspective. In addition, QA needs to have chances to affect decisions earlier regarding quality, to be more proactive in providing feedback. The concrete suggestions for QA to improve their activities are collected in Figure 19.

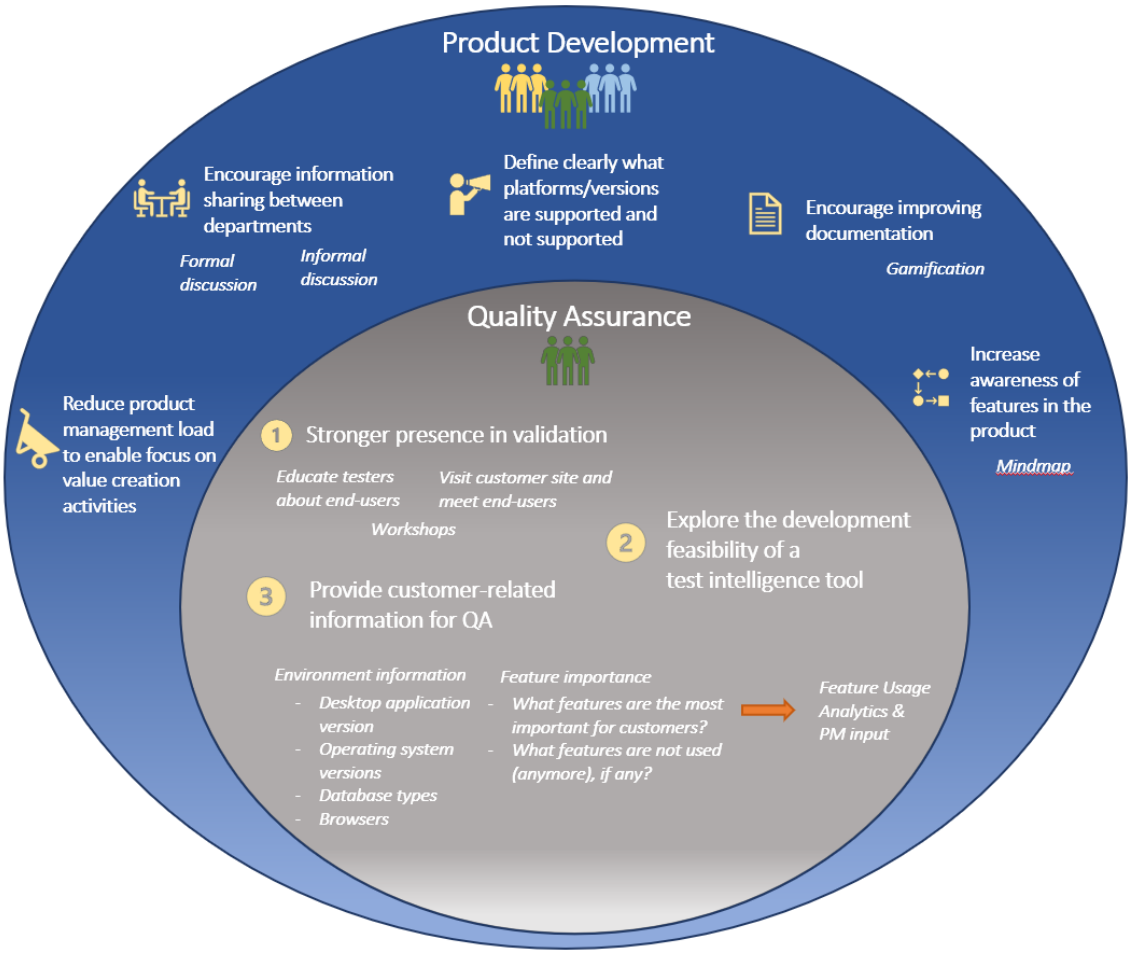

Figure 19: *Ways to guide QA activities*

Figure 19 summarizes the suggestions for ways QA can become more data-driven and customer-centric. The suggestions are two-leveled, some suggestions are directly for QA management as they are responsible for organizing activities in the right priority, and some suggestions are for the product organization level.

6.2.1 Suggestions for quality assurance

In software engineering, understanding customer value and taking it into account is crucial (Aho & Uden 2013). As mentioned in the previous section, the success in QA was defined as having a product that works and that users find useful. The framework in Figure 19 includes proposed actions to achieve such success.

To influence the quality outcome of a feature, QA should be involved in feature planning earlier. So far, QA has given feedback of the development only after the implementation is implemented. Some QA members have been involved in feature planning due to own active take, but the involvement has not been systematic across all teams. The earlier QA is involved, the better testers can comment on quality aspects and understand what customer problems the feature is supposed to solve. Including QA earlier can help avoid situations

where features are developed to be less user friendly, as QA would have more direct contact with product management during the planning phase.

Literature recognizes that one challenge in QA is not deprecating older platforms, operating systems, browser versions and databases (Nader-Rezvani 2019). The areas of the product which do not bring value to customers should be minimized in testing. The product should be analyzed if there are areas of the product that used to be valuable before but are not valuable anymore. Such product areas should be considered for deprecation. Thus, QA needs input from product management to evaluate what product areas to prioritize in testing. When such input is received, for example release acceptance testing efforts could be re-evaluated if it includes non-valuable product areas that are tested manually.

Feature usage analytics could provide input on what are the most used areas of the product. In addition to product usage, product management can give input on the importance of the feature. For instance, a new feature might not be used as much yet, but the feature might be planned to replace an old feature. Such information is also needed from product management to evaluate the importance of features. The priority or importance of an implemented feature could be categorized on a high level. The importance of a feature should be determined based on at least the combination of usage analytics and general evaluation by product management. First, there is a need to find out if such importance of features is known already in product management. If not, the importance should be formulated. For new features and user stories, it could be explored if they could also be prioritized. For example: how many customers are waiting for the feature, what type of customers are waiting (large or small), what is the business potential of the feature? Not only is the importance of a feature valuable information, but also the feature maturity. In this case, feature maturity refers to how ready is the development of the feature. For QA, such information can help evaluate how much testing effort is reasonable to use in the user story or feature. When testing the monthly release, testers could evaluate if the feature is worthy of testing with the available resources. Is the feature already released to customers and only requires bug fixing, or is the feature completely new? Having information of the importance of features could give guidance in testing. The usage data and importance evaluation of features is substantial information for testing because test automation efforts can be aligned with the usage of the product.

Another action point for QA is to educate testers about value-based thinking. Technical aspects are only part of value creation and there is a need to understand more broadly what features should do to ensure customer satisfaction. Ensuring customer satisfaction is a joint effort with different stakeholders. To improve customer-centricity in QA activities, QA can work together with customer-facing internal stakeholders to increase the testers' awareness of customers and end-users. For example, QA can work with Customer Support and Customer Success teams to understand current pain points regarding to product quality, or with

Training and Enablement teams to see what type of end-users exist and how new users are able to learn the use of the product. Testers could meet different types of users: super users, business administrators and champions. Another possibility is to arrange a direct visit to a customer to see the usage behavior with the product. The more QA can learn about customers, the more testers can validate if implementations answer the needs of end-users. Bringing such awareness can help testers to not only rely on their imagination. The transfer of information should be systematic. With better understanding of end-users, QA can also become more capable in giving feedback during usability testing. To give feedback about usability, there is also a need to create a formal feedback channel to product management when usability concerns come up, since product management is responsible for defining concepts for the product.

Testers need better availability on customer related information. There is a need to know what type of customer system environments exist, so that testers could mimic the environments more closely during testing activities. For example, information about M-Files desktop application versions or operating systems versions is beneficial in testing, because such information can bring guidance in what versions testing should cover. The information would help solve the issue of testers making assumptions on system environments. To create analytics reports that serve QA needs, the information needs recognized in this thesis should be communicated to product management and the analytics team. To make the information use more systematic, the possibility of integrating the data to development tools could be explored. For example, if most customers use operating system version A, the test requests could include such analytics data to help guide testers in their environment related decisions.

Having supporting data of the context can help practitioners make informed decisions in software development (Matthies & Hesse 2019). To help testers make more informed decisions daily, the feasibility of test intelligence could be explored in the case company. Creating a test intelligence tool requires upfront development efforts and does not necessarily bring immediate benefits. However, in the long run, it could bring benefits when all areas of the product don't have to be tested. It is important to note the tool would only give *guidance* to testers where to focus their efforts, not the necessarily the absolute truth what should be tested and what not.

The investigation of the feasibility of change-driven testing could be started from one product area. There would also be a need to discuss with software architects and developers about the feasibility. The case company has a complex software system; thus, the development of test intelligence is not necessarily an easy task. To implement test intelligence, there is a need to gather data on the changes made, stored in the version-control system and compare it to the testing tool that contains data on what test suites cover. Visualization of such data would require a tool to be developed to track the test coverage of changes and to identify

test gaps. Another option is to investigate if such visualization tools already exist that could be utilized.

6.2.2 Suggestions for the product development organization

Product management is a key function in ensuring the company develops a valuable product for customers. As mentioned in the previous section, product management has had narrow resources and has been involved in activities not relating to their main purpose, like operational activities. To ensure customers keep using a product that brings value in their daily business activities, it is vital to have product management focusing on their core activities, i.e., value creation activities. The performance of product management also affects quality assurance activities. Thus, there is a need to invest in product management and involve new roles to take responsibility of operative tasks. In Scrum, Product Owners work in between stakeholders, and Scrum Masters oversee the process (Larusdottir et al. 2016). Thus, having such roles could support the product management function.

Another suggestion is to improve information and knowledge sharing practices. The organization can encourage to share information and knowledge by information sharing sessions, improving documentation, and arranging job rotations. One option is to gamify documentation practices to encourage and create excitement for improving documentation. During information sharing sessions, good practice is to create notes to summarize what has been learned to ensure the benefit of such sessions. As found in interviews, the most important resource in QA is the people and their knowledge. Not only is documentation one way to extract knowledge and educate employees, but also a way to avoid risk of losing knowledge when transitions of employees happen. In general, information sharing should be encouraged by e.g., information sharing workshops arranged regularly and even through informal sharing. Based on literature, information flow is found to be effective in cross-functional teams during problem-solving (Barney 2014). Forsten-Astikainen et al. (2017) also favored the encouragement of creating communities of practice that could possibly help break information silos if communities are created horizontally (Forsten-Astikainen et al. 2017). In addition, after the product development organization learns more about the subsegments of one certain customer segment, the information should be shared across the product organization.

Furthermore, the product development organization could provide appropriate materials to educate and learn about features. The documentation should be done in a way that is easy to find and easily accessible for employees to find in a dedicated repository. The possibility of visualizing the product features could be explored, for example into a mindmap format. The mindmap could include all the features or have aggregations of features to create appropriate-leveled feature themes. Each feature theme on the map could have links to appropriate documentation and employees who are responsible or have knowledge of the features.

Visualizing may not be an easy task, but the possibility could be explored. Having information in one place would help employees to find necessary information:

- What is the feature?
- What needs does the feature fulfil?
- What use cases relate to the feature?
- What is the feature maturity?
- Other development related documentation

Getting answers to such questions can help QA to guide their activities to become more customer-centric, as they can understand the aspects that affect customer value and how the product features can fulfil customer and end-user needs.

7. CONCLUSIONS

7.1 Answering research questions

What type of activities does QA consist of? What type of internal stakeholder activities relevantly affect QA activities?

QA activities consist of manual user story testing, test automation, release acceptance testing, feature acceptance testing, performance testing and security testing. The testing team is responsible for performing such activities. QA management leads the QA team to ensure the product is high quality and that testing is a key part of the product development process. A successful development outcome is to have a product that users find useful and features work functionality-wise.

QA collaborates with many different internal stakeholders and each stakeholder has an impact on QA either directly or indirectly. The key stakeholders identified are developers, product management and customer-facing stakeholders. To ensure overall product quality, developers are responsible for creating software that answers the need of customers. Product management is responsible for managing customer requirements and creating concepts that answer customer needs. Customer-facing stakeholders have information valuable for product development, but information does not flow between departments sufficiently enough.

What challenges relate to performing QA activities?

In product development, there is a growing demand for testing efforts. The product is being developed fast and there is little knowledge of efficient ways to test the product. QA has no hard facts to guide or prioritize testing activities. Currently, decisions are mainly based on intuition.

The success of QA is partly dependent on the performance of product management. Product management is responsible for creating a vision of a product that serves customers and generates value for their business needs. Managing the product also involves removing non-valuable features or product areas. In the case company, there is urgency to know if non-valuable features exist and whether such areas can be deprecated. If deprecation plans exist, more detailed information on plans currently have not been shared.

Literature considers validation as one testing activity. Currently, validation activities are done outside of QA. The general approach in product development is seen to be engineering based. Testing is not strong in the evaluation of usability aspects.

QA has barely any awareness of customers, what type of system environments customers have, and what type of end-users exist. To evaluate if the product is high quality, i.e., the

product is valuable from the end-user perspective, QA needs better awareness of customers. QA does not have concrete data to verify or correct assumptions. Information and data may reside in other organizational functions and teams, but collaboration and communication are weak between some organizational functions. Thus, information silos are blocking the improvement of current practices.

What customer information is needed for QA to improve activities?

Currently, QA does not have sufficient knowledge about customers and their system environments. To become more customer-centric, QA needs to enhance activities to take customer information into account. To improve testing activities, the following customer data related needs were identified:

- Operating system versions
- Database systems used
- Data amounts in size
- Feature usage analytics data

In addition, the following information can support QA:

- Importance of features/product areas
- End-user roles
- Descriptions of customer business domains
- Customer preferences (e.g., on-premises or cloud environment) and factors affecting preferences
- Customer's general business needs
- Other requirements such as GDPR

To obtain the information needs identified, there is a need to work together with customer-facing internal stakeholders to get better availability on such information.

What type of data analytics can and should be established?

Literature presented four types of analytics: descriptive, predictive, prescriptive and discovery analytics. Findings suggest that descriptive analytics is the most feasible type of analytics to utilize in product development. Currently, descriptive analytics can provide value for QA activities. Predictive analytics capabilities are being explored in other parts of the organization. In the future, predictive analytics could be explored for product development purposes, after the benefits of previous experiments have been found.

In descriptive analytics, visualization can help describe the data in a readable format. The option of including analytics data into development or testing tools to is possible to explore.

Through the possible adoption of test intelligence, QA have supporting visualization the areas of the software that are covered by tests and also the areas that are not covered by tests.

In what ways can QA activities become customer-centric and data-driven?

QA activities can become more customer centric and data-driven by communicating the information needs identified in this thesis to product management. After the needs are communicated, the analytics data needs to be shared by visualization or by exploring ways to include the information into existing tools to make the information use systematic. For guiding testing efforts in a data-driven manner, the adoption of test intelligence tool can be explored.

QA needs to be more involved in feature planning to understand where requirements or the need for the feature derives from, in order to be more capable of perform validation properly. Understanding end-users and their needs help evaluate the implementations from the usability perspective. Thus, QA can become stronger in non-functional testing.

Improvement also needs actions from the product organization level. Product development organization should support and rearrange product management resources to allow product managers to focus on value creation activities. Product organization should also encourage breaking organizational and information silos. Product organization could enable more informal encounters to create natural communities of practice.

7.2 Evaluation and future research

This value of this thesis is provided for the management in QA. This thesis reveals the status of QA activities, what are the key challenges and how challenged can be tackled. The results of this thesis give QA management the opportunity to improve QA activities to become more customer centric and data driven. Not only are key information needs regarding customers identified, but also supporting tools are introduced that can, for example, help solve scalability concerns. This work suggests concrete actions to take within the product organization.

Literature does not provide much research on quality assurance in the context of continuous software development. The interview results and literature cannot be precisely compared since literature on quality assurance focuses on software projects, while the case company context is continuous development. For instance, in continuous development, the gathering of customer requirements is different to software projects. The adaptability of project literature can be questioned whether it is valid for continuous development, as their nature are rather different. Thus, the accuracy of the results can be questioned to some extent.

The time between gathering empirical evidence and the publishing of the thesis affects the practical value, as the product development has had structural changes happen during the time between gathering evidence and publishing the thesis. Therefore, some suggestions

have become obsolete by the time of publishing. However, most findings are still relevant and bring value to the case company.

Data analytics research evolves at a fast pace and has a lot of potential in the future. Utilizing data analytics in software quality assurance is one topic to research further. Especially literature about utilizing more advanced types of analytics in the context of software engineering, and especially quality assurance, is not currently available. However, there is significant potential to improve business performance with better and more accurate data-driven practices in organizations.

7.3 Limitations

In qualitative research, the generalizability of findings may be raised when having only a single case in the study (Saunders et al. 2019, p. 450). In this thesis, imitating this case study might be difficult in another context as interview questions were tailored for almost each participant. This study was highly context specific, and findings mainly support the case company.

Interviews conducted did not involve all wanted internal stakeholders due to time limitations. From the empirical standpoint, a more thorough answer to the research question could have been achieved if the sample included representatives from Customer Support and User Experience (UX) specialists. Customer support could have provided accurate information on direct customer pain points of using the product. UX representatives could have provided answers how QA can consider usability better during testing. The possibility of interviewing customers directly could have been an option as well.

REFERENCES

Adams, W.C., 2015. *Conducting semi-structured interviews*. Handbook of practical program evaluation, 4, pp.492-505. Available from: http://www.robertrenaud.ca/uploads/2/2/9/6/22962838/chapter_19_-_semi-structured_interviews__2015_.pdf. Read 6.10.2021.

Aho, A. M., & Uden, L. (2013). *Customer knowledge in value creation for software engineering process*. In 7th International conference on knowledge management in organizations: Service and cloud computing (pp. 141-152). Springer, Berlin, Heidelberg. Available from: https://link.springer.com/chapter/10.1007/978-3-642-30867-3_13. Read 25.8.2022.

Ahmad, A., Leifler, O. and Sandahl, K. (2021). *Software professionals' information needs in continuous integration and delivery*. In Proceedings of the 36th Annual ACM Symposium on Applied Computing (pp. 1513-1520). Available from: https://dl-acm-org.libproxy.tuni.fi/doi/10.1145/3412841.3442026. Read 26.9.2021

Alsaqqa, Samar & Sawalha, Samer & Abdel-Nabi, Hiba. (2020). *Agile Software Development: Methodologies and Trends*. International Journal of Interactive Mobile Technologies (iJIM). 14. 246. 10.3991/ijim.v14i11.13269. Available from: https://www.researchgate.net/publication/342848746_Agile_Software_Development_Methodologies_and_Trends. Read 3.8.2022.

Barney S., Wohlin C. (2009). *Software Product Quality: Ensuring a Common Goal*. In: Wang Q., Garousi V., Madachy R., Pfahl D. (eds) Trustworthy Software Development Processes. ICSP 2009. Lecture Notes in Computer Science, vol 5543. Springer, Berlin, Heidelberg. Available from: https://doi-org.libproxy.tuni.fi/10.1007/978-3-642-01680-6_24. Read 5.12.2021

Barney, S. et al. (2014). *Software quality across borders: Three case studies on company internal alignment*. Information and software technology. [Online] 56 (1), 20–38. Available from: https://browzine.com/libraries/2233/articles/45788272. Read 13.7.2022

Batarseh, F.A. and Gonzalez, A.J. (2018). *Predicting failures in agile software development through data analytics*. Software Quality Journal, 26(1), pp.49-66. Available from: https://link.springer.com/article/10.1007/s11219-015-9285-3. Read 30.9.2021

Baumgartner, M, Klonk, M, Mastnak, C, Pichler, H, Seidl, R, & Tanczos, S. (2021). *Agile Testing: The Agile Way to Quality*. Springer International Publishing AG, Cham. Available from: https://ebookcentral.proquest.com/lib/tampere/detail.action?pq-origsite=primo&docID=6724614#. Read 8.10.2021.

Biesialska, K., Franch, X. and Muntés-Mulero, V. (2020). *Big Data analytics in Agile software development: a systematic mapping study*. Information and Software Technology, p.106448. Available from: https://www.sciencedirect.com/science/article/pii/S0950584920301981. Read 30.9.2021.

Bobrovskis, S., & Jurenoks, A. (2018). *A Survey of Continuous Integration, Continuous Delivery and Continuos Deployment*. In BIR workshops (pp. 314-322). Available from: https://andor.tuni.fi/permalink/358FIN_TAMPO/176jdvt/cdi_scopus_primary_624178805. Read 29.9.2022.

Braun, V. & Clarke, V. (2006). *Using thematic analysis in psychology*. Qualitative Research in Psychology, Vol. 3(2), pp. 77–101. Taylor & Francis Group. London. Available from: 10.1191/1478088706qp063oa. Read 10.9.2021.

Buse, R. P., & Zimmermann, T. (2010). *Analytics for software development*. In Proceedings of the FSE/SDP workshop on Future of software engineering research (pp. 77-80). Available from: 10.1145/1882362.1882379. Read 14.11.2021.

Choo, C.W. (2002). *Information management for the intelligent organization: the art of scanning the environment*. Information Today, Inc. Available from: https://www.worldcat.org/title/information-management-for-the-intelligent-organization-the-art-of-scanning-the-environment/oclc/38580130. Read 6.10.2021.

Crispin, L. & Gregory, J. (2009). *Agile Testing: A Practical Guide for Testers and Agile Teams*. Pearson Education Inc.

Dingsøyr, T. & Lassenius, C. (2016). *Emerging themes in agile software development: Introduction to the special section on continuous value delivery*. Website: https://www.sciencedirect.com/science/article/pii/S0950584916300829. Read 24.3.2022.

Ebert, C. et al. (2016). *DevOps*. IEEE Software, 33(3), 94–100. Available from: https://doi.org/10.1109/MS.2016.68. Read 29.9.2022.

Entrepreneurial Operating System. (2022). *What is EOS?* Available from: https://www.eosworldwide.com/what-is-eos. Read 29.9.2022.

Fabijan, A., Dmitriev, P., Olsson, H. & Bosch, J. (2017). *The Evolution of Continuous Experimentation in Software Product Development*. Available from: https://www.researchgate.net/publication/317006885_The_Evolution_of_Continuous_Experimentation_in_Software_Product_Development. Read 23.11.2021.

Forsten-Astikainen, Hurmelinna-Laukkanen, P., Lämsä, T., Heilmann, P., & Hyrkäs, E. (2017). *Dealing with organizational silos with communities of practice and human resource management*. The Journal of Workplace Learning, 29(6), 473–489. Available from: https://doi.org/10.1108/JWL-04-2015-0028. Read 30.10.2022

Galin, D. (2018). *Software Quality: Concepts and Practice*. 1st edition. [Online]. Piscataway: Wiley. Available from: https://www.kiwi.fi/display/finelib/O%27Reilly+Safari+Learning+Platform%3A+Academic+Edition. Read 5.12.2021

Garousi, V., Felderer, M., Kuhrmann, M., Herkiloğlu, K.& Eldh, S. (2020). *Exploring the industry's challenges in software testing: An empirical study*. Journal of Software: Evolution and Process. Available from: https://www.researchgate.net/publication/339095645_Exploring_the_industry's_challenges_in_software_testing_An_empirical_study. Read 16.12.2021

Goericke, S. (2020). *The Future of Software Quality Assurance*. Available from: http://library.oapen.org/handle/20.500.12657/22847. Read 10.10.2021

Homès, B. (2012) *Fundamentals of software testing*. Hoboken, New Jersey: ISTE/Wiley. Available from: https://ebookcentral.proquest.com/lib/tampere/detail.action?docID=1120766&pq-origsite=primo. Read 13.3.2022

Kiradoo, G. (2019). *Software engineering quality to enhance the customer satisfaction level of the organization*. International Journal of Advanced Research in Engineering and Technology, 10 (3), pp. 297-302. Available from: https://www-scopus-com.libproxy.tuni.fi/record/display.uri?eid=2-s2.0-85086465511&origin=resultslist&sort=plf-f&src=s&st1=%22customer+experience%22+AND+software+quality&sid=b8511d8d976d976ae2006d502591f7c1&sot=b&sdt=b&sl=57&s=TITLE-ABS-KEY%28%22customer+experience%22+AND+software+quality%29&relpos=19&citeCnt=1&searchTerm=. Read 16.12.2021

Laihonen, H. et al. (2013) *Tietojohtaminen*. Tampereen teknillinen yliopisto, Tietojohtamisen tutkimuskeskus Novi. Available from: https://trepo.tuni.fi//handle/10024/116695. Read 5.8.2022.

Larusdottir, M., Cajander, Å., Gregory, P., Cockton, G., Salah, D., Kuusinen, K. & Nauwerck, G. (2016). *Stakeholder Involvement in Agile Software Development. In Proceedings of the 9th Nordic Conference on Human-Computer Interaction (NordiCHI* '16). Association for Computing Machinery, New York, NY, USA, Article 139, 1–3. Available from: https://doi-org.libproxy.tuni.fi/10.1145/2971485.2987678. Read 24.9.2021

Lindgren, E. & Münch, J. (2016). *Raising the Odds of Success: The Current State of Experimentation in Product Development*. Information and Software Technology. Available from: https://www.researchgate.net/publication/301599296_Raising_the_Odds_of_Success_The_Current_State_of_Experimentation_in_Product_Development. Read 24.3.2022.

Li Q., Shu F., Boehm B., Wang Q. (2010) *Improving the ROI of Software Quality Assurance Activities: An Empirical Study*. In: Münch J., Yang Y., Schäfer W. (eds) New Modeling Concepts for Today’s Software Processes. ICSP 2010. Lecture Notes in Computer Science, vol 6195. Springer, Berlin, Heidelberg. Available from: https://doi.org/10.1007/978-3-642-14347-2_31. Read 24.11.2021

Marciuska, S., Gencel, C., & Abrahamsson, P. (2013, June). *Exploring how feature usage relates to customer perceived value: A case study in a startup company*. In International Conference of Software Business (pp. 166-177). Springer, Berlin, Heidelberg.

Available from: https://link.springer.com/chapter/10.1007/978-3-642-39336-5_16. Read 29.8.2022.

Matthies, C. and Hesse, G. (2019). *Towards using Data to Inform Decisions in Agile Software Development: Views of Available Data*. In Proceedings of the 14th International Conference on Software Technologies - ICSOFT, ISBN 978-989-758-379-7; ISSN 2184-2833, pages 552-559. DOI: 10.5220/0007967905520559. Available from: https://www.scitepress.org/Link.aspx?doi=10.5220/0007967905520559. Read 4.12.2021.

Mili, A., Tchier, F. (2015). *Software testing: concepts and operations (1st edition).* John Wiley & Sons, Inc. Available from: https://andor.tuni.fi/permalink/358FIN_TAMPO/1j3mh4m/alma9910681628005973. Read 10.9.2022.

Mistrik, I. et al. (2016). *Software quality assurance: in large scale and complex software-intensive systems*. 1st edition. Ivan Mistrík (ed.). Amsterdam, [Netherlands: Morgan Kaufmann. Available from: https://learning.oreilly.com/library/view/software-quality-assurance/9780128025413/?sso_link=yes&sso_link_from=tampere-university. Read 7.8.2022.

M-Files. (2021a). *M-Files Exceeds 30 Percent Growth in Annual Recurring Revenue in 2021*. Available from: https://resources.m-files.com/press-release-en/m-files-exceeds-30-percent-growth-in-annual-recurring-revenue-in-2021. Read 13.9.2021

M-Files. (2021b). *M-Files Completes Software-as-a-Service Transition with Subscription Revenue Fueling 94 percent of its Annual Recurring Revenue in 2020*. Available from: https://resources.m-files.com/press-release-en/momentum-2020-m-files-completes-saas-transition-with-subscription-revenue-fueling-94-percent-arr. Read 13.9.2021

Nader-Rezvani, N. (2018). *An Executive's Guide to Software Quality in an Agile Organization: A Continuous Improvement Journey*. Berkeley, CA: Apress L. P. Available from: https://learning.oreilly.com/library/view/an-executives-guide/9781484237519/?sso_link=yes&sso_link_from=tampere-university. Read 5.12.2021

Najdawi, A. and Patkuri, S. K. (2021). *Modeling Business Intelligence Process: Toward Smart Data-Driven Strategies*. International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), 2021, pp. 198-202, doi: 10.1109/ICCIKE51210.2021.9410804. Available from: https://ieeexplore.ieee.org/document/9410804. Read 4.12.2021

Namvar, M., Intezari, A., & Im, G. (2021). *Sensegiving in organizations via the use of business analytics*. Inf. Technol. People, 34, 1615-1638. Available from: https://www.semanticscholar.org/paper/Sensegiving-in-organizations-via-the-use-of-Namvar-Intezari/754410a59800402faca1dff1dc8e82c9f53012e0. Read 4.12.2021

Nogeste & Walker. (2006). *Using knowledge management to revise software-testing processes*. Available from: https://andor.tuni.fi/permalink/358FIN_TAMPO/176jdvt/cdi_emerald_primary_10_1108_13665620610641283. Read 18.2.2022

Raghupathi, W. & Raghupathi, V., (2021). *Contemporary Business Analytics: An Overview. Data*, 6(8), p.86. Available at: http://dx.doi.org/10.3390/data6080086. Read 30.9.2021

Sahoo, H., & Pattnaik, S. (2020). *Importance of Agile Methodology in Software Development*. International Journal of Computer Techniques, 7(6), 5. Available from: http://www.ijctjournal.org/volume7/issue6/ijct-v7i6p7.pdf. Read 13.7.2022

Saunders, M., Lewis, P., & Thornhill, A. (2019). *Research Methods for Business Students*. Ebook. Available from: https://ebookcentral.proquest.com/lib/tampere/detail.action?docID=5774742. Read 10.9.2021

Sauvola, T., Lwakatare L., E., Karvonen, T., Kuvaja, P., Bosch, J., Oivo, M. & Olsson, H. (2015). *Towards Customer-Centric Software Development: A Multiple-Case Study*. 10.1109/SEAA.2015.63. Available from: https://www.researchgate.net/publication/290920159_Towards_Customer-Centric_Software_Development_A_Multiple-Case_Study/citation/download. Read 2.12.2021.

Seth, F.P., Mustonen-Ollila, E., Taipale, O. & Smolander, K. (2015). *Software quality construction in 11 companies: an empirical study using the grounded theory*. Software Quality Journal, vol. 23, no. 4, pp. 627-660. Available from: https://www.proquest.com/docview/1722622304?OpenUrlRefId=info:xri/sid:primo&accountid=14242. Read 16.12.2021

Singi, K., Kaulgud, V., Sharma, S., Dubash, N. & Podder, S. (2017). *Test Optimization from Release Insights: An Analytical Hierarchy Approach*. 2017 IEEE/ACM 3rd International Workshop on Rapid Continuous Software Engineering (RCoSE), 2017, pp. 16-19. Available from: https://ieeexplore.ieee.org/document/7967947. Read 15.11.2021.

Shuib, R. & Hassan, S. (2021). *Towards Adopting Software Quality Assurance in Agile Development Methodology*. Turkish Journal of Computer and Mathematics Education (TURCOMAT). 12. 2152-2157. 10.17762/turcomat.v12i3.1158. Available from: https://www.researchgate.net/publication/350797183_Towards_Adopting_Software_Quality_Assurance_in_Agile_Development_Methodology. Read 24.9.2021.

Smith, J. B. & Colgate, M. (2007). *Customer Value Creation: A Practical Framework. Journal of Marketing Theory and Practice*. Available from: https://libproxy.tuni.fi/login?url=https://www.proquest.com/scholarly-journals/customer-value-creation-practical-framework/docview/212205812/se-2?accountid=14242. Read 22.3.2022.

Wang, Y., Mäntylä, M., Liu, Z., Markkula, J., Raulamo-jurvanen, P. (2021). *Improving test automation maturity: A multivocal literature review*. Softw Test Verif Reliab.

2022;e1804. Available from: https://doi-org.libproxy.tuni.fi/10.1002/stvr.1804. Read 22.3.2022.

Yin, R.K. (2018). *Case Study Research and Applications: Design and Methods (6th edn).* London: Sage.