Leiden University

ICT in Business and the Public Sector

A comprehensive mapping of the software selection capability in terms of operational routines

Name:	Y.S. van der Vorm
Studentnr:	s1483757
Date:	August 2021
1st supervisor:	Prof.dr.ir. J.M.W. Visser
2nd supervisor:	T.D. Offerman MSc

MASTER THESIS

Leiden Institute of Advanced Computer Science
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

A comprehensive mapping of the software selection capability in terms of operational routines

August 2021

Leiden University

Abstract

Faculty of Science
Leiden Institute of Advanced Computer Science

Master of Science

A comprehensive mapping of the software selection capability in terms of operational routines

by Y.S. van der Vorm

Considering the digital transformation many organizations are in, and innovative software packages being developed at a rapid pace, it is important for organizations to be aware of software opportunities. To be able to profit from these opportunities, organizations should have high-level sensing, seizing and reconfiguring routines in place to continuously execute software selections. To find out if this is the case, this research opens up the blackbox that is the software selection capability. Using a mixed qualitative and quantitative research method, we close the gap between academic research and practice. Through fourteen interviews with experts, this research has discovered operational routines that exist within the software selection capability. This resulted in an inventory list consisting of seven main routines, 22 subroutines and 90 subsubroutines. Using this list, a survey was created to test the frequency and perceived effects of these routines on a group of 34 software selection experts. This resulted in a list of best practices based on what software selection experts believe to be the most important routines. The research also showed that software selection, regarding current capability research, is generally executed as an ordinary capability or ad hoc problem solving technique. It changes products, processes and capabilities but the high-level routines to initiate these changes are not in place. To be able to install these routines, this research supplies a three step approach on implementing software selection as a dynamic capability. Organizations could profit from the continuous execution of software selection because it can enable product leadership, operational excellence and a competitive advantage.

Acknowledgement

First of all, I would like to thank my second university supervisor, T.D. Offerman MSc. Our one hour weekly meetings often resulted in 45 minutes of discussing the latest reality television series and 15 minutes of Thesis talks. His positivism and faith in my skills motivated me to keep going, even though the last weeks sometimes felt as for ever. Furthermore, I would like to thank Prof.dr.ir. J.M.W. Visser for his experienced vision and guidance at key points during the research.

I also want to thank my colleagues at BearingPoint. Particularly Daan and Tessa, who guided me through the first, exciting months at the firm. I could reach them with any questions I had and they were always there to help. Other colleagues at BearingPoint motivated me by just talking and laughing with me. I'm looking forward to start working with all of them.

Finally, I want to thank my girlfriend Loes, who had to deal with me when this research made me grumpy, but whose love was always able to cheer me up.

Contents

Chapter 1

Introduction

Over the last decades, many organizations have been going through a huge transformation: the digital transformation. Digital technologies enable disruptions in every industry. Companies are making huge change efforts to benefit from the latest trends and to keep up with their competitors. By striving for innovation and digital new business models, value is created for customers, employees and other stakeholders. To be quicker, cheaper, more innovative and, most of all, better than their competitors, businesses need to be flexible and adaptive. More capable players constantly displace weaker players and emerging technologies replace older technologies [Miller and Yeoh, 2006]. One thing that is indispensable to the digital transformation is software. With software being developed at a rapid pace, we expect organizations to continuously be on the look out for new software opportunities.

There are three ways in which a company can get the required software support: outsource the entire process, develop the software, or procure the software. Most software which is used in organizations is procured. This is faster, requires less effort and is qualitatively better [Albert and Brownsword, 2002]. In order to buy software they go through the process of selecting software that is applicable to the organization. Despite the positive aspects of procuring software, it brings about risks and challenges like incompatibility with current systems or a lack of software support skills [Tran and Liu, 1997]. Software selection is a delicate, complex process because implementing it involves people, technology and processes. It takes multiple years and costs a huge amount of money. It is an extensive business process re-engineering practice [Yusuf et al., 2006].

1.1 Problem statement

Even though software selection is important for every organization, scientific insights into software selection as a capability are scarce. In fact, the entire software selection capability is vague. It is not clear whether software selection is even considered as a capability or if organizations see it as ad hoc problem solving. Research on how software selection is executed in practice is outdated and most studies propose methodologies and techniques for the evaluation and selection of software. Software selection entails more that those two aspects. To close the research gap between academic literature and practice, we want to know what organizations do when selecting a piece of software and which steps they perceive to be important to the successful completion of a selection. By analyzing the operational routines that exist within the software selection capability, we want to discover what the state of the capability is; operational, dynamic, or no capability at all. By opening up the blackbox of software selection, we will find out if organizations execute software selection continuously or whether they see it as an ad

hoc problem solving technique. We want to know if organizations' software selection capabilities are ready for the rapidly changing environment that is already here. This goal has resulted in the following research questions:

To what extent do organizations execute continuous software selection?

- *What are the operational routines that help organizations in their software selection capability?*
- *Which routines are perceived to be most important to the successful completion of a software selection?*
- *How can organizations implement software selection as a dynamic capability?*

1.2 Thesis outline

We will unbox the software selection capability by using the lens of organizational capabilities and routines. Therefore, literature research will be conducted on these topics. We want to know what capabilities and routines are, what types there are and how they fit in the picture of capability research. A deeper understanding of these topics is necessary before any other research can be carried out. Secondly, interviews will be conducted with key experts on software selection within organizations. By asking open questions in a semi-structured interview, useful qualitative data will be collected. The goal is to create an inventory list of routines that help organizations in their software selection capability. These routines will be tested on a bigger population using a survey. The analysis of the survey responses will be used to gain more insights in the frequency of the routines used and the importance of them to the successful execution of software selection. The perceived importance will be used to create an inventory list of best practices. These are the routines that software selection experts find most important.

Then, we will zoom out. To gain a deeper understanding of the software selection capability and to determine its current state, we need to put it in the perspective of capability research. We will see if software selection has characteristics belonging to a capability and whether organizations deploy software selection as a capability. The discovered routines, triggers and effects of software selection will show whether it is executed continuously. Applying this knowledge, this research will provide recommendations on how to implement software selection as a dynamic capability. To discover if organizations are ready to face the rapidly changing environment of today, the question whether organizations execute software selection continuously or ad hoc will be answered.

Chapter 2

Literature review

The goal of this literature review is to obtain more knowledge about the software selection capability. This knowledge will be used in the preparations of the interviews and to start the unboxing of the capability. This literature review is divided in three sections: software selection, capabilities and operational routines. We will start by looking at how software selection is described in the literature. Different approaches, process steps and other considerations are highlighted by analyzing theoretical papers and case studies. We research software selection because we want to know which routines, actors and artifacts are already identified in the literature. Secondly, we will study capability research. We need to know what a capability fundamentally is, what the difference is between an operational and a dynamic capability, and how an organization can deploy a dynamic capability. Capability research will allow us to analyze the current state of software selection. Thirdly, we will research the lens; operational routines. In order to use routines as a lens, we need to fully understand what they are composed of and how they can be identified within organizational capabilities. During this research we will identify routines used in the software selection capability and use them as a lens to look at the software selection black box. This will bridge the gap between academic research and software selection in practice.

2.1 Software selection

A substantial amount of research has been carried out on the practicalities of software selection. First of all, on the importance of it. Implementing large, complex software systems takes years, is expensive and is a major business process reengineereering practice. It is a combination of interactions between humans, processes and technology [Yusuf et al., 2006]. According to Motwani et al. [Motwani et al., 2002] implementing the right new software enhances quality, performance, flexibility and responsiveness, and lowers costs. On the other hand, failure means wasting time and a substantial amount of funds. Besides, improper software selection can also lead to ill-considered strategic decisions, resulting in economic loss [Zaidan et al., 2015]. Consequently, it is very important to select the right software. So how do organizations go about doing that? This section will describe process steps and tools we find in the literature. We will also discuss criteria that need to be taken into account when selecting software.

2.1.1 Process steps of software selection

A software selection process is triggered by a stimulus for change. After that, a selection (or acquisition) team needs to be formed. This team can consist of many roles: project managers, business

experts/managers, IT experts/managers, procurement experts, controllers, consultants and end-users [McQueen and Teh, 2000]. At least senior executives and personnel with technical and business skills should be part of the team to be successful [Wei et al., 2005]. The shaping of the project team is influenced by an organizations in house skills. All team members should have an appropriate skill set to be able to complete specific activities or handle responsibilities [Verville et al., 2005]. If these skills are not available within an organization, the service of consultants can be called in to lead the project to a successful end. When a team is formed, they can start the selection process.

In 2009, Jadhav and Sonar [Jadhav and Sonar, 2009] analyzed 27 methodologies that are stage-based and proposed a seven step methodology, based on the best practices from those methodologies, to select and purchase any type of software. After listing the stages we will dive deeper into some of the stages.

1. Identifying the exact reason why new software is necessary and what specifications this software should meet (gathering needs, requirements and selection criteria). High level investigation of availability for suitable candidates of software packages, including research into software features the vendor provides, should be conducted (e.g. [Colombo and Francalanci, 2004]). Also, analyzing of social actors and wider market forces [Howcroft et al., 2010].
2. Creating a shortlist of possible solutions that might solve the reason why new software is necessary (e.g. [Bhuta and Boehm, 2005]).
3. Reviewing the shortlist and eliminating products that do not have the required features, or fit current systems (hardware, database, operating system or other software packages) (e.g. [Bhuta and Boehm, 2005]).
4. Use an evaluation tool to score and rank the remaining options on the list. Eliminate options that rank low (e.g. [Stylianou et al., 1992]).
5. Testing trial copies of remaining software packages in an appropriate environment and performing an empirical evaluation (e.g. [Illa et al., 2000]).
6. Negotiating price, number of licenses, schedule of payment, specifications of functionalities, maintenance responsibilities, delivery time table and legal specifications (e.g. [Kontio, 1996]).
7. Selecting and purchasing the best fit, and entering the implementation phase [Jadhav and Sonar, 2009].

In Appendix 1, a more in-depth list of process steps is presented that were extracted from 39 papers between 1989 and 2015. The process steps do not necessarily occur in the order that they are listed since it varies per paper which steps are performed when. During the qualitative, exploratory research a similar table will be created from 14 interviews about processes between 2015 and 2020.

Needs, requirements and criteria

Software selection is a multi-criteria decision making process. This means that there are many criteria decision makers need to take into account when selecting the right software. If the criteria are defined incorrectly, an organization might end up with a software package that does not fit the organization. Evaluation criteria are set up after the needs and requirements have been defined. Needs are studied and translated to a set of requirements. Since the software selection process can be iterative [Verville and Halingten, 2003] these requirements can change as the project progresses. Studying needs, gathering requirements and identifying evaluation criteria overlap [Kusumo et al., 2011].

In their systematic literature review, Jadhav et al. [Jadhav and Sonar, 2009] classified the criteria discussed in the literature in seven categories:

1. Functional criteria: factors specific to a certain type of software. What should the software be able to do? What should it look like? This also involves characteristics like adaptability and interoperability.
2. Software quality criteria, also known as non-functional requirements (NFR) [Şen and Baraçlı, 2010]. The NFRs mentioned most in the literature are: usability, efficiency, maintainability, reliability and portability.
3. Vendor criteria, e.g.: training, response time and reputation.
4. Cost and benefit criteria, e.g.: licensing, maintenance and training costs, and direct and indirect benefits.
5. Technical criteria, e.g. criteria regarding the hardware, software and configuration.
6. Opinions of stakeholders, e.g.: technical opinions (IT experts, consultants, potential vendors) and non-technical opinions (end users, project managers).
7. Output criteria, e.g.: ability of the software to report, export or print output data.

No two processes are exactly the same. The industry, organization, department, stakeholders and type of software are all aspects that alter the selection criteria. That is why there is no standardized checklist to discover which software fits your problem. Each process requires stakeholders to identify specific criteria. It is important to involve stakeholders in different functions to get a selection criteria list that covers all needs.

Creating a shortlist

The next step is to find possible software solutions that are able to meet the organization's criteria. The first activity is to obtain information about possibilities. This can be done in two ways: through an active search for solutions and vendors [Keil and Tiwana, 2006, Schrödl, 2012], or by putting out a Request For Proposal. By formally making your defined criteria known, vendors will be able to make an offer if they think they can meet your criteria. Possible software solutions are put on a shortlist and will be considered for detailed evaluation. Jain et al. [Jain et al., 2008] argue that "vendor demonstrations, reference site visits and contacting existing users of the software" are a necessity when reviewing the shortlist. Just as involving end users in the process.

Evaluation and selection

The literature identifies three categories for evaluation methods within software selection: Multiple-criteria Decision Making (MCDM), artificial intelligence and integrated approaches [Hanine et al., 2016]. Within MCDM, we see four techniques that are used to simplify this complicated problem and remove bias as much as possible: Analytic Hierarchy Process (AHP, e.g. [Wei et al., 2005]), Weighed Scoring Method (WSM. e.g. [Wright, 1990]), a fuzzy based approach (e.g. [Cochran and Chen, 2005]) and the TOPSIS approach [Zaini et al., 2015]. Methods within artificial intelligence are genetic algorithms [Guo et al., 2011] and artificial neural networks [Yazgan et al., 2009]. For the third category, integrated approaches, we see a decision making application based on AHP and TOPSIS for Extract Load Transform software selection goals [Hanine et al., 2016] and the combination of CRITIC and WASPAS, which are both MCDM methods [Tuş and Adalı, 2019]. These (combinations of) methods can help a project team in scoring and ranking software packages. The techniques are too complex to be performed manually so tools have been developed that facilitate the use of these techniques and can be used to support the decision makers in selecting the right software. Apart from facilitating evaluation techniques, some of these tools support the evaluation and selection phase using libraries containing information on software packages and predetermined selection criteria. The tools facilitate the scoring, and automate the ranking of packages. Experiments are done with new MCDM methods and combinations of methods

on a regular basis. In fact, most of the literature on software selection of the last four years focuses on MCDM methodologies.These methods are highly promoted in the literature, especially AHP, but there are drawbacks as well. AHP requires organizations to put in extra efforts, in the form of many pairwise comparisons and computing consistency levels, while not assuring them of more reliable results than a simple MCDM method. Managers probably prefer to look at selection process intuitively [Asadabadi et al., 2019]. So while evaluation and selection methods are highly represented in the literature and seem to objectify the software selection project, many companies avoid employing these methods [Bernroider and Schmöllerl, 2013].

After evaluation, a list of the software packages ranked on their given scores is produced. Most of the time, the software solution with the highest score is the solution that is selected. Sometimes a sensitivity and/or price-performance analysis is done to check whether the solution with the highest score is really the best option [Wright, 1990]. The project team can either come to a decision jointly, or the team can act as consultants and the project lead makes the final decision independently. Also, the ones who make decisions about requirements are not always the same people that are involved in the final decision. The literature shows that IT and procurement experts are involved throughout the requirement gathering process but the final decision is mainly made by people on the business side [Verville and Halingten, 2003, Jain et al., 2008].

Howcroft and Light [Howcroft and Light, 2002] describe a case where an organization fails to make the best decision. In this case salesmanship takes the upper hand and overshadows the project team's efforts. Political, social and economic factors take over and satisfying top management becomes more important than finding the best fit for the organization. Another reason for failure in the software selection process is the lack of user involvement. In three of their four case studies, Verville and Halingten [Verville and Halingten, 2002] found that input from users is an important factor in the decision process. They conclude that if end users would not have been involved, the wrong package would have been selected.

2.1.2 Critical success factors

Our second research question looks at the perceived importance of the process steps in software selection. We want to know what steps experts view as most important when selecting software. Which routines in the software selection capability make it a successful capability? To find out, we need to know how success in software selection is measured. Critical success factors are mentioned in the literature and are deemed important, but they are rarely measured or reported. Examples of success factors are user-participation [Chau, 1994, Verville et al., 2005, Howcroft et al., 2010], leadership styles [Verville et al., 2005], well-structured processes [Verville et al., 2005], support from management [Kunda and Brooks, 2000] and dealing with resistance [Kunda and Brooks, 2000]. Many studies on critical success factors in software selection do not actually relate to the selection phase, but to the implementation phase. We see an opportunity here for further research.

2.2 Organizational capabilities

An organizational capability is a (collection of) routine(s) that, together with an input, creates decision options for an organization's top management to produce a particular output [Winter, 2003]. They can be seen as an organization's ability to perform its most basic functional activities [Collis, 1994]. A capability is repetitious and requires organized activity. It differs from a routine in the way that it has "a recognizable purpose expressed in terms of the significant outcomes it is supposed to enable, and that is significantly shaped by conscious decision both in its development and deployment" [Dosi et al., 2000].

Routines are building blocks of capabilities. This section will clarify what (dynamic) capabilities are and why they are important for organizations. The lens of capabilities enables us to analyze the current state of software selection. By researching characteristics of types of capabilities we can compare them to characteristics we find in the software selection capability. This allows us to identify what type of capability software selection is executed as in practice.

2.2.1 Operational capabilities

Dosi, Nelson and Winter [Dosi et al., 2000] argue that "to be capable of something is to have a generally reliable capacity to bring that thing about as a result of intended action." A generally reliable capacity consists of three things: a firms assets, an input, and the capability. To have a capability is to be able to operate and coordinate a bundle of routines [Nelson et al., 2018]. Since capabilities consist of routines, they share many of their characteristics. Capabilities are also patterned, repetitive and exist to have a certain, predictable outcome [Helfat et al., 2009]. Furthermore, capabilities are context-dependent and specific to the organization they are performed in. The difference lies mainly in size and importance. A capability entails a larger, more crucial part of organizational activity [Dosi et al., 2000, Winter, 2000]. Capabilities make or break the capacity of firms to perform activities which make them a living. An organization without capabilities will not be able to survive. Capabilities emerge from organizations looking for a solution to a problem or simply wanting to do things differently [Nelson et al., 2018]. Marketing activity, like trend research, could lead to new channels to reach a target group. Likewise, testing in a Research & Development department leads to the development of new products. Capabilities can also emerge through learning processes like performance evaluation sessions in which individuals feedback each other and share experiences [Zollo and Winter, 2002]. Organizations and their capabilities are constantly evolving. They need to stay ahead, at the same level, or at least close to the capability level of their competitors in order to survive [Winter, 2006, Schumpeter, 1934]. In seeking profitability organizations search for innovations that improve their capabilities. This involves the concept of "dynamic capabilities".

2.2.2 Dynamic capabilities

The big difference between dynamic capabilities and operational capabilities, is their purpose and intended outcome. Dynamic capabilities alter the way in which an organization makes a living by changing the product, production process, the scale, or the customers served to constitute economically significant change [Winter, 2003, Helfat and Winter, 2011]. It does this by creating, modifying, or extending the current resources and capabilities and altering the external environment [Helfat et al., 2009, Teece et al., 1997, Teece, 2007]. They develop just like operational capabilities; through learning processes, but they consist of a different type of routine as operational capabilities, high-level routines. High-level routines are aimed at learning, and repeatedly promote change from within the organization. The opposite of a dynamic capabiliy is ad hoc problem solving. This occurs on a one-off basis and entails the sensing of a problem, solving it, and not looking back or improving on the result [Winter, 1986, Winter, 2003].

Operational capabilities enable a firm to continue making a living and maintain "the same techniques on the same scale to support existing products and services for the same customer population" [Winter, 2003, Helfat and Winter, 2011]. In research, they are often referred to as zero-order capabilities. Dynamic capabilities are referred to as first order capabilities or higher [Collis, 1994, Winter, 2003]. A first order capability alters a zero-order capability. For example, new product development is a first-order capability for an organization which earns a living selling a line of products, which is the zero-level capability. It is not always easy to distinguish dynamic capabilities from operational capabilities since it

depends on the point of view of an observer whether economically significant change has been established [Helfat and Winter, 2011].

According to Teece [Teece, 2007], all dynamic capabilities consist of routines regarding sensing, seizing and reconfiguring. These are the microfoundations of dynamic capabilities. Sensing is about recognizing and shaping new opportunities. By creating, scanning, exploring and learning, organizations are able to recognize and create innovative products or markets. After an opportunity has been sensed, it needs to be seized. The opportunity needs to be built into a service, product or process. This requires investment in commercialization and development, and involves improving and maintaining a wide array of technological competences. Investments have to be made in particular designs and technologies that are expected to achieve the best result on the market. Early entry and full commitment are essential in seizing. After successful sensing and seizing, the organization is likely to grow and gain profitability. As markets and technologies change and the organization grows, being able to reconfigure originating resources and assets is the key to sustaining profitable growth. Adapting and changing routines is often necessary. Since this is costly and sudden change is not always embraced immediately, it is important to adapt structures and routines gradually. To be clear: sensing, seizing and reconfiguring are not dynamic capabilities themselves, they are organizational and managerial processes that support and enable dynamic capabilities [Helfat et al., 2009].

Two types of dynamic capabilities have been identified: dynamic managerial capabilities and dynamic technical capabilities [Tripsas, 1997]. Harreld et al. [Harreld et al., 2007] argue that the manager is the one who is responsible for the deployment of dynamic capabilities. Managers should be able to sense technological, competitive, regulatory and client based change, and act on opportunities and threats that emerge from that. In fact, many scholars find senior managers to play a crucial role in the ability of organizations to deploy dynamic capabilities [Helfat et al., 2009]. Motivation, skills, experience, and how a managers perceives his or her environment are critical aspects of understanding how and why dynamic capabilities have come into place [Adner and Helfat, 2003, Zahra et al., 2006]. The dynamic technological capability enables organizations to develop new technology and overcome the threat of radical technological change [Cohen and Levinthal, 1990]. In turn, newly developed technologies cause dynamic technical capabilities to develop. Dynamic capabilities play an important role in markets and industries that are characterized by technological change. The more an organization is focused on technological development, the more the organization needs dynamic capabilities.

2.2.2.1 Competitive advantage

Empirical research on the impact of dynamic capabilities is scarce. However, in conceptual work, dynamic capabilities are regularly linked to competitive advantage. The literature is divided on this. Teece et al. [Teece et al., 1997], Griffith and Harvey [Griffith and Harvey, 2001] and Lee et al. [Lee et al., 2002] argue that dynamic capabilities are directly linked to competitive advantage since dynamic capabilities are the creation of complex, hard to imitate combinations of resources that provide an advantage over competitors that do not have that. Other scholars found a link, but argue that it is indirect. Zott [Zott, 2003], for instance, states that dynamic capabilities alter the combination of resources, those resources affect economic performance and the improved performance provides a competitive advantage. Helfat et al. [Helfat et al., 2009], on the other hand, argue that it is possible for a dynamic capabilities to cause a competitive advantage, but not every dynamic capability necessarily does. The effect of a dynamic capability may also be negative, or only give competitive parity when the altered resources allow the firm to operate in the industry instead of outperforming competitors.

2.2.2.2 Costs

Long-term commitments to specialized resources are necessary to develop such a capability [Winter, 2003]. Dynamic capabilities usually involve a team or department of specially educated people who are dedicated to their change role full-time. Allocating resources to a dynamic capability and finding there is no opportunity for change, wastes time and money. Looking for change too aggressively has the risk of imposing additional costs while not achieving competitive value. Substantial operational, cognitive and managerial costs will be made because a manager misperceived the situation of the firm and deployed a dynamic capability that did not enhance or maintain performance. So before trying to implement a dynamic capability, an organization has to consider whether they have enough specialized resources, personnel and time available, and whether the dynamic capability is the right fit for the competitive landscape.

Ad hoc problem solving has the advantage that costs largely disappear when there is no problem to solve. Costs in problem solving only arise from opportunity costs when personnel who have alternative roles in the organization are used to solve the specific problem. The reason that dynamic capabilities can't exist like that, is the rustiness problem. To perform a skill or routine successfully, frequent exercise is necessary.

2.3 Operational routines

In order to unbox the software selection capability, we need to zoom in on operational routines. About 40 years ago, Nelson and Winter [Nelson, 2009] introduced a revolutionary theory about economic change, putting routines center-stage. This work has caused an impulse in the awareness of routines and research on it. Routines help understand how organizations work and how they change over time. They are regarded as the primary means by which an organization reaches their goals [March and Herbert, 2013, Cyert et al., 1963, Thompson, 1967, Nelson, 2009]. It is an essential part of organized work. This section will provide a literature review on what routines are. A note: this piece will exclusively be about operational routines so when we use the term 'routine', we mean operational routine.

Ever since Feldman and Pentland's [Feldman and Pentland, 2003] work on reconceptualizing the concept of routines, the process perspective has developed and gained popularity in operational routine research [Howard-Grenville and Rerup, 2016]. Their definition of routines is: ¨repetitive, recognizable patterns of interdependent organizational actions carried out by multiple actors¨. Feldman et al. [Feldman and Pentland, 2003] identify two dimensions of a routine: ostensive and performative. The ostensive dimension is the patterned description of how to do a specific task and the performative dimension involves the behavioural patterns performed by specific people at a specific time in a specific place. So they argue that a routine consists of cognitive and recurrent interaction patterns. Contrary to Nelson and Winter's [Nelson and Winter, 1982] perspective on routines, the process perspective sees routines as dynamic and generative [Feldman, 2000, Feldman and Pentland, 2003, Rerup and Feldman, 2011]. It emphasizes the ability of people and artifacts to enable routines to emerge, persist and change. This subsection will highlight process perspective focused theories on how routines evolve and emerge, and how artifacts play an important role in that.

2.3.1 Actors

People (or actors) do not perform routines identically. This is called performance variability. The actor's performance is shaped by power and subjectivity. Sometimes an individual performance is idiosyncratic and has no impact on the routine but sometimes it triggers changes to a routine. Many

routines are the subject of performance variability. We can see that the performance variability enables work achievements entrusted to routines. Turner and Rindova's study [Turner and Rindova, 2012] for example, shows garbage collectors that generated different patterns of garbage collection routines. A pattern when consistency is required and a pattern for when flexibility is required. This enables the organization to adjust to the situation. Skillful actors are able to understand and generate variations of the routine based on context.

2.3.2 Artifacts

Artifacts are, next to the ostensive and performative dimension, a building block of a routine. First of all, they are an important part of many routines' ostensive dimension [Feldman and Pentland, 2003]. An artifact is some sort of documentation or supportive non-human system that is related to the routine. This could be a manual or standard procedure, but also a tool, like software or a machine. However, artifacts are not the only part of the ostensive dimension. It also comprises the subjective and tacit understanding of abstract rules and norms by multiple actors. That is why Pentland and Feldman [Pentland and Feldman, 2008] emphasize the fact that routine artifacts are not the same as the routine itself. Creating artifacts to design a routine, like a checklist or manual, is not sufficient and will not generate the desired patterns of action. The performative dimension plays a big role as well. According to D'Adderio [D'Adderio, 2011], artifacts can influence the emergence and persistence of routines, in the way that they destabilize routines, or are the glue that holds these patterns together. Routines can evolve when delegating human actions to non-human actions, for example automating a process. The non-human artifacts can be further developed by human actors which causes routines to keep evolving. As can be seen in figure 2.1:

Artifacts influence and support the human actors in a routine. Research shows how they enable and alter interactions between human actors [Bapuji et al., 2012, Turner and Rindova, 2012]. Another study looks at technological development and the impact it has on routines. New technologies create new artifacts, which causes routines to develop [Labatut et al., 2012].

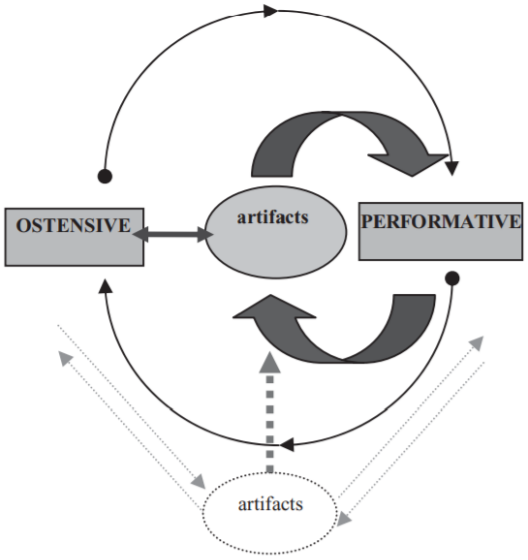

FIGURE 2.1: The building blocks of a routine (D'Adderio [D'Adderio, 2011])

2.3.3 Emergence

Parts of the ostensive dimension can be created through artifacts but how does an entire routine come about? Feldman and Pentman [Feldman and Pentland, 2003] find routines to be a product of action that occurs in the structure of modern organizations. The constraining and enabling nature of these structures shape routines. The first way in which a routine can emerge is when multiple lines of individual actions are combined into recognizable patterns through role taking [Dionysiou and Tsoukas, 2013]. Team members naturally take up a role in a team and start to interact with each other in different ways. Miller et al. [Miller et al., 2014] found that individual actors search for and/or memorize what other actors do. Using their transactive memory, actors try to copy process steps and find the best possible combination of these steps. Eventually, this gives shape to a new routine. By sharing knowledge in communities, organizations are seeking to improve collaborations with each other [Swinglehurst et al., 2010, Ribes and Bowker, 2009] which enables the emergence of routines. Finally, routines can originate by creating or adding artifacts to the organization [Bapuji et al., 2012] and the emergence of new technologies [Labatut et al., 2012].

2.4 Overview

Now that we have fully researched operational routines and capabilities, we can zoom out and take a look at the bigger picture. Salvato and Rerup [Salvato and Rerup, 2011] captured the organization structure of collective entities in figure 2.2.

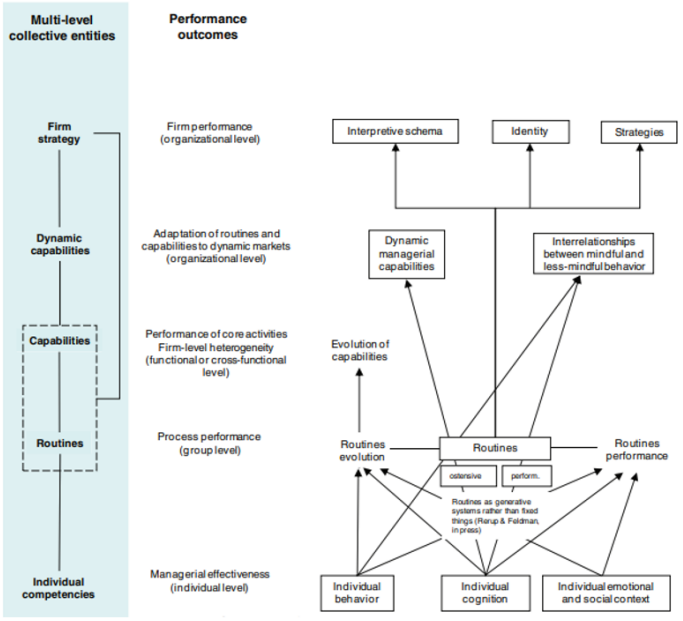

FIGURE 2.2: Organization structure of collective entities (Salvato and Rerup [Salvato and Rerup, 2011])

Since this research focuses on routines and capabilities, it is unnecessary to dedicate sections to "Individual competencies" and "Firm strategy" but they are elaborated in short here. Competencies are critical in determining performance and consist out of measurable clusters of KSAs (Knowledge, social skills and aptitude) [Aguinis, 2009]. Individual behavior, individual cognition, and individual emotional and social context, are components of individual competencies and, together with artifacts, are what routines are comprised of [Simon, 1957]. Artifacts are missing in this diagram but they influence and

support the interaction of individual competencies and are at the center of routine evolution. One level up we see routines and its two dimensions: the ostensive and performative dimension. Performance and evolution are key factors that drive organizational change. The evolution of routines leads to the evolution of capabilities, which can be seen as the performance of core activities. Capabilities consist of multiple routines and therefore consist of many of the same characteristics as routines. However, a capability expresses an intended, recognizable purpose and conscious decision-making shapes it. Furthermore, a capability comprises a bigger and more crucial part of organizational activity. Individual cognition of top managers lead to the deployment of dynamic capabilities. By creating high-level routines, dynamic capabilities alter operational routines, capabilities and resources to gain a competitive advantage, or at least keep up with competition in dynamic markets. (Dynamic) capabilities and routines are in place to execute the firms strategy.

When we start mapping routines and using them as a lens to look at the software selection capability, we need to identify the ostensive part and performative part of each routine, and need to know which actors and artifacts are involved. Mapping out the routines in detail will help us open the black box, and determine whether organizations deploy software selection as an ad hoc problem solving technique, operational capability or dynamic capability.

Chapter 3

Methodology

3.1 Approach

The goal of this research is to open up the black box surrounding the software selection capability. To do this, we apply the lens of operational routines. First of all, the study aims to explore which operational routines are the building blocks of the software selection capability. We will try to gain a complete view of the routines by asking what types of roles (actors), and tools and documentation (artifacts) are used in the process. This research will try to describe, interpret and contextualize operational routines in order to gain in-depth insights. We want to produce knowledge about human behaviour, social structures and documentation within the software selection capability. In the following sections we will describe methodologies that can be used to discover routines, the methodologies used in this research and the approach to data collection.

3.1.1 Methodology

There are multiple ways in which one can collect data from organizations on their routines: look at internal data and system logfiles, conduct interviews, conduct surveys, and make observations. This will be a mixed research, focusing on two methods: conducting interviews and surveys. The Literature Review lacked routines in software selection in recent years. The interviews will be conducted to explore which routines are used in practice. By asking open questions in a semi-structured interview, qualitative data will be collected. This data will be tested on a broader audience using a confirmatory survey. The survey will also be aimed at researching other characteristics of the software selection capability like triggers and perceived effects. The result of mixing these methods is a high validity and a complete picture of the routines that are executed in the software selection capability.

3.1.2 Data collection

This section will elaborate on how and which data will be collected using the research methods described earlier. For both the qualitative and the quantitative research the population, distribution and structure will be presented.

3.1.2.1 Qualitative research

The interviews will be conducted with experts in different dimensions of the software selection capability and will be exploratory. The population for the qualitative research is described as follows: people who are involved or have been involved in software selection projects in the last two years. We aim to conduct fourteen one-hour interviews with participants of different organizations. This will result in a wide array of routines from different industries and roles. The organizations that will be approached in this research will be contacted via the networks of Van der Vorm, Offerman and Visser. Through the snowball effect we hope to reach sufficient suitable interview candidates.

During the interviews we will ask about routines that the interviewee sees in software selection projects at their organization. First, we will ask for an overview of the project stages, which will be drawn out during the interview by the interviewer. After that, we will dive deeper into each stage. The exploratory questions are based on the process steps that were discovered in the Literature Review (Appendix 1) and can be found in the Appendix. The goal is to gain a detailed view on routines that organizations have in their software selection capability. The interviews will be transcribed and the routines will be distilled and stored in a list per participant. These lists will be checked with the interviewees on completeness and correctness. After each list is checked on validity, an inventory list can be made containing all the routines that were mentioned in the interviews. This list will be used as an input for the survey.

3.1.2.2 Quantitative research

The survey will be confirmatory. Again, we will be looking for experts in different dimensions of the software selection capability. The population is described as: people who are involved or have been involved in a software selection project in the last five years. We choose to look for respondents within five years instead of two years because we want to reach more respondents than during the interviews. We aim for 100 respondents from different industries and roles. Similar to the interview candidates we will approach organizations via the networks, channels and communities of Van der Vorm, Offerman and Visser. Through LinkedIn, personal messages and snowballing we will distribute the survey to a broad audience.

The survey consists of questions in the following categories:

- Respondent characteristics: background of the respondent.
- Project characteristics: software selection projects in general. This includes asking about actors.
- Capability level: capability level based on the People capability maturity model as seen in figure 3.1
- Reasons: triggers for software selection.
- Effect: effects of software selection on business operations.
- Routines: frequency of routines and their importance to the successful completion of a software selection process.
- Artifacts: use of methodologies, tools and documentation.

The goal is to find out more about the characteristics of the capability, and the frequency, perceived importance, actors and artifacts of the routines discovered in the qualitative research. The data will be filtered on completion and missing values before being analyzed using statistics software. The survey questions can be found in the Appendix.

During the interviews and surveys we will refer to “operational routines” as “process steps”, since an “operational routine” is an academic term which is not widely known in the non-academic world.

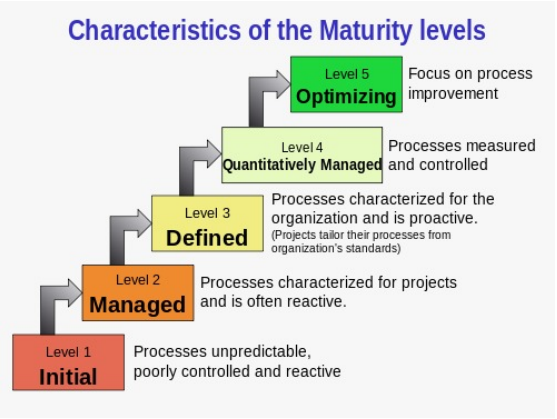

FIGURE 3.1: Maturity model by Curtis et al. [Curtis et al., 2009]

3.1.2.3 Tools

Due to the corona crisis and the subsequent social distancing, the interviews can not be conducted in person. Instead, the video call-application Microsoft Teams will be used as a tool to facilitate the interviews. This application allows participants in the video call to record the meeting. Using the recording functionality, the transcribing of the interview can be done after the interview instead of during the interview. This allows the interviewer to pay undivided attention to the interview itself. Besides MS Teams, ActivePresenter will be used as a tool to record the screen during the interviews. In case something goes wrong with one recording, there will be another. Also during the interview, the tool Excalidraw will be used to map the project phases the interviewee mentions. The transcribing will be done in oTranscribe. The transcribed interviews will be stored in Microsoft Word and the resulting inventory list will be stored in Microsoft Excel. Finally, we will use Qualtrics to create and send out the survey and IBM SPSS software to analyze the results. Excel is used to visualize the data.

3.1.2.4 Ethics

Participation of respondents in the research is completely voluntary and respondents have the right to withdraw from the research if they wish to do so. All the gathered data will be anonymized and processed via software and hardware approved by the University of Leiden to ensure the protection of privacy and anonymity. We will use the Leiden University consent form to ask consent for the use of data collected in the interviews, surveys and internal documentation. We will ask for consent again at the start of the interview and survey. This way we have full consent from the respondent before the research starts.

3.2 Procedure

This section discusses the research phases. Arguing why certain choices were made, and looking at what went well, what went wrong and how those difficulties were dealt with. This section will show that this research was made as rigorous as possible.

3.2.1 Qualitative research

There were no troubles in gathering interview candidates for this research. Through snowballing, the goal of fourteen candidates was quickly reached. All candidates are Dutch but work within different organizations, in different functions, in different industries, so they are well spread. Before planning the interviews we talked about interviewing multiple people from the same organization but with different functions. This way we could shed a light on the differences between how people look at, and execute the routines within the same organization, giving us a more in-depth view of the routines. By interviewing more people from different organizations instead of the same organization, the list of routines became more extensive but less in-depth. The trade-off between extensive and in-depth was a hard one to make but since the approach of the interviews was to explore which routines exist, we thought going for an extensive list would be more applicable to this research.

All interviews fitted within the one hour time frame. At some of the first interviews it was difficult to keep the interviewees to the point if they would keep talking. One interview could not go as in-depth on the routines as preferred because of it. In the interviews after, the interviewer put the emphasis more on the routines and digressions were interrupted more quickly. This caused the interviews to be more to the point. All the interviews are an added value to this research so there were no failures. Some interviewees wanted to use Google Meet instead of Microsoft Teams. Since it is also possible to record the conversation in Google Meet, this was no problem. One issue with MS Teams was that recorded conversations could not be downloaded using a Leiden University account. This meant it could not be played at a slower speed, making transcribing difficult. Therefore, the exportable ActivePresenter full-screen recording was used as a main source and the Teams recording was kept as a back-up. Finally, using Excalidraw to draw the software selection project phases during the interviews was effective. It made it easier to go through the phases step by step after drawing them out.

After transcribing the interviews word for word, process steps could be distilled from the interviews. We were not able to use the project phases drawn in Excalidraw because most interviewees do not use standard project phases in their software selection. This caused the phases to vary in detail per candidate. One candidate would call something a phase, while another would refer to it as a step within a phase. Even though it was useful during the interviews, it was not an added value in the organizing of the data. Everything that was drawn was also said by the candidate, so the data was extracted from the transcription instead of the drawings. By thoroughly going through the transcriptions, a list of detailed process steps per interview was distilled in Excel. This Excel sheet was sent to the candidate to check for correctness and completeness and most of them replied with some commentary. This commentary was included in finalizing the lists. The independent lists of routines were then combined to create a complete inventory list of all the routines mentioned in the interviews.

3.2.2 Quantitative research

The first week of promoting the survey resulted in 28 respondents.The survey was widely spread on LinkedIn resulting in 1400 views in one week. The survey was also posted on internal channels from BearingPoint, due to an internship, and clients of BearingPoint. The main driver of getting respondents was sending people personal messages. All fourteen interviewees were approached and most of them completed the survey and shared it. Other people within Offerman's and Van der Vorm's network who we knew were familiar with software selection were also messaged. Despite the effort we were far off the goal of 100 respondents. After emailing the CIO Platform Nederland they decided to put the survey in their newsletter to 130 CIOs. However, the number of respondents did not significantly rise after that. Reasons for not reaching the goal of 100 respondents could be the specificity of the population. There are not many people that completed software selection projects in the last five years and were reached by the survey. Another reason is the length of the survey, which was fifteen minutes.

The final size of respondents used for analysis is 34. These responses came out of 59 people starting the survey, resulting in a completion rate of 58%. Due to the nature of this survey, there is no information about the non-respondents, therefore there is no input for the non-response bias, so this could not be determined. The total of 34 respondents is not enough to find significant correlations in the data set. That is why this research focuses on descriptive analytics. Further research will have to be conducted to find significant relationships between variables.

Chapter 4

Results

The results of the mixed qualitative and quantitative research are presented in this chapter. The first section shows the general characteristics of the respondents and of the software selection projects they executed. The second section zooms in on the discovered routines, their frequency of occurrence and perceived importance to the successful completion of a software selection. In the final section we will look into results regarding software selection as a capability.

4.1 General characteristics

This subsection describes the general characteristics of the samples of this mixed research. Table 4.1 shows the characteristics of the interview participants. We see that relatively many participants work in Professional services (29%), followed by Public services (21%) and Retail and Government (both 14%). The three participants that work in Public services work at the same organization. This also counts for the two participants that work in the Government sector. During the interviews, the participant was asked to keep one project in mind when drawing out the process steps. Some participants have executed software selections in different roles but the role presented here is the role that the participants have taken on during that one project. We see that most participants performed the software selection from a Budget owner role (36%), followed by Procurement (14%) and External consultant (14%). Finally, 50% of the interview participants have executed 21 or more software selections, 21% did between 10 and 20 and 29% have been involved in 5 to 10 software selections.

Industry	% Respondents
Professional services	29%
Public services	21%
Retail	14%
Government	14%
Manufacturing	7%
Software	7%
Other	7%

Role	% Respondents
Budget owner	36%
Procurement	14%
External consultant	14%
Solution architect	7%
Enterprise architect	7%
Internal consultant	7%
Quality assurance	7%
Supplier	7%

Nr. projects	% Respondents
5 - 10	29%
10 - 20	21%
21+	50%

TABLE 4.1: % of interview participants per industry (left). % of participants per role (middle) and % of participants per number of projects executed

Of the respondents to the quantitative research, a relatively large part works in the Professional services industry as well (35%). Other industries represented in the results are Public Services (9%), Manufacturing (9%), Government (9%) and the rest is evenly spread over other industries, as can be seen in table 4.2.

Since it is possible for stakeholders in a software selection project to take on multiple roles, we see, in table 4.2, per role what percentage of respondents fulfilled that role. To the contrary of the qualitative research, respondents could give multiple answers to this question. Most respondents were involved (partly) as IT representative (38%), followed by Budget holder (32%), Project manager (32%) and External consultant (32%). Of the respondents, 38% fulfilled one role, 12% fulfilled two roles, 32% fulfilled three or four roles and 18% fulfilled more than four roles. The maximum of roles per respondent is nine.

Industry	% Respondents
Professional services	35%
Other	15%
Public services	9%
Manufacturing	9%
Government	9%
Financial services	6%
Education	3%
Healthcare	3%
Media & Entertainment	3%
Retail	3%
Software	3%
Telecom	3%

Role	% Respondents
IT representative	38%
Budget holder	32%
Project manager	32%
External consultant	32%
Solution architect	21%
Process expert	18%
Problem owner	15%
Business representative	15%
Security expert	12%
Data expert	12%
Internal consultant	12%
Procurement officer	9%
(Key) user	9%
Enterprise architect	6%
Supplier	6%
Risk expert	3%

TABLE 4.2: % of respondents per industry (left) and % of respondents per role (right)

Another characteristic is experience, measured in the number of software selection projects that the taken sample has executed. Four participants executed just one or two projects (12%), eight executed three to five projects (24%), seven executed six to ten projects (20%) and eleven to twenty projects (20%) and eight executed 21 projects or more (24%). Of the participants 31 are currently involved in a software selection project or executed their last software selection project a year ago (91%). The remaining respondents executed their last project respectively two and three years ago.

Team size	% Respondents
1	0%
2 - 3	6%
4 - 6	50%
7 - 10	29%
11 - 14	6%
15+	9%

External consultants/employees	% Respondents
0	30%
1 - 3	70%
4 - 6	0%
7 - 9	0%
10+	0%

Duration in months	% Respondents
1	3%
2	12%
3	38%
4	12%
5	3%
6	15%
8	6%
9	3%
12	9%

TCO	% Respondents
No idea	6%
€0 - €10,000	3%
€10,001 - €50,000	3%
€50,001 - €200,000	30%
€200,001 - €500,000	15%
€500,001 - €1,000,000	9%
€1,000,001 - €5,000,000	21%
€5,000,001 - €10,000,000	6%
€10,000,001 - €20,000,000	3%
€20,000,001+	3%

TABLE 4.3: From left to right: % of respondents per team size, % of respondents per number of external consultants/employees, % of respondents per project duration and % of respondents per TCO

The survey tested the typical project characteristics that respondents recognize in their projects. Table 4.3 shows the results for team size, external consultants involved, project duration in months and TCO. A typical software selection project is mostly done in a team of four to six people (50%) but we also see slightly larger teams of seven to ten people (29%) regularly. Most respondents typically see one to three external consultants/employees involved in their team (70%) and the rest typically sees none involved (30%). Projects typically take around three months but can also take six months and in the longest cases twelve months. Finally, the typical TCO lies for 75% between €50,001 and €5,000,000 but there are cases of €0 to €10,000 and €20,000,001+ as well.

4.2 Software selection routines

After looking at the general characteristics of the qualitative and quantitative research, we can focus on the software selection routines. These routines were discovered during the qualitative research. The interviews resulted in an inventory list of 222 process steps. These steps were discussed in the interviews and validated afterwards. After critically assessing the list, unique steps were combined to form a shorter, more usable list. This list consists of 7 unique routines, 22 subroutines and 90 subsubroutines and can be found in Appendix 2. The scoped list gives a clear view of the routines, subroutines and subsubroutines in the software selection capability. The subsubroutines are the steps mentioned by the interview candidates. The routines and subroutines columns were created to classify and order the mentioned subsubroutines. The routines and subroutines are not direct results from the interviews. After finishing the qualitative research, the subroutines were used in the survey to discover the frequency at which they occur in software selection projects. The respondents were also asked to rate the importance of the subroutines to the successful completion of a software selection project on a Likert scale. The discovered main routines are:

1. Initiate project
2. Gather requirements
3. Research the market
4. Pre-select vendors
5. Select vendor
6. Conduct pilot
7. Negotiate contract

Each routine consists of two, three or four subroutines. The following sections will zoom in on the discovered routines. Each section consists of a table with the routine and corresponding subroutines and subsubroutines, followed by two figures concerning the frequencies and perceived importance of the subroutines in each routine. An elaboration of the routine, subroutine and subsubroutines is found after that.

4.2.1 Initiate project

The first routine, "Initiate project", consists of three subroutines: "Identify need", "Make reuse, make or buy decision" and "Set up project organization". Table 4.4 shows the routine with its subroutines and subsubroutines.

1. Initiate project		
1.1. Identify need	**1.2. Make reuse, make or buy decision**	**1.3. Set up project organization**
1.1.1. Recognize arising need from the business	1.2.1. Check for in-house solution	1.3.1. Involve project manager
1.1.2. Create base of support	1.2.2. Consider process reengineering solutions	1.3.2. Write first version business case
1.1.3. Translate need to IT	1.2.3. Consider developing software	1.3.3. Involve right stakeholders in team
	1.2.4. Decide to buy software	1.3.4. Hire consultants
		1.3.5. Plan project
		1.3.6. Decide procurement strategy
		1.3.7. Kick off project

TABLE 4.4: "Initiate project" routine, subroutines and subsubroutines

Figure 4.1 and 4.2 show the frequency and perceived importance of the subroutines in "Initiate project".

Interviewee number thirteen, a sourcing consultant, says that software selection "starts with the vision, strategy, or objectives of the organization." They result in a need arising from the business and the triggering of "Identify need". The business can be any department within the organization.

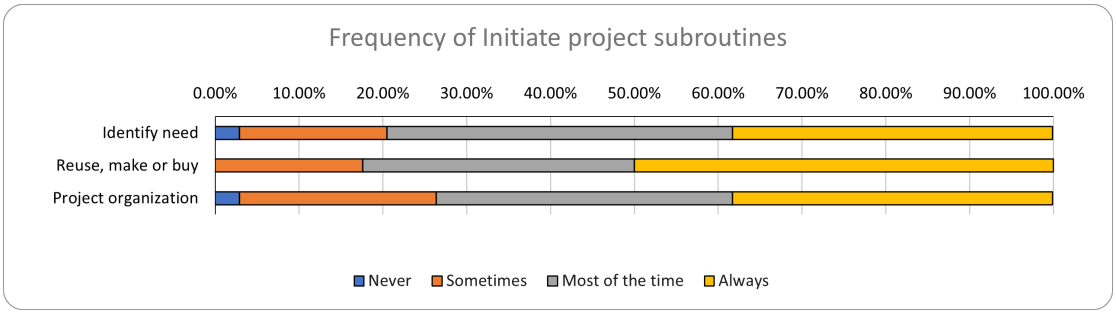

FIGURE 4.1: Frequency of Initiate project subroutines

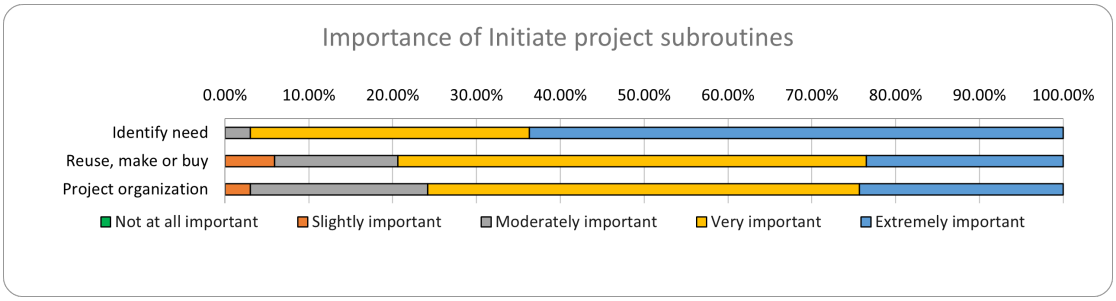

FIGURE 4.2: Importance of Initiate project subroutines

Usually when a trigger occurs, organizations first analyze the trigger. It might be the case that the software is not the problem or the need, but something else is. Faulty data or an organizational issue can be a problem masked as a software problem. A common mistake in software selection is that the business decides they need new software before analyzing the problem they have (interview 8, 12). Other common mistakes are when the business creates a need based on software they want (interview 2) or they purchase a software package without consulting IT at all (interview 5). Therefore, an important subsubroutine is "Translate need to IT", to check whether the need from the business truly is something the business needs. "Identify need" is perceived to be the most important subroutine of all as 97% of respondents find it very important or extremely important to the successful completion of a project.

After translating a need to IT, an organization can execute the "Make reuse, make or buy decision" subroutine. Half of people who do software selections always make this consideration. If the necessary piece of software is strategic and an organization might want to distinguish themselves from other organizations, they can choose to develop the software. Most of the time, the decision is made to buy. Then, according to interviewee number 12, a Global IT Director: "You have to build a project organization, a governance, to stay in control from start to finish." Having clear roles and responsibilities will give an organization a better picture of where mistakes were made and how to improve on them in the future. Information manager and interviewee number 10 said that no clear roles and responsibilities due to a lack of organization is something that happens in their software selections. That is why during this subroutine a project manager is appointed (in 59% of the time), external consultants are hired (in 70% of the time) and the right stakeholders are involved. Then, a first version business case is written which will be updated throughout the process. The procurement department makes the decision on a procurement strategy and a project planning/scope is created. The "Initiate project" routine ends with the execution of the "Kick off project" subsubroutine.

4.2.2 Gather requirements

The "Kick off project" subsubroutine, triggers "Gather requirements". This routine consists of three subroutines: "Analyze impact of new software", "Draw up requirements" and "Specify requirements". The subsubroutines are presented in table 4.5.

2. Gather requirements		
2.1. Analyze impact of software	**2.2. Draw up requirements**	**2.3. Specify requirements**
2.1.1. Make a Business Impact Analysis	2.2.1. Gather business/IT requirements	2.3.1. Distill proofpoints
2.1.2. Make a Privacy Impact Analysis	2.2.2. Collect specialists' requirements	2.3.2. Define knock-out criteria
	2.2.3. Draw up requirements	2.3.3. Define acceptance criteria
	2.2.4. Store requirements in one document	2.3.4. Prioritize requirements
		2.3.5. Determine scoring mechanism
		2.3.6. Refine business case

TABLE 4.5: "Gather requirements" routine, subroutines and subsubroutines

Figure 4.3 shows the frequency of the subroutines within "Gather requirements" and 4.4 shows the perceived importance.

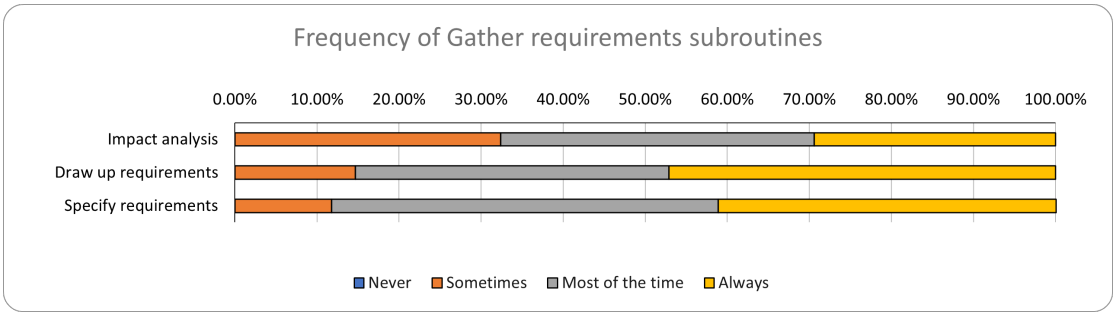

FIGURE 4.3: Frequency of Gather requirements subroutines

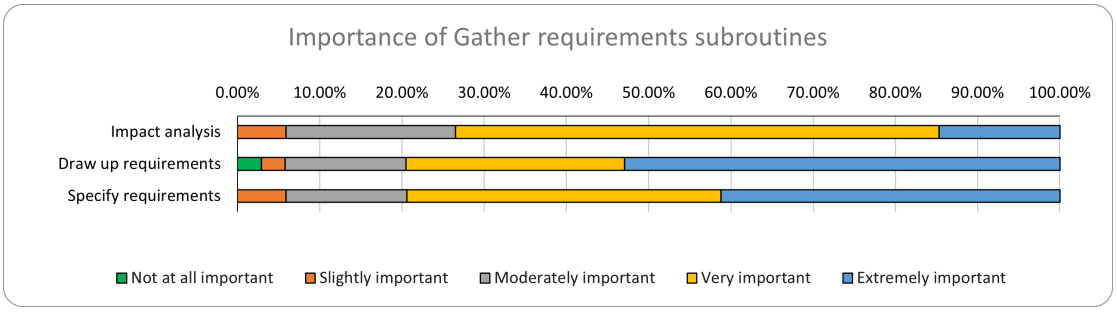

FIGURE 4.4: Importance of Gather requirements subroutines

Overall, "Draw up requirements" and "Specify requirements" are executed significantly more often than "Analyze impact of new software" while they are perceived to be just a little more important. Interviewee number 8, CIO of a food processing solutions company said: "What we do insufficiently, is doing a business analysis. What is the problem? Is this solution going to solve the problem?" A Business Impact Analysis and Privacy Impact Analysis make sure that teams know which business processes are affected by the new software, and consequently know which users to involve in the "Draw up requirements" subroutine. This positively influences the completeness of the requirements.

When it is known what business processes will be affected by the to be selected software, a software selection team can start gathering requirements from the right stakeholders. Requirements are gathered in the following categories: functional, non-functional (performance), technical, security, privacy, vendor, communications and archiving. They are gathered through workshops, brainstorms, interviews, conversations and via email, and are stored in text processors, spreadsheets or a Program of Requirements

if they are made in the form of a list. Requirements can also be stored in the form of user stories or use cases. No matter how they are stored, it is important that all the requirements are gathered in one document and are not scattered over the organization. Interviewee number seven, sourcing specialist, said: "The more that is done at the front of the process and the clearer the specifications, the faster the process goes." There are two aspects of this routine that are sometimes executed differently than they should: the business is set on acquiring a certain piece of software instead of taking the time to gather requirements (interview 2, 5, 6, 7) and requirements focus too much on functionality of the tool instead of how it fits within the organization (interview 3, 11).

So, boarding up the requirements at the start of the project speeds up the rest of the project. One way to specify the requirements is by creating proofpoints in the form of user stories. These proofpoints are considered to be the "Must have", requirements that vendors must meet in order to have a chance of winning. Another way is to split up the requirements using the MoSCoW prioritization: " Must haves", "Should haves", "Could haves" and "Would haves". A simpler version of this that organizations use, is splitting requirements up in "Must haves" and "Nice to haves". Organizations sometimes choose to define knock-out criteria and/or acceptance criteria and create a scoring mechanism to use when executing the "Pre-select vendors" and/or "Select vendor" routines. This scoring mechanism includes assigning weights to requirements, like in the WSM technique.

4.2.3 Research the market

The third routine is "Research the market". It consists of three subroutines: "Conduct market research", "Request vendor information" and "Create longlist". This routine is generally executed (partly) in parallel with the "Gather requirements" routine. The subsubroutines are presented in table 4.6.

3. Research the market		
3.1. Conduct market research	**3.2. Request vendor information**	**3.3. Create longlist**
3.1.1. Browse the internet	3.2.1. Organize Q&As	3.3.1. Classify options in Kraljix matrix
3.1.2. Go to events	3.2.2. Send out RFI	3.3.2. Create longlist
3.1.3. Converse with peers	3.2.3. Conduct reference visits	
3.1.4. Consult software experts	3.2.4. Post tender on tender publication website	

TABLE 4.6: "Research the market" routine, subroutines and subsubroutines

Figure 4.5 and 4.6 show the frequencies and perceived importance of the "Research the market" subroutines.

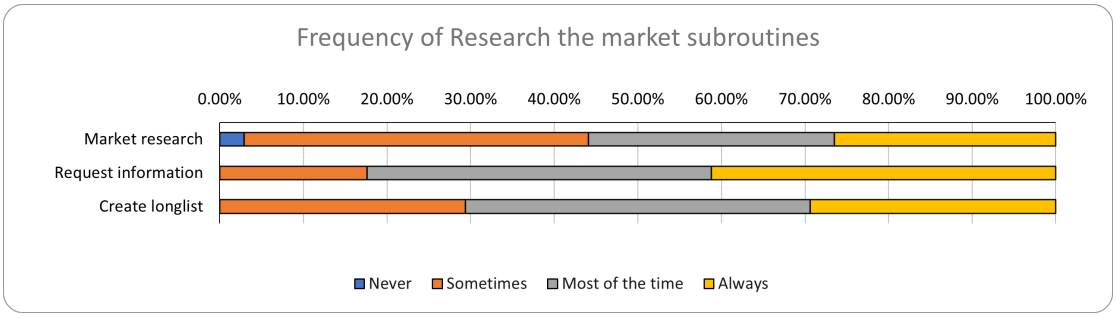

FIGURE 4.5: Frequency of Research the market subroutines

Organizations are try to keep up with the latest trends and learn from one another. The "Research the market" routine is about knowing which solutions are on the market. Interviewee number four, senior manager at a consultancy firm said: "We will look up information, retrieve it, have conversations, possibly reference visits, can also be a part. By talking to other parties who have gone through a similar process. That gives you a better picture." One subsubroutine that came up in almost all interviews is

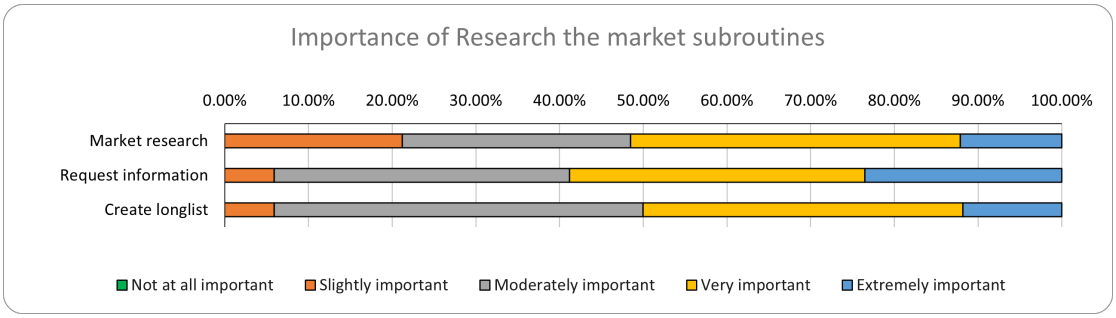

FIGURE 4.6: Importance of Research the market subroutines

"Consult software experts". Many organizations of interview participants are a client of research and advisory companies in the field of software. Research companies can be consulted during a software selection process if an organization wants to learn more about players in the market and their software packages. Less time and effort needs to be put in researching possible solutions and the analyses of these research agencies are extensive.

Government agencies are obliged by law to go through a public tender process. This means that they post a tender (detailed RFP with extensive requirements) on a tender publication website and software vendors can reply to the request with a proposal. It is not necessary to do a market orientation since software vendors that are able to fulfill the needs will approach that government agency. Commercial organizations can also choose to do this but it take more time in later routines than when you pick and choose a longlist, since more vendors will want to compete for the deal.

After executing "Conduct market research" and "Request vendor information", "Create a longlist" can be executed. Before creating the longlist, organizations can classify the optional solutions in a Kraljic matrix [Caniels and Gelderman, 2005]. In the Kraljic matrix products are classified on profit impact and supply risk. It depends on the goal of the software what category of software you need: a leverage item, strategic item, non-critical item or bottleneck item. If a strategic item is necessary an organization will know to continue with the vendors in that category and put them on the longlist. A longlist typically consists of ten software vendors.

4.2.4 Pre-select vendors

This routine consists of four subroutines: "Conduct additional research", "Enter RFx process", "Assess vendors" and "Create shortlist". The longlist created in "Research the market" is an input for the start of this routine. The subsubroutines are presented in table 4.7.

4. Pre-select vendors			
4.1. Conduct additional research	**4.2. Enter RFx process**	**4.3. Assess vendors**	**4.4. Create shortlist**
4.1.1. Research vendors	4.2.1. Send out RFx to longlisted vendors	4.3.1. Assess offers	4.4.1. Rank vendors
4.1.2. Talk to peers		4.3.2. Assess vendors	4.4.2. Consultants advise project team
4.1.3. Organize Q&As		4.3.3. Assess solutions	4.4.3. Discuss offers/assessments
4.1.4. Organize vendor presentations/demonstrations		4.3.4. Score individually	4.4.4. Create shortlist

TABLE 4.7: "Pre-select vendors" routine, subroutines and subsubroutines

The frequency of subroutines within "Pre-select vendors" can be seen in figure 4.7 and the perceived importance in figure 4.8.

"Conduct additional research" is comparable to the "Conduct market research" subroutine but is focused on a smaller number of vendors, the vendors on the longlist. Subsubroutines like "Research vendors", "Talk to peers" and "Organize Q&As with longlisted vendors" are practically the same subsubroutines. A difference is that "Organize Q&As with vendors" is often combined with "Organize vendor

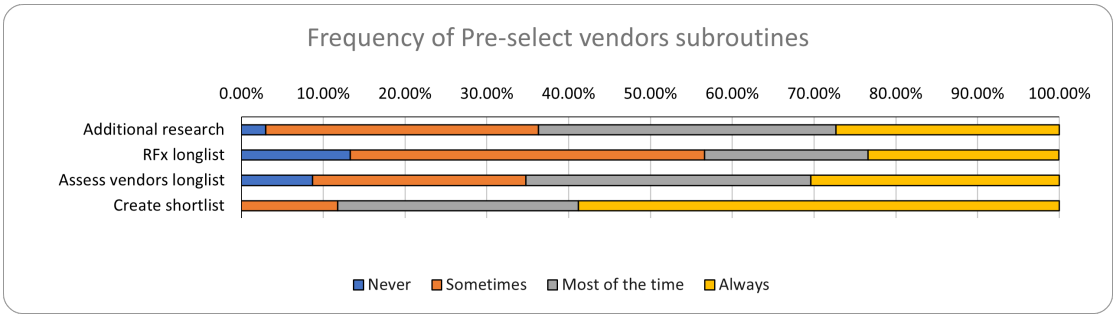

FIGURE 4.7: Frequency of Pre-select vendors subroutines

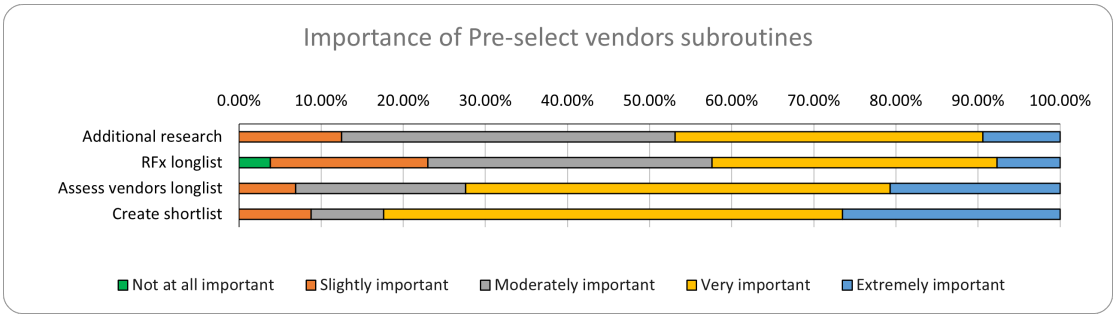

FIGURE 4.8: Importance of Pre-select vendors subroutines

presentations/demos". These presentations can be organized in different ways. We see organizations that organize one day and all vendors come by at once, and there are organizations that invite vendors over one by one. The "Enter RFx process" subroutine consists of one subsubroutine: "Send out RFx to longlist". It is executed relatively infrequently and is perceived to be relatively unimportant.

On the contrary, the "Assess vendors" subroutine is considered to be relatively important. The way in which vendors on the longlist are assessed differs per organization. Some organizations choose to do a high-level assessment, looking at "Must haves" and/or applying knock-out criteria to drop vendors. Other organizations score the vendors, their solutions, presentation and/or their offers on all set requirements. In both cases the stakeholders individually score, or tick-off requirements. "Create shortlist" is executed significantly more than "Create longlist". Interviewee number two, an Enterprise architect, said that due to time constraints the longlist and entire "Pre-select vendors" routine are sometimes skipped.

4.2.5 Select vendor

The shortlist created in "Pre-select vendors" is used as an input for the start of "Select vendor". The four discovered subroutines: "Enter RFx process", "Interact with vendors", "Assess vendors" and "Make a preliminary selection" are presented in table 4.8.

5. Select vendor			
5.1. Enter RFx process	**5.2. Interact with vendors**	**5.3. Assess vendors**	**5.4. Make a preliminary selection**
5.1.1. Send out RFx to shortlisted vendors	5.2.1. Organize QAs	5.3.1. Assess offers	5.4.1. Advise stakeholders
	5.2.2. Brief vendors about expectations	5.3.2. Assess vendors	5.4.2. Advise budget holder
	5.2.3. Organize vendor presentations/demos	5.3.3. Assess solutions	5.4.3. Approve selection
	5.2.4. Organize vendor pitches	5.3.4. Assess implementation plan	5.4.4. Inform vendors
	5.2.5. Have vendors create a prototype	5.3.5. Score individually	5.4.5. Continue with one or two best vendors
		5.3.6. Make reference visits	
		5.3.7. Find a consensus with stakehodlers	
		5.3.8. Rank vendors	
		5.3.9. Finalize business case	

TABLE 4.8: "Select vendor" routine, subroutines and subsubroutines

In figure 4.9, the frequencies of the subroutines within the "Select vendor" routine are visible. Figure 4.10 shows the perceived importance of the subroutines.

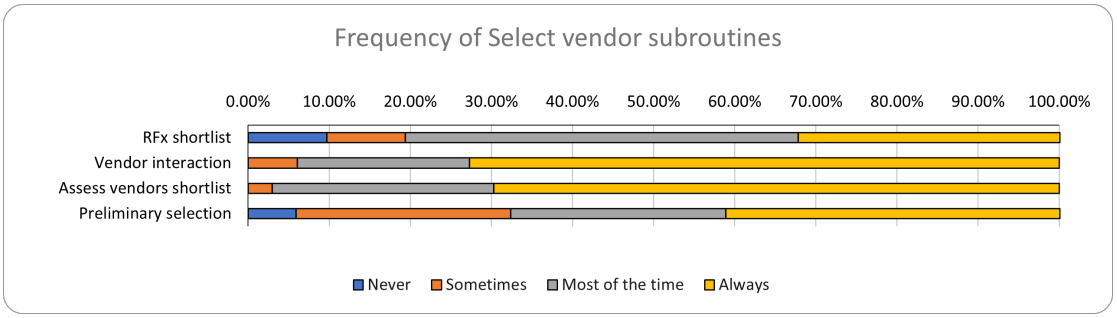

FIGURE 4.9: Frequency of Select vendor subroutines

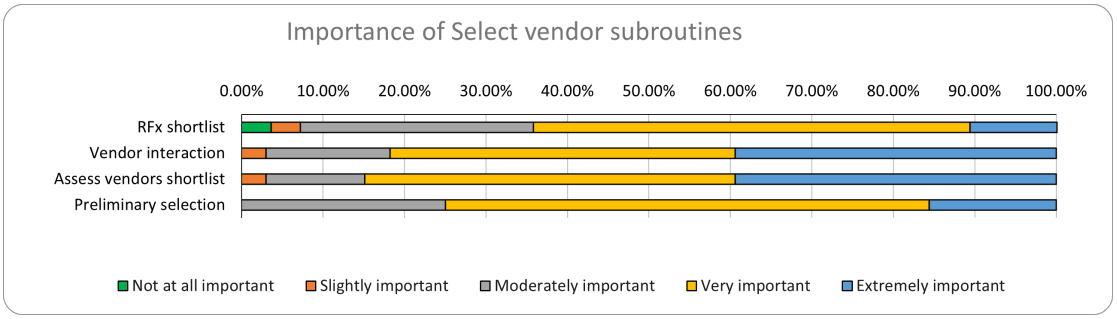

FIGURE 4.10: Importance of Select vendors subroutines

Similar to entering an RFx process with the longlisted vendors this subroutine can also be executed with shortlisted vendors. Organizations execute this subroutine more in the "Select vendor" routine than in the "Pre-select vendors" routine. This can be related to the fact that this is a time consuming subroutine and executing it on the shortlist will take less time than executing it on the longlist.

Within the "Interact with vendors" subroutine, we can identify four types of vendor presentations from the interviews: a presentation, a product demonstration, a personalized product demonstration and a prototype. In all interview cases, organizations brief vendors of their expectations. Enterprise architect, interviewee number two, said that sometimes he/she adds something unexpected to the presentation requirements, in order to test the flexibility of the vendors. These presentations, and the vendors in general, are assessed more thoroughly during this routine than during "Pre-select vendors". We see reference visits here, asking clients of software vendors how they feel about the cooperation. Weighting factors may be determined for scoring the vendors and the vendors are assessed on their offers, ability to meet vendor requirements and product requirements and implementation plan, which were presented in the "Interact with vendors" subroutine. Typically, the assessments are done individually first. This way stakeholders do not influence each other's decisions.

Finally, "Make a preliminary selection" can be executed. In some cases the budget holder is involved in the project team throughout the process but in other cases the selection and substantiation have to be presented to the budget holder or top management. The budget holder is likely to approve the preliminary selection as the project team has gone through an extensive process to reach the selection. However, it can happen that the budget holder is not convinced with the selected solution (interview 4). Being able to clearly substantiate the selection for a certain vendor is an important aspect here. A project team needs to convince the decision maker that they have considered every aspect and made the right decisions throughout the entire process. Interviewee number thirteen, sourcing consultant, said documentation is important in this subroutine: "You need to be able to show how you did it at every step."

4.2.6 Conduct pilot

After the last subsubroutine in "Select vendor", "Conduct pilot" starts. "Pilot the software", "Make final assessment" and "Make selection final", are subroutines discovered in this routine. The subsubroutines are presented in table 4.9

6. Conduct pilot		
6.1. Pilot the software	**6.2. Make final assessment**	**6.3. Make selection final**
6.1.1. Agree with vendor(s) on acceptance criteria	6.2.1. Question users on experiences with pilot	6.3.1. Send out RFP to vendor(s)
6.1.2. Supply vendor(s) with business processes to support	6.2.2. Score PoC individually	6.3.2. Select highest scoring vendor
6.1.3. Set up PoC with vendor(s)	6.2.3. Test impact on ROI and TCO	6.3.3. Approve selection
6.1.4. Organize hack-a-thon	6.2.4. Perform cost-benefit analysis	
6.1.5. Test integration with as-is IT landscape	6.2.5. Discuss point of improvement with vendor(s)	

TABLE 4.9: "Conduct pilot" routine, subroutines and subsubroutines

Figure 4.11 shows the frequency of the three subroutines and figure 4.12 shows the perceived importance.

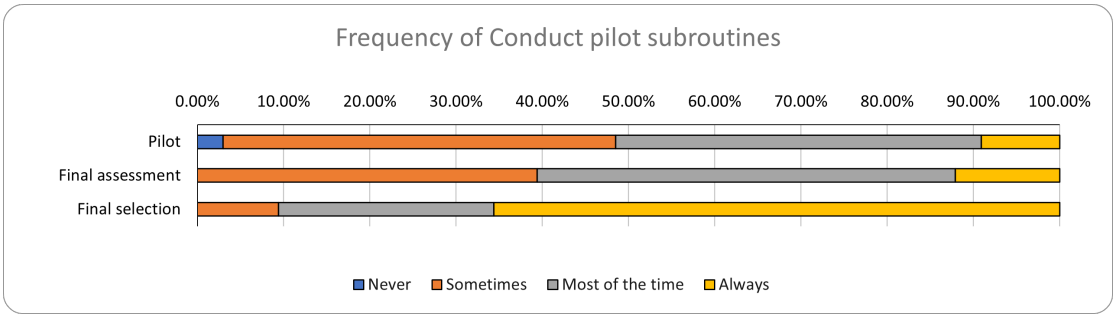

FIGURE 4.11: Frequency of Conduct pilot subroutines

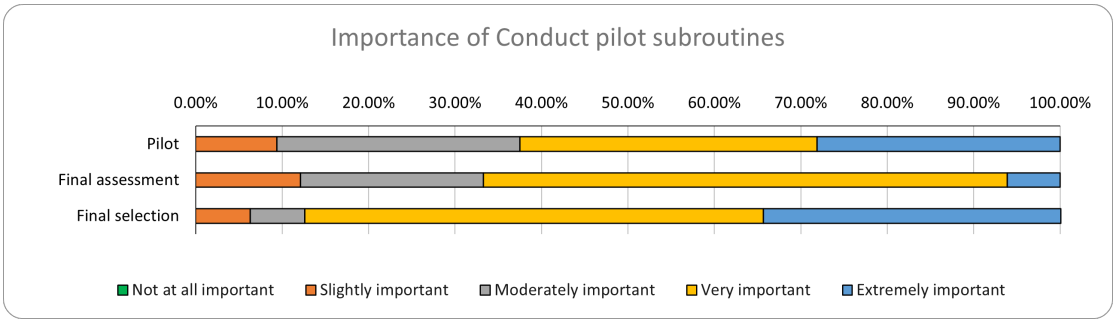

FIGURE 4.12: Importance of Conduct pilot subroutines

"Pilot the software" is the least executed subroutine of all with 52% of respondents executing it most of the time or always (figure 4.11. Even though this routine is executed the least, the interview results and perceived importance (63% very important or extremely important (4.12)) show that it can be very valuable. CIO and interviewee number 9 said that his/her organization always does a pilot. The number of vendors involved depends on the risk of the project and effort it takes. Sourcing specialist and interviewee number 7 always advises his/her client to do a pilot in one department so a vendor can show that their product works.

If a software system is piloted, organizations can execute "Agree with vendor(s) on acceptance criteria". This way the organization and the vendor share the same view on whether the pilot is successful or not. Also, the business processes that need to be supported are supplied, often in the form of user stories. Then the vendor starts building and integrating. Usually, they create a Proof of Concept (50% of interviewees). By realizing a Proof of Concept a vendor proves its solution is feasible. Another

way to test the vendor is by organizing a Hack-a-thon (interview 13). A vendor is invited to the office, the business processes and user stories are supplied on that day and the vendor is tested on flexibility and ability to develop under pressure. Either way, the most important aspect of a pilot is testing the ability of the vendor to provide the intended solution. During or after a pilot, users are asked how they experienced working with the software and are asked to score the pilot individually. Scoring does not always take place during this subroutine, checking the agreed upon acceptance criteria is enough in most cases. The points of improvement are talked over with the vendor(s). Some final analyses might be done by executing subsubroutines: "Test impact on ROI and TCO" and "Perform cost-benefit analysis". During the "Make selection final" subroutine, an organization decides which software vendor is the definitive preferred vendor. An RFP is sent out to the vendor(s) and it is up to them to write a proposal. After "Identify need", "Make selection final" is perceived to be the most important subroutine.

4.2.7 Negotiate contract

When an organization has decided a preferred vendor the seventh routine can be executed: "Negotiate contract". This routine consists of the "Enter negotiations" and "Close the deal" subroutines. Table 4.10 shows this routine's subsubroutines.

7. Negotiate contract	
7.1. Enter negotiations	**7.2. Close the deal**
7.1.1. Enter negotiations with vendor(s)	7.2.1. Draw up contract
7.1.2. Negotiate price and contract details	7.2.2. Enter an escrow agreement
7.1.3. Continue with second best vendor if negotiations fail	7.2.3. Sign contract formally

TABLE 4.10: "Negotiate contract" routine, subroutines and subsubroutines

The frequency and perceived importance for the final routine, "Negotiate contract", are visible in figure 4.13 and 4.14.

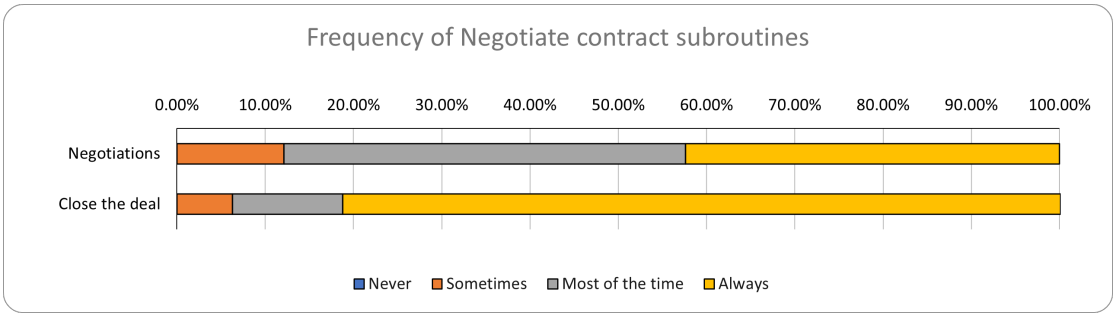

FIGURE 4.13: Frequency of Negotiate contract subroutines

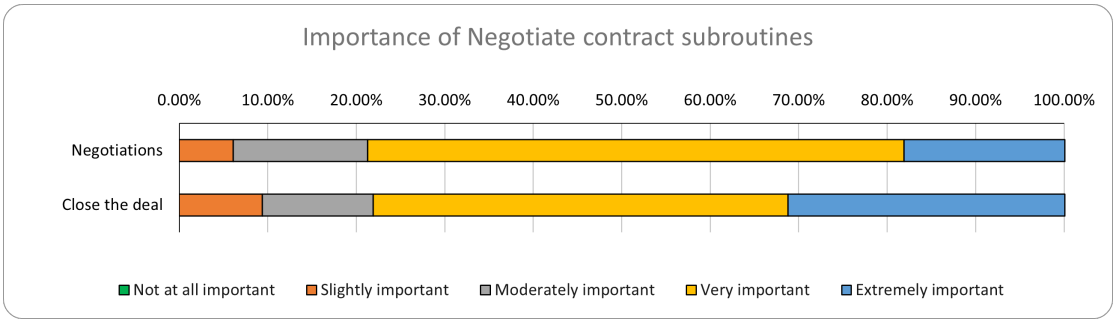

FIGURE 4.14: Importance of Negotiate contract subroutines

Once an organization gets to this routine with a vendor, it rarely happens that the deal does not go through. “If that happens, you have not gone through your preliminary phases properly because the contract should come as no surprise.”, is what interviewee number 11, director of the consultancy branch of an IT service provider said about this. Throughout the process there have been price indications and assessments of functionality and cultural fit, so if negotiations fail there must have been a major miscommunication. Interviewee number 12, a global IT director said this routine actually starts right from the beginning of a software selection process:“You cannot start early enough with the negotiation and legally securing it. There must be a lawyer from day one. You enter a commitment, especially with large software selection processes, you enter a multi-year commitment. And the software supplier then says: the technology can progress and therefore we also must adjust our terms and conditions over time. ... Then you must check whether they can immediately double the price after year one. Then you need to know the terms and conditions. For what reasons are they allowed to change the price during the term of the contract.” This interviewee advocates executing the “Enter an escrow agreement” subsubroutine when buying large software packages. More interviewees have outed their distrust towards software vendors’ pricing models. Also the pressure vendors can put on organizations to sign a contract causes distrust (interview 8). We see in practice that only 32% of survey respondents involve a legal expert in their software selection team.

When executing “Enter negotiations”, organizations can choose to do that with just their preferred vendor or with multiple vendors. Some organizations do not let the preferred vendor know they won it, because, as interviewee number 8, a CIO, said: “They need to know someone else is in the race. ... You will pay 10 or 20% more if those parties know that they are the only one.” When negotiating with more than one vendor an organization could work out a lower price. On the other hand, entering negotiations with only the preferred vendor shows that an organization trusts the capability of the vendor to successfully implement its product and trusts that the negotiations will run smoothly. What is important to remember is that during this routine it is about contract details. Important arrangements have been made earlier on in the process and as mentioned before, if “Negotiate contract” is not executed successfully, something has gone wrong in an earlier routine.

“Close the deal” is one of the most frequently executed subroutines with 81% always, 13% most of the time and 6% of the respondents sometimes executing it. Not being able to close a deal at the end of the process means an organization has put a lot of time and effort in a process that did not yield anything. “Sign contract formally” marks the end of a software selection process. Routines that are triggered after this are outside the scope of this research.

4.2.8 Common pitfalls

In table 4.11 the pitfalls that were mentioned in each routine section are collected and presented.

Routine	Interview	Pitfall
Initiate project	2	Business creates a need based on software they want instead of selecting software that fulfills their need
	8, 12	Business decides they need new software before analyzing the problem they have
	5	Due to time constraints business purchases a software package without consulting IT at all
	13	Bad planning causes the project to stretch over a long period of time
Gather requirements	2, 5, 6, 7	Business is set on acquiring a certain piece of software instead of taking the time to gather requirements
	3, 11	Requirements focus too much on the functionality of the tool instead of how it fits in the organization
Research the market	2	Due to time constraints no extensive market research is done and the longlist and "Pre-select vendors" routine are skipped
Select vendor	4	Budget holder is not convinced with selected solution
Negotiate contract	8	Software vendors pressure the business into signing a contract
	12	Start legally securing the contract too late
General	1	Insufficient involvement of stakeholders
	10	No clear roles and responsibilities due to a lack of organization

TABLE 4.11: Common pitfalls in software selection routines

4.2.9 Actors and artifacts

The actors that execute the routines are expressed as the roles of members in a software selection team. These are visible in the left side of table 4.12. The percentages represent what percentage of respondents have the role in their team. Not all roles are involved throughout the entire process. Business representatives, IT representatives, solution architect, problem owner, project manager, procurement officer, enterprise architect and the consultant roles have a larger role within projects than the experts. Their knowledge and expertise is needed in more routines. A security expert's expertise, for example, is only necessary when security requirements are set up. For budget holders, the involvement differs, as interviewee number 9, an IT director with the role of budget holder, said: "It varies with content, how detailed my involvement was. Sometimes it is purely decision-making." Organizations should be aware of who to involve in a software selection project as interviewee number 1, a procurement officer, mentioned: "The insufficient involvement of stakeholders throughout the process is a commonly made mistake." A remarkable result is that there are not two survey respondents that involve exactly the same roles in their software selection teams, which means there are 34 unique combinations of actors in this research.

On the right of table 4.12, the percentage of respondents that use each type of documentation is presented. Documentation types represent the artifacts that are used during the routines. Another type of artifact that was added by a respondent is the "Stakeholder presentation template". Similar to the actors, it differs per respondent which artifacts are used and during which routines they are used.

Other artifacts that can be used during the routines are tools and methodologies. The survey asked the respondents whether they use a standardized methodology or tools other than office software. The use of a methodology is evenly divided as can be seen in figure 4.15. 46% use a standardized methodology to execute their software selection capability. Some respondents describe their methodology as a pre-described sourcing process or a structured approach involving multiple stakeholders. Two methodologies described are in-house designed methodologies by consultancy firms. One standardized

Roles	% Respondents
Business representative	76%
IT representative	74%
Budget holder	71%
Solution architect	62%
Problem owner	59%
Project manager	59%
Procurement officer	59%
Enterprise architect	56%
(Key) user	53%
Security expert	50%
Privacy expert	41%
Process expert	38%
Legal expert	32%
External consultant	29%
Data expert	21%
Risk expert	6%
Internal consultant	6%
Archivist	3%
Subject matter experts	3%

Documentation types	% Respondents
Business case template	84%
Requirements template	84%
Contract template	84%
Scoring template	81%
Research reports	78%
BIA template	75%
RFP template	72%
PIA template	69%
Pricing template	69%

TABLE 4.12: % of respondents per role in a software selection team (left) and % of respondents per documentation type used (right)

methodology is a simple longlist, shortlist, RFI, RFP method. Another one is described as: 1) Evaluate current state; 2) Identify key areas of change, quantify impact; 3) Agree on high level future state; 4) Identify vendors; 5) Issue RFP; 6) Conduct demonstrations; 7) Downselect; 8) Follow-up, negotiate, finalize. Finally, a governmental agency uses a methodology designed to help municipalities with their software selections. Interestingly, during the interviews only two of the fourteen participants said they used a standardized methodology, the Van Weele-chain [van Weele, 2014] and the Best Value Procurement process [Storteboom et al., 2017]. The Van Weele-chain consists of the following steps: 1) Specification; 2) Supplier selection; 3) Contracting. The Best Value Procurement process is made up of four phases: preparation, selection, clarification, and execution. 21% of the sample of this research uses tools other than Office tools like word processors and spreadsheets. Mentioned tools are: an application for requirements, two comparing and evaluation tools and a cloud security tool.

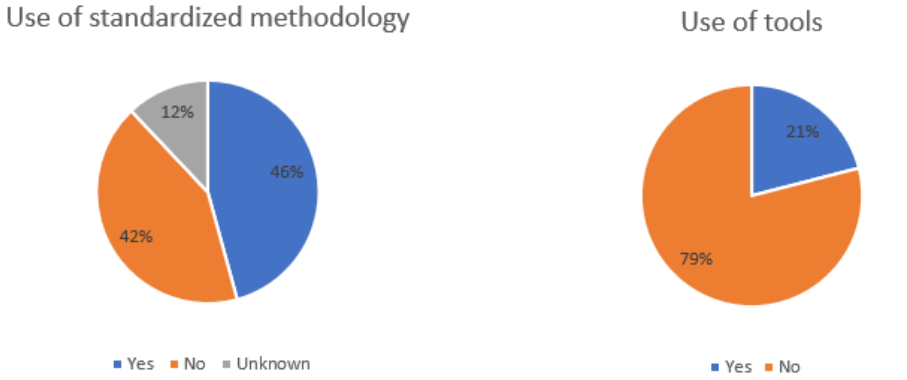

FIGURE 4.15: % of respondents that use a methodology (left) and % of respondents that use tools (right)

4.3 Software selection capability

These routines, subroutines and subsubroutines combined, form the software selection capability. To calculate the frequency and importance of the main routines by combining the subroutines, we have three options: take the average, the minimum or the maximum. This depends on whether we define the execution of a routine by the combination of its subroutines, executing one of its subroutine, or executing all of its subroutines. Because we want to see a representation of all subroutines within a routine we have decided to use the average. The average frequency of routines is presented in figure 4.16 and average perceived importance in figure 4.17. In the survey a Likert scale was used to ask respondents about the frequency and perceived importance. Likert scales are ordinal, for which the "average" function is not available. So by assigning a number to each ordinal category, and then taking the average of these numbers we make the assumption that the numbers somewhat reliably capture a notion of distance between the various categories. We see that the routines that are executed most are "Negotiate contract" (91% most of the time or always) and "Select vendor" (85% most of the time or always). Closely followed by "Gather requirements" (80% most of the time or always) and "Initiate project" (79% most of the time or always). The least frequently executed routines are "Research the market" (70% most of the time or always), "Conduct pilot" (68% most of the time or always) and "Pre-select vendors" (65% most of the time or always). All discovered routines are executed at least sometimes by 94% of the respondents.

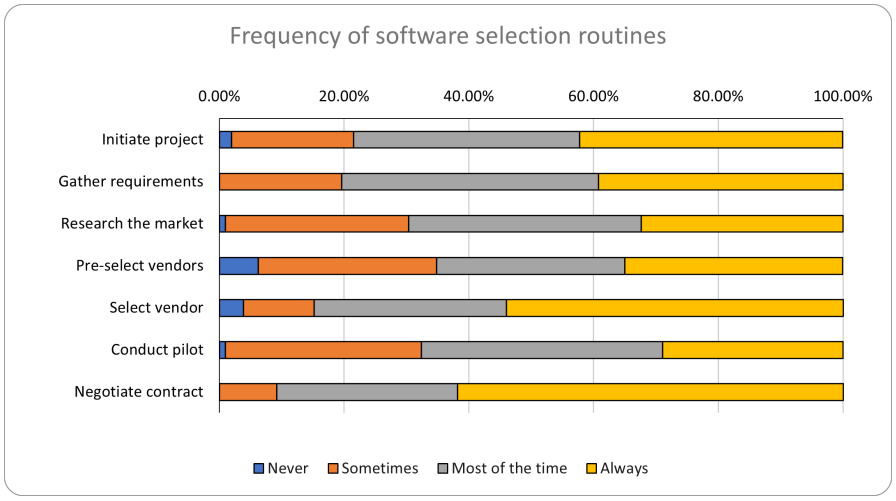

FIGURE 4.16: Frequency of routines in the software selection capability

The same is done for the perceived importance of the routines within the software selection capability as can be seen in figure 4.17. We see that "Initiate project" is perceived to be relatively important, with 84% saying it is very important or extremely important to the successful completion of a software selection project. "Negotiate contract" (79%), "Gather requirements" (77%), "Select vendor" (77%) and "Conduct pilot" (72%) follow. The least important routines are "Pre-select vendors" and "Research the market' with only 61% and 53% of respondents saying they are very important or extremely important. All seven routines are deemed at least moderately important by 87% of respondents and at least slightly important by 99%.

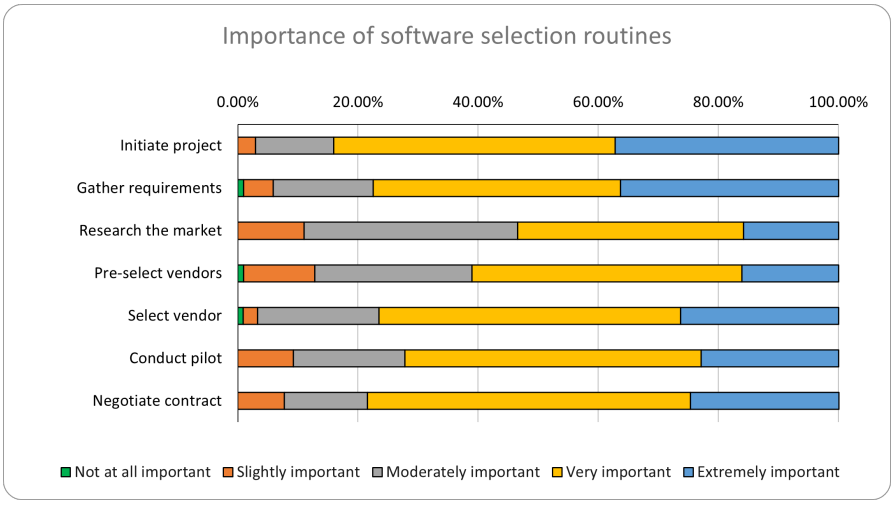

FIGURE 4.17: Importance of routines in the software selection capability

To find out more about the state of the software selection regarding capability research, this survey also asked the respondents to fill in which maturity level they perceive their software selection capability to be at, based on a capability maturity model [Curtis et al., 2009]. Most respondents believed their capability to be at a "Defined" (29%) or "Quantitatively Managed" level (29%). Eight respondents thought of their capability at a "Managed" level (24%), three at an "Initial" level (9%) and three at an "Optimizing" level (9%), as can be seen in figure 4.18.

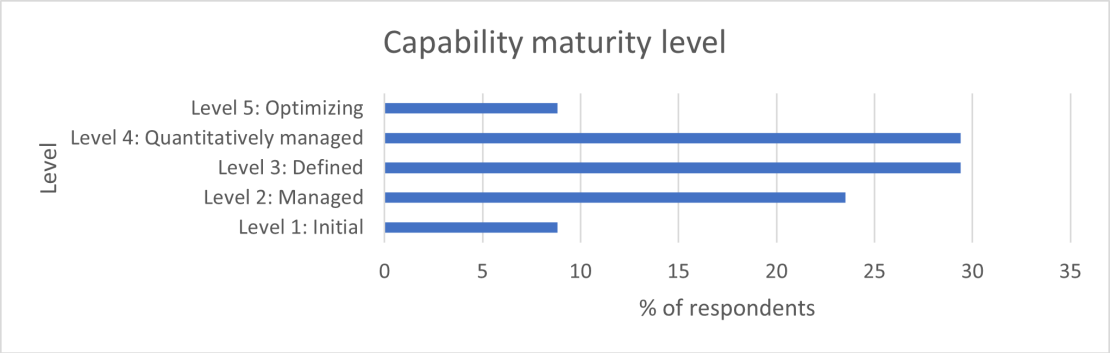

FIGURE 4.18: % of respondents per maturity level

Figure 4.19 shows the self-reported frequency of the respondents with a self-reported maturity level of four or five, quantitatively managed or optimizing. When looking at the differences with the entire sample, we chose to look at the difference in the combination of "most of the time" and "always". Remarkable differences are that respondents with a high level capability execute: "Set up project organization" 11% points more, "Draw up requirements" 15% points more, "Specify requirements" 12% points more and "Request vendor information" 10% points more frequently. On the other hand, they execute the following subroutines less: "Conduct market research" 10% points, "Enter RFx" with vendors on the longlist 10% points, "Make a preliminary selection" 14% points, "Pilot the software" 10% points and "Make final assessment" 20% points. Overall, we do not see respondents with a high maturity executing the subroutines more frequently than respondents with a maturity level of one, two or three.

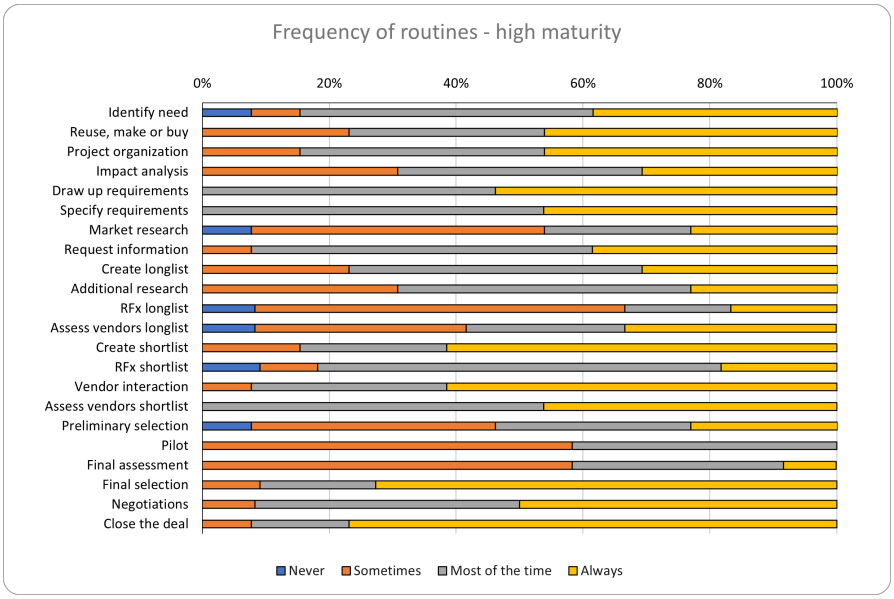

FIGURE 4.19: Frequency of software selection routines for respondents with a self-reported high maturity level

A capability always serves a certain purpose. Therefore, triggers and effects of software selection will help determine if it is perceived as a (dynamic) capability. The reasons in table 4.13 and effects in figure 4.20 were discovered during the interviews and tested against a broader audience in the survey. The reasons represent triggers for the start of the execution of the software selection capability. Respondents could give multiple answers to this question. Hence, the percentages represent the number of respondents that see that reason as a reason for the start of a software selection. The possible choices for the respondents were based on answers from the interviews and conversations with the supervisors of this Thesis. The most frequently answered reasons are "Current systems end-of-life" and "Execution of organization's (digitalization) strategy" (both 71%). Followed by "Dissatisfaction with current software systems" (68%). Other reasons are visible in table 4.13. One respondent added "Rapid changing environment" and "Growth" as an open answer.

Reason	**% Respondents**
Current systems end-of-life	71%
Execution of organization's digitalization strategy	71%
Dissatisfaction with current software systems	68%
Implementation of new processes	62%
Innovation	62%
Striving for operational excellence	59%
Implementation of new products	47%
Complying to changing laws and regulations	44%
The need to increase performance	38%
Contract expiration with current vendor	32%
Anticipating market changes	26%
Seeing software opportunities in other places	18%
Rapid changing environments	3%
Growth	3%

TABLE 4.13: % of respondents that see this reason as a reason for the start of a software selection

Figure 4.20 shows fifteen topics regarding business operations which could possibly be influenced by software selection. Respondents answered whether they thought software selection has an extremely negative, somewhat negative, no effect, somewhat positive or extremely positive effect on the topics. No one answered software selection has an extremely negative effect on any of the topics. The most positive effects are visible in "Minimizing the implementation risks" and "User satisfaction" (100% at least somewhat positive). But also "Adoption rate" (94%), "System integration" (94%) and "Reaching the goals of the business" (91%) are perceived to be affected relatively positively. We see that there are seven topics which have said to be influenced somewhat negatively by software selection, including "Creation of new processes" and "Creation of new resources".

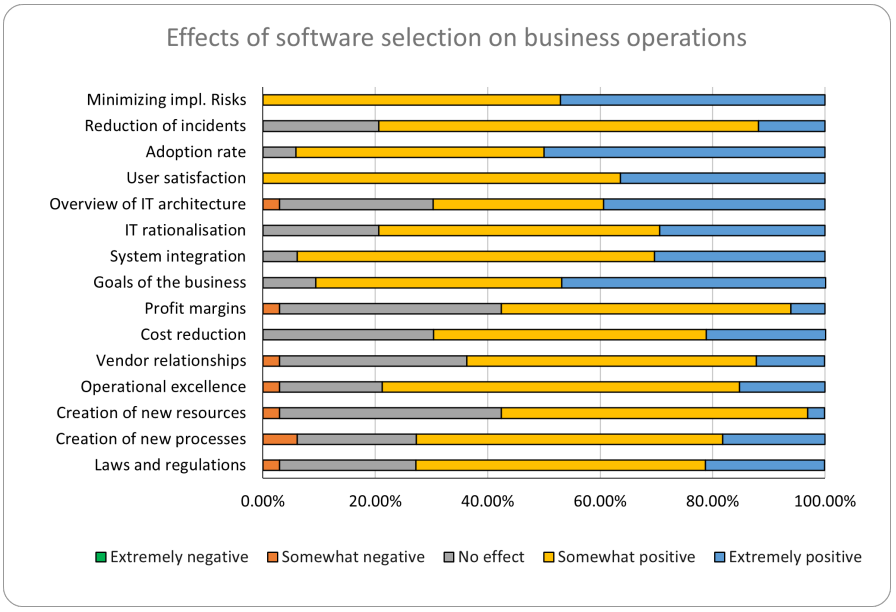

FIGURE 4.20: Effects of software selection on business operations

Chapter 5

Discussion

This chapter will discuss the subquestions and main research question as stated in the Introduction in relation to the Literature Review and Results. The three subquestions are:

- What are the routines that help organizations in their software selection capability?
- Which routines are perceived to be most important to the successful completion of a software selection?
- How can organizations implement software selection as a dynamic capability?

And the main research question is:

> *To what extent do org anizations execute continuous software selection?*

The answers to these questions have resulted in three deliverables that will also be discussed:

- An inventory list of routines that help organizations in the capability of software selection.
- An inventory list of best practices that help organizations in the capability of software selection.
- A recommendation on how to implement software selection as a dynamic capability.

After that, a section will be devoted to the limitations of this research.

5.1 Routines in the software selection capability

Looking at the frequencies and perceived importance we can conclude that each mapped subroutine helps organizations in their software selection capability. This section will look at the inventory list constructed in this research. We will discuss differences with the routines in the literature and relationships with routines in other capabilities that can help organizations in their software selection capability. Since this research did not observe any of the routines, just the ostensive part is discussed.

5.1.1 Initiate project

A major difference between the literature and this research is that we do not see the 'Initiate project' routine in the literature, except for creating a team and involving the right stakeholders in that team.

This can be seen in the list of discovered process steps in Appendix 1. This brings us to the discussion whether 'Initiate project' is a part of the software selection capability. Some people might say that it lies within the project management capability of an organization. Some subsubroutines do not have a clear connection to software selection, like: "Write first version business case" and "Plan project". These subsubroutines are necessary in any project, regardless of the goal. However, the "Initiate project" routine is perceived to be a very important factor to the successful completion of a software selection project. Therefore, being able to execute routines that exist in the project management capability benefits the software selection capability.

"Identify need" is perceived to be the most important subroutine to the completion of a software selection. If a need is identified incorrectly, everything that is done after that is time and money wasted. The entire process has been for nothing. It is important for IT specialists to understand what the business truly wants and needs. Therefore, it is important for organizations to align their business and IT. Literature on the business and IT alignment capability shows that for the business and IT to find each other, understanding of goals, strategies and needs are required [Wegmann et al., 2005]. A strong software selection capability should facilitate a good cooperation between business and IT, resulting in a software package that both sides are content with. Besides that, software in itself is a bridge between the two. The business uses the software and the IT department maintains it. Through talking about the faults and benefits of the software, business and IT can learn from each other. This improves not only the software selection capability, but also the business and IT alignment capability.

5.1.2 Gather requirements

The findings of how the "Gather requirements" routine is built up in this research are similar to the findings in the literature. Overall, "Gather requirements" is an important routine in the software selection capability. Also, it is executed significantly more by respondents with a high maturity level capability. The first subroutine, "Analyze impact of software", is positively influenced by the enterprise architecture (EA) capability. All the business processes affected by IT systems should be defined in the EA. According to Pereira et. al [Pereira and Sousa, 2005], three alignments should be in place to aggregate an EA: "business architecture and information architecture, business architecture and application architecture, and application architecture and information architecture." While only 56% of the respondents involve an enterprise architect in their software selection team, having an enterprise architect within the organization that oversees these alignments will have a positive influence on the software selection capability. Furthermore, business and IT alignment plays an important role in the "Gather requirements" routine. For a procurement department or consultants it is very important to look beyond what the business wants and try to find out what the business needs. Again, bridging the gap between business and IT. Not only will this result in a better fit for the organization, it will also speed up the process (interview 7). But while sourcing specialist and interviewee number 7 said that boarding up requirements at the start is good and you should not want changing requirements, it is important in requirement engineering to thread requirements development and management activities throughout the project, and have a good understanding of project management and risk management [Wiegers and Beatty, 2013]. Requirements can and will change. In a good requirement engineering capability, project management and risk management will allow an organization to deal with these changes in a flexible way. So, other routines that are related to the software selection capability are routines from EA, requirement engineering and risk management. Therefore, having an enterprise architect and a risk expert in a project team could be beneficial.

5.1.3 Research the market

This routine is similar to the routines found in the literature. Usually it is someone in a procurement or sourcing role that executes this routine. Software selection is closely related to the procurement capability. What procurement does, is buy products, processes, capabilities or scale, to support other capabilities. Software is such a product. "Gather requirements", "Research the market", "Pre-select vendors", "Select vendor" and "Negotiate contract" are routines that are also found in procurement. Therefore, the results have shown, the software selection capability usually involves actors capable of executing procurement routines. These people are most of the time, but not limited to: consultants or a procurement officer. Dobrzykowski et al. [Dobrzykowski et al., 2012] have found positive relationships among customer collaboration, supplier collaboration, customer IT integration, and the procurement capability. This means that by investing in the software selection capability, and thus the procurement capability, organizations will see a positive effect on these three factors.

5.1.4 Pre-select vendors

Many of the subroutines in "Pre-select vendors" in the literature, are subroutines in other routines in this research. An explanation for this is that routines in the literature were not clearly defined. It could be that routines are executed in parallel, in a different order, or subroutines are executed while executing another main routine. The inventory list in the literature is less accurate than the inventory list created in this research.

Routines from the procurement capability are important in the "Pre-select vendors" routine. The four subroutines are routines that are executed in that capability. Another capability's routines that could be beneficial to the successful completion of this routine is the stakeholder management capability. It is important to closely manage stakeholders' expectations of the functionalities of the software within budget, and keep track of their thoughts and assessments on the vendors and software. The routines within "Pre-select vendors" are complemented by routines from the stakeholder management capability.

5.1.5 Select vendors

In the literature we see a lot of emphasis on evaluation tools and techniques during this routine. Tools are only used in 21% of the cases in this research and tools that were mentioned are used for finding and comparing packages, document and procurement management, and security. The interviews show that in reality the procurement expert or external consultants supervise the project and calculate the average of the individual scores of the stakeholders in spreadsheets. The literature points out that software selection is a multiple criteria decision making problem and complex, advanced techniques, like AHP, WSM and a fuzzy based approach, are necessary to simplify this complicated problem. This research shows that these techniques and corresponding tools are not utilized in practice. These results are in line with the findings of Asadabi et al. [Asadabadi et al., 2019] who argue that managers prefer to look at selection problems intuitively.

Within "Select vendors", interacting with the vendor and assessing the vendors on the shortlist are perceived as very important subroutines by the participants. Where some interviewees would say ticking off criteria is crucial here, others say the feeling that a team gets from a vendor is evenly important in making the selection. CIO and interviewee number 6 said: "In that phase it is also a matter of feeling. That is why you often do it with a team and the team exchanges their feeling about it." and IT Operations manager and interviewee number fourteen said: "You have two things: a feeling, I also call that business intuition, and the hard criteria of your checklist." It is crucial to have the checklist there as well. Howcroft and Light [Howcroft and Light, 2002] did a case study where they saw an organization

failing to make the right decision because salesmanship convinced management. To counter this, it is important for organizations to ask vendors to demonstrate their product, instead of giving just a slick presentation. Vendors have to show what their product can do, so stakeholders can score them based on the requirements. The interviews have shown that the most frequently used method is the personalized demonstration. The organization asks vendors to demonstrate their ability to meet the drawn up user stories.

5.1.6 Conduct pilot

In the literature, pilot testing is done in six of the 39 researched cases while in the interviews it came forward in nine out of fourteen cases. Of the survey respondents 52% said that they conduct a pilot most of the time or always. We see that in practice, even though it is one of the least frequently executed subroutines, "Pilot the software" is executed more often than in the literature. During the qualitative research seven out of the nine participants that did a pilot, had vendors build a Proof of Concept (PoC). One interviewee asked the vendor to build a prototype and the final employee asked the vendor to do a pilot. The difference between the three can be seen in figure 5.1, created by employees of Nesta, an innovation agency from the UK. The use of a Minimal Viable Product was not mentioned by any of the software selection experts and is therefore disregarded in this research. We have made sure that the participants of this research use the same terminological distinctions by clarifying in the survey question that piloting methods like the PoC are also part of this subroutine.

	LAB / STUDIO		REAL WORLD	
	Proof of Concept	**Prototype**	**Pilot**	**Minimal Viable Product (MVP)**
What is the method about?	**Testing the feasibility of a crude idea or assumption to justify further development**	**Testing how an idea may work, look, or feel like to learn from and identify assumptions**	**Testing whether a solution will work in the real context to justify scaling or implementing**	**Testing the viability of the essential core of your solution in action and continuously adapting to create value**
When is it used in the process?	Early stage	Early stage	Roll out	Live testing
What are you testing?	A hunch or assumption	An idea	A solution	The core of a solution
What is the purpose of the test?	You have a hunch and want to test if it can be made real	You have an idea and want to test how it might work and learn from it	You have a solution and want to test if it actually will work and iron out minor creases before implementing or scaling it	You have the core of a solution and want to test if there is demand, if not you change your approach
When is your test a success or proven?	When your idea is feasible	When your idea works as anticipated – if not, you must have gained insights to improve it	When a solution works as anticipated	When there is demand and the solutions works as anticipated
Who's involved in testing it?	Internal stakeholders	Users, citizens, decision makers, sponsors	Real users, decision makers, sponsors	Real users
How much development time* is needed?	A couple of minutes, hours or a few days	From half an hour up to a few days or even weeks	A few weeks up to a couple of months or a year	Continuous
What costs* are involved?	A few pennies up to 1,000 GBP	A few pennies up to 5,000 GBP	10,000 GBP up to hundreds of thousands	Core part of the business model 100k up to millions.

* These numbers are indicative

FIGURE 5.1: Elaboration on PoC, prototype and pilot [Leurs and Duggan, 2018]

A reason why respondents most often use a PoC to test the software solution could be the relatively low time and costs that it takes to build one. The downside is that it gives an organization the minimal confirmation of feasibility. The result of a PoC is that an organization knows whether the vendor can build the solution. How it works and if it will completely fit the as-is IT landscape is not certain. If it

is a low-risk commodity package, this is fine. However, for high-risk strategic packages asking a vendor to develop a prototype or run a pilot could be considered. The downside of this is that not all vendors are willing to put that much resources in a product of which they are not 100% sure they will get the deal. Agreements can be made where the organization and the vendors share the costs. Depending on the risk of the project, organizations should consider which type of software testing suits their project best. A pilot might not be worth the time and money if a vendor and organization are certain about achieving the desired result.

5.1.7 Negotiate contract

Most interviewees, especially the ones that select expensive software packages from big vendors, do not trust software vendors. The distrust is found mainly in the pricing models. This is in line with research conducted on pricing strategies of software vendors. Lehmann and Buxmann [Lehmann and Buxmann, 2009] asked users what they thought of pricing models of vendors. The assessments were negative. The main reason for this is the complexity of (the combination of) pricing models. This complexity causes users to believe that vendors increase their prices for no good reason and execute software audits to earn money. Everything is aimed at increasing sales and profits. "Negotiate contract" is a routine within the negotiation capability of an organization. If an organization does not have this capability in-house, it would be recommended to hire a third party legal expert to support the software selection team.

5.1.8 Actors and artifacts

Software selection teams typically consist of four to ten people taking on multiple roles. This research did not go into enough detail to pinpoint which routines are executed by which actors. Besides, actors differ per project and organization. We can only say which actors are involved in software selection projects in general. As mentioned in the literature, a selection team of stakeholders is formed. All but one of the roles mentioned in the literature came forward in this research as well: business and IT experts/managers, procurement experts, consultants and end-users. The controller was not discovered as a role in this research. The business and IT managers are often involved as a budget holder, project manager and/or problem owner, or are not involved in the selection team but function as a decision maker at the end of the process. A project manager is important for a software selection team since routines from the project management capability play a big role in software selection. The literature puts emphasis on involving users in the selection process [Chau, 1994, Verville et al., 2005, Howcroft et al., 2010], even calling it a success factor. In practice, (key) users are involved in only 53% of the software selection projects. There are many roles that came forward in this research and are not mentioned in the existing literature: budget holder, problem owner, enterprise architect, solution architect, process expert, security expert, privacy expert, legal expert, data expert and archivist.

Artifacts other than the documentation mentioned in the routine sections are methodologies and tools. The methodologies described in the results all consist of routines described in this research, although sometimes named differently. This shows that software selection experts can experiment with the routines discovered in this research to find the ideal process for them. Going from there they can create a standardized methodology that fits their organization. Other artifacts that can be used in software selection routines are tools. Tools and techniques as described in the literature are not used in practice, following the research of Asadabi et al. [Asadabadi et al., 2019].

5.2 Best practices in the software selection capability

The second deliverable is an inventory list of best practices within software selection. This inventory list is based on the results regarding frequencies and perceived importance of the discovered routines. Certain routines are combined and others are left out entirely. The inventory list can be found in table 5.1.

Best practice	When?	How?	Why?
1. Identify need	Continuously	1.1. Analyze business goals, strategy and needs 1.2. Analyze business processes 1.3. Translate need to IT 1.4. Consider process reengineering or developing	A big factor in the successful completion of a software selection is business and IT alignment. These steps facilitate a clear understanding between business and IT. Determining whether sourcing is the right solution is also important.
2. Set up project organization	The decision has been made to select software	2.1. Involve the right stakeholders 2.2. Decide procurement strategy 2.3. Create project planning 2.4. Draw up first version business case	Having clear roles and responsibilities is important when aiming to improve a capability. Mistakes are tracked more easily and prevented in the future. Also, a clear time schedule and budget are necessary to determine feasibility of certain software solutions.
3. Gather requirements	Project kick-off	3.1. Make Business Impact Analysis (BIA) 3.2. Gather functional, performance and vendor requirements through workshops with stakeholders and users 3.3. Gather technical, security and privacy requirements from subject matter experts 3.4. Create user stories 3.5. Determine knock-out criteria	A BIA guarantees the involvement of the right stakeholders and makes sure everyone is on the same page about process reengineering tasks. Clear requirements ensure a better organizational fit and a speedier process. User stories create a comprehensible overview and knock-out criteria will be used to drop vendors early on in the process.
4. Create shortlist	Requirements are gathered	4.1. Consult technology research agencies about possible solutions 4.2. Acquire vendor information through documents and Q&As 4.3. Apply knock-out criteria 4.4. Create shortlist	Research agencies like Gartner and Forrester are aware of the latest trends in software and can help organizations select the software that fits best. To be able to apply the knock-out criteria, information about these vendors is necessary. The most effective way to get this information is by asking if they can meet the knock-out criteria. The three vendors that come out best are put on the shortlist.
5. Select vendor	The shortlist is created	5.1. Ask vendors to demonstrate ability to meet user stories 5.2. Have stakeholders individually score the demonstrations on functionality, performance and cultural fit 5.3. Consider price and implementation plan 5.4. Discuss scores in a consensus meeting 5.5. Select preferred vendor	By demonstrating the ability to meet user stories in a personalized setting, vendors can give an extensive view on functionality and performance of their software while also showing if the cultural fit is there. Individually scoring these aspects makes sure stakeholders do not affect each other in the process. Discussing the scores and selecting the vendor that comes out best ensures an unbiased selection.
6. Sign contract	A preferred vendor is selected	6.1. Enter negotiations with one or more vendors depending on the procurement strategy 6.2. Secure pricing and other contract details 6.3. Close the deal	Large software vendors have the reputation of "tricking" organization out of their money by increasing their prices over the years for no apparent reason. Legally securing the pricing model is key. Hiring a third party legal team to oversee this process is an option.

TABLE 5.1: The best practices in software selection as perceived by experts

5.3 Software selection as a (dynamic) capability

We know that software selection is a capability consisting of 90 routines, classified in 22 subroutines and seven main routines, and we have learned about actors and artifacts that exist within the routines. When looking at the organization structure of Salvato and Rerup [Salvato and Rerup, 2011] in figure 2.2, we see that a capability is in place to execute the firm strategy. Of the respondents, 71% have said that executing the organization's (digitalization) strategy is a reason to execute software selection. Furthermore, the results show that 67% of the sample define, quantitatively manage or optimize their software selection capability. This means that one third of the respondents have not defined software selection as a capability and use it as an ad hoc problem solving technique. We can conclude that software selection is at least seen as an operational capability by two thirds of the people that have executed software selections.

Looking at the capability landscape of organizations, we have already determined that project management, business and IT alignment, enterprise architecture, requirement engineering, risk management, stakeholder management, procurement, negotiation and software selection itself are capabilities that can positively influence the software selection capability. In turn, software selection has an effect on other capabilities. Table 5.2 shows which perceived effects of software selection can positively influence other organizational capabilities.

The creation of new processes and resources, operational excellence, cost reduction, profit margins and goals of the business are effects that can positively influence all capabilities within the organization.

Software implementation capability	**Human resource capability**	**Enterprise architecture capability**
Minimizing implementation risks	Adoption rate	Overview of IT architecture
Reduction of incidents	User satisfaction	IT rationalisation
		System integration
		Laws and regulations
Software selection capability	**Procurement capability**	**Business & IT alignment capability**
IT rationalisation	IT rationalisation	IT rationalisation
Vendor relationships	Vendor relationships	Adoption rate
		User satisfaction

TABLE 5.2: Perceived effects of software selection that influence other capabilities

Since the effects of software selection discovered in this research are not measured but only perceived by participants, we do not know for certain if software selection has an effect on these capabilities.

Figure 5.2 shows the relationship of software selection with other capabilities in organizations' capability landscapes. This figure is based on interpretations of overlapping routines with other capabilities and the perceived effects of software selection. These relationships are not tested. The software selection capability is strengthened by routines from the red capabilities, software selection routines strengthen the green capabilities and software selection and the blue capabilities complement each other.

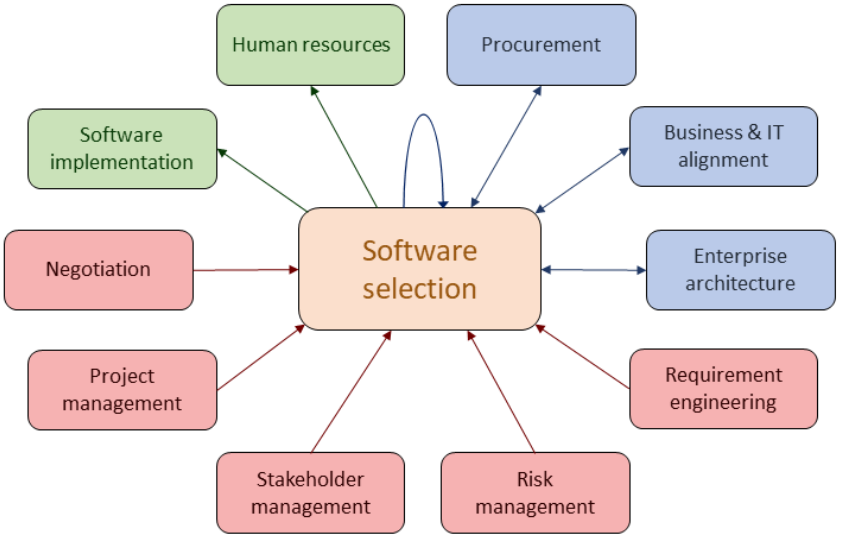

FIGURE 5.2: Software selection as the center of a capability landscape

To discover if software selection can be seen as a dynamic capability, we need to take a look at the characteristics of a dynamic capability. The literature has shown that dynamic capabilities continuously alter the way in which an organization makes a living. It does this by changing the product, production process, capabilities, the scale, or the customers served to constitute economically significant change. According to Teece [Teece, 2007], all dynamic capabilities consist of routines regarding sensing, seizing and reconfiguring. These are the microfoundations of dynamic capabilities. Product innovation and certain marketing capabilities aimed at attracting new customers are examples of dynamic capabilities. Before determining how software selection can be implemented as a dynamic capability, we should see which characteristics of a dynamic capability software selection has in its current state. Five questions should be answered to learn more about the current state, and discuss the future state of the software selection capability.

1. Does software selection change a product, production process, capability, scale or the customers served to constitute economically significant change?

When looking at the reasons why organizations execute software selection projects, many are aimed at changing a product or production process. Table 5.3 shows which reasons indicate software selection to be a method for ad hoc problem solving and which reasons could be indications of a dynamic capability.

Indications of ad hoc problem solving	Possible indications of a dynamic capability
Current systems end-of-life	Execution of organization's (digitalization) strategy
Dissatisfaction with current software systems	Innovation
Complying to changing laws and regulations	The need to increase performance
Contract expiration with current vendor	Striving for operational excellence
	Implementation of new products
	Implementation of new processes
	Growth

TABLE 5.3: Reasons for software selection as indications of a dynamic capability

Interviewee number six, CIO of a large online retailer, said the big strength of his organization is being able to select the right software packages and combine them to form a service that gives them a competitive advantage. This way, software selection can cause product leadership, constituting economically significant change. With regards to the (perceived) changing of capabilities, see table 5.2. We can conclude that the software selection capability is already deployed to change products, processes and capabilities to constitute economically significant change.

So, in order to deploy software selection as a dynamic capability, organizations must not see software selection as a way to solve problems. It must see software selection as a way to improve current products, processes and capabilities. By creating a digitalization strategy that strives for innovation, operational excellence and the creation of new products and processes, organizations will be able to use software selection as a dynamic capability instead of a problem solving technique.

2. Does software selection consist of routines regarding sensing?

Routines regarding sensing can be two types of routines: routines that sense opportunities within the organization and routines that sense opportunities from outside of the organization. The subsubroutine 'Recognize arising need from the business" can be interpreted as a routine regarding sensing within the organization. But this does mean that the business has to actively try to find points of improvements that can be solved with software. We can argue that organizations that use software selection to improve their business through "Innovation", "Striving for operational excellence" and "The need to increase performance", are actively trying to find points of improvement. Furthermore, most organizations have a procurement department in place that is constantly keeping up with the latest trends. By executing the "Conduct market research" subroutine on a constant basis instead of just during a project, it can be used as a routine to sense opportunities. However, "Anticipating market changes" and "Seeing software opportunities in other places" are a reason to select software for only 26% and 18% of respondents.

An organization should execute high-level sensing routines to be able to deploy software selection as a dynamic capability. Within organizations, needs are recognized by the business, while IT procurement knows more about what is possible software wise. For example, when a contract is signed for a piece of software that is necessary for an innovative product, procurement has supported the product innovation capability. Even though the procurement capability has caused the product to change, there are no routines within procurement that make it a dynamic capability. For software selection, this might also be the case. As long as other capabilities consist of the high-level routines and software selection is used

to support the changing of products and processes in other capabilities, software selection is executed as an operational capability.

We see two possibilities for implementing high-level sensing routines. The first solution involves implementing software selection as a dynamic managerial capability. That way managers will be responsible for sensing software opportunities from within and outside of the organization. The second, and better solution to implement sensing routines would be to have a "sensing team" of process and software experts that bridges the gap between business and IT, and works full-time on spotting software improvements in business processes. The sensing team will continuously execute new sensing routines like: "Evaluate IT landscape" and "Analyze business processes". By combining these routines with original subroutines regarding knowledge of the market, the sensing team will continuously be acquiring information from inside and outside the organization. Deploying a sensing team and continuously executing sensing routines could result in a rapid growth in the number of software packages that need to be implemented. That requires a flexible EA. Gromoff et al. [Gromoff et al., 2012] have analyzed how an EA can be made more flexible. They anticipate the movement of the EA to a real-time business architecture (RBA) to become more flat and market adaptive. An RBA is cloud-based and supported by other vendors (Software-as-a-Service (SaaS)). It stimulates self-generated business services by the employees that recognize points of improvements in their business processes. Hai and Sakoda [Hai and Sakoda, 2009] argue that organizations are looking for less risky, cheaper alternatives with more predictable pricing models and a faster return on investment. With no large upfront investments, SaaS requires less resources to commit to a tool. On the other hand, SaaS applications need to be configured, processes need to be adapted and users need to be trained and switched over. Because of these reasons there are switching barriers. Even though it increases the flexibility of the IT architecture, it is not profitable to constantly change between SaaS tools. To lower these barriers even more, organizations should switch over to microservices for small business needs. Microservices are small applications with a single responsibility that can be procured, deployed and tested independently [Thönes, 2015]. This way change happens gradually. Specific needs can be solved with very specific services. This means that employees do not have to get used to large software packages at once and no extensive business process reengineering takes place. It also makes the pricing model more predictable. An organization has a better overview of what it pays for exactly.

3. Does software selection consist of routines regarding seizing?

The entire software selection capability is aimed at seizing an opportunity. Namely, selecting the right piece of software. Typical for seizing routines are spreading chances and investing in multiple technologies, before committing fully to one. This is exactly what the routines in a software selection project are aimed to do. "Pre-select vendors" and "Select vendor" are routines that help organizations go from a wide array of possibilities to the package that requires full commitment. By executing the "Close the deal" subroutine, an organization fully commits to one opportunity. We could argue that routines within the implementation capability are also necessary to seize an opportunity. A piece of software can not be used before it is implemented.

In order for software selection to be a dynamic capability, routines to finalize the seizing are necessary. This can be achieved by selecting software that does not need implementation. As mentioned before, SaaS tools and microservices are required. These tools require minimal implementation since they are accessible through the cloud. By removing the implementation capability, the software selection capability's routines are sufficient to seize the opportunity.

4. Does software selection consist of routines regarding reconfiguring?

The same problem as with seizing routines arises here: the software selection capability is not complete enough to be dynamic. Routines regarding reconfiguring are usually part of the software implementation capability. As presented in question one, software selection does reconfigure products, processes and capabilities, but this can only be seen after the software is implemented.

Selecting microservices covers part of this problem as well. To complete the reconfiguring routines, we need to include some of the post-selection routines in the software selection capability. These routines include: "Set up strategic, tactical and operational consultations" to evaluate the use of the new software, "Convert data to fit new tooling", "Train users", "Dismantle previous system" (if not a SaaS tool), and finalizing the reconfiguration with "Hand over to operational processes". These routines were usually part of the implementation capability. If software selection is deployed as a dynamic capability, the routines should be executed in the software selection capability.

5. To what extent do organizations execute software selection continuously?

To answer the main research question, we will first look at the triggers for software selection. This shows mixed results. Organizations that use software selection mainly to replace current systems execute it as ad hoc problem solving. They wait for a problem or need to arise and react with setting up a software selection project. The fact that software is constantly necessary to solve business problems makes the execution of software selection projects continuous, but no routines are clearly aimed at actively, continuously looking for opportunities regarding the use of software. Consequently, software selection is not a continuous process. However, organizations that use software selection to execute an innovative digitalization strategy might execute software selection continuously. Information manager and interviewee number 10 said that each year they make a planning of which software selection projects to execute in the upcoming year. In that case, sensing routines should be in place to actively look for points of improvement. These routines have not come forward in the results of this research. It might be that respondents did not mention those routines as they do not see them as part of the software selection capability. Other capabilities like product innovation, enterprise architecture or business and IT alignment are capabilities with the sensing routines, with software selection being a capability executed to seize the sensed opportunities. As mentioned before, the entire software selection capability is aimed at seizing opportunities. Sensing and reconfiguring are, in the current state of the software selection capability, part of other capabilities.

Concluding, organizations do execute software selection continuously because the business needs caused by the digitalization strategy arise continuously. To be able to seize as many of these opportunities as possible, organizations execute software selection projects at a constant pace. However, software selection is not executed as a continuous process. Most organizations do not have a dedicated team that works full-time on software selections. Instead of software selection being one continuous process, each software selection is its own process. It is linear rather than circular.

How to implement software selection as a dynamic capability?

The discovered routines, reasons and effects show that software selection is generally not seen as a dynamic capability. One third of software selectors have not defined their software selection capability. While software selection is used to change products, processes and capabilities to constitute economically significant change, this is mostly done in an ad hoc problem solving manner. The routines to be able to execute software selection continuously are, for most organizations, not in place. The third deliverable of this Thesis consists of a recommendation on how organizations can implement software selection as

a dynamic capability. Long-term commitments to specialized resources are necessary to develop such a capability [Winter, 2003] but these three steps, based on the previous section, will make it a profitable investment:

1. Create a digitalization strategy that strives for innovation, operational excellence and the creation of new products and processes through the use of software.
2. Create a flexible enterprise architecture by committing to SaaS tools and cloud-based microservices.
3. Deploy a "sensing team" that continuously looks for software opportunities within the enterprise architecture, products, business processes and capabilities.

With the continuous execution of software selection, organizations will be able to quickly adapt to the changing digital environment, create more effective product leadership and achieve operational excellence. While most organizations might not realize it yet, optimizing the software selection capability is going to be necessary to survive in a world dominated by disruptive technology.

5.4 Limitations

There are multiple limitations to this research. First of all, the participants of the interviews were all Dutch, working mostly for Dutch organizations. Therefore the discovered routines might be typical for Dutch organizations and cause a cultural bias. Another limitation of this study has been the lack of prior research done on this topic. Due to the limited input, interviews had to be conducted with experts in the field before survey questions could be created. This means that the survey questions have not been based on academic literature. Furthermore, because of a lack of respondents to the survey it was not possible to find significant correlations between variables in the quantitative research. For further research it would be recommended to find more respondents across the boarder. The survey is still online and we keep looking for respondents. Because interviews had to be conducted first, there was not enough time to find more respondents within the set time frame of this thesis.

Chapter 6

Conclusion

The goal of this research was to open up the black box that is the software selection capability. To reach this goal, the study mapped out the capability in terms of operational routines, identified the importance of these routines, and analyzed the current state of the software selection capability. By interviewing fourteen experts in the field, a list of 222 process steps could be created. This list was transformed into an inventory list consisting of 90 subsubroutines, classified in 22 subroutines, classified in 7 routines that help organizations in their software selection capability (table 2 in the Appendix). The main routines are:

1. Initiate project
2. Gather requirements
3. Research the market
4. Pre-select vendors
5. Select vendor
6. Conduct pilot
7. Negotiate contract

Under each routine are two, three or four subroutines that were used as input for a survey. The survey tested the frequency and the perceived importance of the subroutines on 34 software selection experts. Looking at the results of these two factors, the qualitative research and the literature, an inventory list of best practices in software selection was created. This inventory list consists of six routines and 25 subroutines and can be found in table 5.1 in the Discussion. The main routines in this list are:

1. Identify need
2. Set up project organization
3. Gather requirements
4. Create shortlist
5. Select vendor
6. Sign contract

To place software selection in the capability landscape of an organization, we have interpreted which of the discovered routines might complement other capabilities, and looked at the perceived effects of software selection. This has shown that the software selection capability might be positively influenced by the development of the following capabilities: procurement, business and IT alignment,

enterprise architecture, requirement engineering, risk management, stakeholder management, project management, and negotiation. A strong software selection capability might influence other capabilities as well: software implementation, HR, software selection, procurement, business and IT alignment and enterprise architecture. Despite the effect on other capabilities, we have seen that software selection in its current state is not seen as a dynamic capability. It is generally deployed as an operational capability or an ad hoc problem solving technique. It is used to execute an organization's strategy but it does not have the high-level routines that facilitate economically significant change. Software selection is a capability that is executed to seize opportunities. It lacks the routines to sense and reconfigure.

To answer the third subquestion of this research, we focused on how a software selection can be turned from an operational capability or ad hoc problem solving technique, to a dynamic capability. We recommend focusing on the strengths of software and making it a leading aspect in the organization's strategy. Furthermore, we advise organizations to optimize their enterprise architecture capability to improve flexibility and install high-level sensing routines. That is the key to profiting from continuous software selection. This resulted in a three step approach:

1. Create a digitalization strategy that strives for innovation, operational excellence and the creation of new products and processes through the use of software.
2. Create a flexible enterprise architecture by committing to SaaS tools and cloud-based microservices.
3. Deploy a "sensing team" that continuously looks for software opportunities within the enterprise architecture, products, business processes and capabilities.

After opening up the blackbox that is the software selection capability, we can conclude that organizations execute software selection projects continuously. Digitization needs arise at such a rapid pace that constant software selection is necessary to keep up. However, it is not a continuous process. It is rather the continuous kick off of new processes. This research has shown that high-level routines to sense software opportunities and reconfigure resources are not in place in the current landscape. Software selection is used to support other capabilities because organizations are not aware of the benefits that software selection could have. Software selection can contribute to product leadership, operational excellence and an advantage over competitors. With software becoming more and more indispensable, the opportunities are also growing. Continuous software selection is the key to grasping those opportunities.

Further research

This study lacks in research on the performative part of software selection routines. The data collection techniques used, focus on the ostensive dimension. This is the part of the routine that people have in mind or is documented. By observing software selection projects at different organizations we can learn more about what the performative dimension looks like. Another follow-up step would be to map out the actors and all documentation per routine. Since in cases there does not seem to be one team but rather a group of stakeholders working on a software selection part-time, the use of RACI roles might be interesting to look at. The ostensive dimension is investigated extensively in this research but in order to create a more complete picture the performative dimension, artifacts and actors should be researched more accurately. Finally, further research can be done on software selection as a capability. For example, try to measure the effects of software selection, or research the relationship with other capabilities. The three step recommendation for software selection as a dynamic capability can be developed further to create comprehensive guidelines. Then, the feasibility of it can be researched. It would be interesting to see if deploying software selection as described in this thesis will increase the positive effects on the business operations and create a competitive advantage.

Bibliography

[Adhikari et al., 2004] Adhikari, A., Lebow, M. I., and Zhang, H. (2004). Firm characteristics and selection of international accounting software. *Journal of International Accounting, Auditing and Taxation*, 13(1):53–69.

[Adner and Helfat, 2003] Adner, R. and Helfat, C. E. (2003). Corporate effects and dynamic managerial capabilities. *Strategic management journal*, 24(10):1011–1025.

[Aguinis, 2009] Aguinis, H. (2009). *Performance management*. Pearson/Prentice Hall Upper Saddle River, NJ.

[Albert and Brownsword, 2002] Albert, C. and Brownsword, L. (2002). Evolutionary process for integrating cots-based systems (epic): An overview.

[Asadabadi et al., 2019] Asadabadi, M. R., Chang, E., and Saberi, M. (2019). Are mcdm methods useful? a critical review of analytic hierarchy process (ahp) and analytic network process (anp). *Cogent Engineering*, 6(1):1623153.

[Ayağ and Özdemir, 2007] Ayağ, Z. and Özdemir, R. (2007). An intelligent approach to erp software selection through fuzzy anp. *International Journal of Production Research*, 45(10):2169–2194.

[Bandor, 2006] Bandor, M. S. (2006). Quantitative methods for software selection and evaluation. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

[Bapuji et al., 2012] Bapuji, H., Hora, M., and Saeed, A. M. (2012). Intentions, intermediaries, and interaction: Examining the emergence of routines. *Journal of Management Studies*, 49(8):1586–1607.

[Bernroider and Koch, 2001] Bernroider, E. and Koch, S. (2001). Erp selection process in midsize and large organizations. *Business Process Management Journal*.

[Bernroider and Schmöllerl, 2013] Bernroider, E. W. and Schmöllerl, P. (2013). A technological, organisational, and environmental analysis of decision making methodologies and satisfaction in the context of it induced business transformations. *European Journal of Operational Research*, 224(1):141–153.

[Bhuta and Boehm, 2005] Bhuta, J. and Boehm, B. (2005). A method for compatible cots component selection. In *International Conference on COTS-Based Software Systems*, pages 132–143. Springer.

[Caniels and Gelderman, 2005] Caniels, M. C. and Gelderman, C. J. (2005). Purchasing strategies in the kraljic matrix—a power and dependence perspective. *Journal of purchasing and supply management*, 11(2-3):141–155.

[Chau, 1994] Chau, P. (1994). Selection of packaged software in small businesses. *European Journal of Information Systems*, 3(4):292–302.

[Chau, 1995] Chau, P. Y. (1995). Factors used in the selection of packaged software in small businesses: views of owners and managers. *Information & Management*, 29(2):71–78.

[Cochran and Chen, 2005] Cochran, J. K. and Chen, H.-N. (2005). Fuzzy multi-criteria selection of object-oriented simulation software for production system analysis. *Computers & operations research*, 32(1):153–168.

[Cohen and Levinthal, 1990] Cohen, W. M. and Levinthal, D. A. (1990). Absorptive capacity: A new perspective on learning and innovation. *Administrative science quarterly*, pages 128–152.

[Collier et al., 1999] Collier, K., Carey, B., Sautter, D., and Marjaniemi, C. (1999). A methodology for evaluating and selecting data mining software. In *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences. 1999. HICSS-32. Abstracts and CD-ROM of Full Papers*, pages 11–pp. IEEE.

[Collis, 1994] Collis, D. J. (1994). Research note: how valuable are organizational capabilities? *Strategic management journal*, 15(S1):143–152.

[Colombo and Francalanci, 2004] Colombo, E. and Francalanci, C. (2004). Selecting crm packages based on architectural, functional, and cost requirements: Empirical validation of a hierarchical ranking model. *Requirements engineering*, 9(3):186–203.

[Curtis et al., 2009] Curtis, B., Hefley, B., and Miller, S. (2009). People capability maturity model (p-cmm) version 2.0. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INST.

[Cyert et al., 1963] Cyert, R. M., March, J. G., et al. (1963). A behavioral theory of the firm. *Englewood Cliffs, NJ*, 2(4):169–187.

[D'Adderio, 2011] D'Adderio, L. (2011). Artifacts at the centre of routines: Performing the material turn in routines theory. *Journal of Evolutionary Economics*.

[Dionysiou and Tsoukas, 2013] Dionysiou, D. D. and Tsoukas, H. (2013). Understanding the (re) creation of routines from within: A symbolic interactionist perspective. *Academy of Management Review*, 38(2):181–205.

[Dobrzykowski et al., 2012] Dobrzykowski, D. D., Hong, P. C., and Park, J. S. (2012). Building procurement capability for firm performance: a service-dominant logic view. *Benchmarking: An International Journal*.

[Dosi et al., 2000] Dosi, G., Nelson, R. R., Winter, S. G., et al. (2000). *The nature and dynamics of organizational capabilities*. Oxford university press.

[Feldman, 2000] Feldman, M. S. (2000). Organizational routines as a source of continuous change. *Organization science*, 11(6):611–629.

[Feldman and Pentland, 2003] Feldman, M. S. and Pentland, B. T. (2003). Reconceptualizing organizational routines as a source of flexibility and change. *Administrative science quarterly*, 48(1):94–118.

[Grau et al., 2004] Grau, G., Carvallo, J. P., Franch, X., and Quer, C. (2004). Descots: a software system for selecting cots components. In *Proceedings. 30th Euromicro Conference, 2004.*, pages 118–126. IEEE.

[Griffith and Harvey, 2001] Griffith, D. A. and Harvey, M. G. (2001). A resource perspective of global dynamic capabilities. *Journal of international business studies*, 32(3):597–606.

[Gromoff et al., 2012] Gromoff, A., Kazantsev, N., Kozhevnikov, D., Ponfilenok, M., and Stavenko, Y. (2012). Newer approach to create flexible business architecture of modern enterprise. *Global Journal of Flexible Systems Management*, 13(4):207–215.

[Guo et al., 2011] Guo, J., White, J., Wang, G., Li, J., and Wang, Y. (2011). A genetic algorithm for optimized feature selection with resource constraints in software product lines. *Journal of Systems and Software*, 84(12):2208–2221.

[Hai and Sakoda, 2009] Hai, H. and Sakoda, S. (2009). Saas and integration best practices. *Fujitsu Scientific and Technical Journal*, 45(3):257–264.

[Hanine et al., 2016] Hanine, M., Boutkhoum, O., Tikniouine, A., and Agouti, T. (2016). Application of an integrated multi-criteria decision making ahp-topsis methodology for etl software selection. *SpringerPlus*, 5(1):1–17.

[Harnisch, 2014] Harnisch, S. (2014). Enterprise-level packaged software acquisition: a structured literature review through the lens of it governance.

[Harreld et al., 2007] Harreld, J. B., O'Reilly III, C. A., and Tushman, M. L. (2007). Dynamic capabilities at ibm: Driving strategy into action. *California management review*, 49(4):21–43.

[Helfat et al., 2009] Helfat, C. E., Finkelstein, S., Mitchell, W., Peteraf, M., Singh, H., Teece, D., and Winter, S. G. (2009). *Dynamic capabilities: Understanding strategic change in organizations*. John Wiley & Sons.

[Helfat and Winter, 2011] Helfat, C. E. and Winter, S. G. (2011). Untangling dynamic and operational capabilities: Strategy for the (n) ever-changing world. *Strategic management journal*, 32(11):1243–1250.

[Hlupic and Mann, 1995] Hlupic, V. and Mann, A. S. (1995). Simselect: a system for simulation software selection. In *Proceedings of the 27th conference on Winter simulation*, pages 720–727.

[Howard-Grenville and Rerup, 2016] Howard-Grenville, J. and Rerup, C. (2016). A process perspective on organizational routines. *The SAGE handbook of organization process studies*, pages 323–337.

[Howcroft and Light, 2002] Howcroft, D. and Light, B. (2002). A study of user involvement in packaged software selection. *ICIS 2002 Proceedings*, page 7.

[Howcroft and Light, 2006] Howcroft, D. and Light, B. (2006). Reflections on issues of power in packaged software selection. *Information Systems Journal*, 16(3):215–235.

[Howcroft et al., 2010] Howcroft, D., Light, B., et al. (2010). The social shaping of packaged software selection. *Journal of the Association for Information Systems*, 11(3).

[Illa et al., 2000] Illa, X. B., Franch, X., and Pastor, J. A. (2000). Formalising erp selection criteria. In *Tenth International Workshop on Software Specification and Design. IWSSD-10 2000*, pages 115–122. IEEE.

[Jadhav and Sonar, 2009] Jadhav, A. S. and Sonar, R. M. (2009). Evaluating and selecting software packages: A review. *Information and software technology*, 51(3):555–563.

[Jain et al., 2008] Jain, V., Benyoucef, L., Bennett, D., Deep, A., Guttridge, P., Dani, S., and Burns, N. (2008). Investigating factors affecting erp selection in made-to-order sme sector. *Journal of Manufacturing Technology Management*.

[Keil and Tiwana, 2006] Keil, M. and Tiwana, A. (2006). Relative importance of evaluation criteria for enterprise systems: a conjoint study. *Information Systems Journal*, 16(3):237–262.

[Kontio, 1996] Kontio, J. (1996). A case study in applying a systematic method for cots selection. In *Proceedings of IEEE 18th International Conference on Software Engineering*, pages 201–209. IEEE.

[Kunda and Brooks, 2000] Kunda, D. and Brooks, L. (2000). Identifying and classifying processes (traditional and soft factors) that support cots component selection: a case study. *European Journal of Information Systems*, 9(4):226–234.

[Kusumo et al., 2011] Kusumo, D. S., Zhu, L., Staples, M., and Zhang, H. (2011). A systematic mapping study on off-the-shelf-based software acquisition.

[Labatut et al., 2012] Labatut, J., Aggeri, F., and Girard, N. (2012). Discipline and change: How technologies and organizational routines interact in new practice creation. *Organization studies*, 33(1):39–69.

[Le Blanc and Korn, 1992] Le Blanc, L. A. and Korn, W. M. (1992). A structured approach to the evaluation and selection of case tools. In *Proceedings of the 1992 ACM/SIGAPP symposium on Applied computing: technological challenges of the 1990's*, pages 1064–1069.

[Lee et al., 2002] Lee, J., Lee, K., and Rho, S. (2002). An evolutionary perspective on strategic group emergence: a genetic algorithm-based model. *Strategic management journal*, 23(8):727–746.

[Lehmann and Buxmann, 2009] Lehmann, S. and Buxmann, P. (2009). Pricing strategies of software vendors. *Business & Information Systems Engineering*, 1(6):452–462.

[Leurs and Duggan, 2018] Leurs, B. and Duggan, K. (2018). Proof of concept, prototype, pilot, MVP – what's in a name?

[Lin et al., 2007] Lin, H., Lai, A., Ullrich, R., Kuca, M., McClelland, K., Shaffer-Gant, J., Pacheco, S., Dalton, K., and Watkins, W. (2007). Cots software selection process. In *2007 Sixth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems (ICCBSS'07)*, pages 114–122. IEEE.

[March and Herbert, 2013] March, J. G. and Herbert, A. (2013). Simon1958 organizations. *New York: WileyMarchOrganizations1958*.

[McQueen and Teh, 2000] McQueen, R. and Teh, R. (2000). Insight into the acquisition process for enterprise resource planning software derived from four case studies. *PACIS 2000 Proceedings*, page 49.

[Miller and Yeoh, 2006] Miller, J. and Yeoh, H. C. (2006). Cots acquisition process: incorporating business factors into cots vendor evaluation taxonomies. *Software Process: Improvement and Practice*, 11(6):601–626.

[Miller et al., 2014] Miller, K. D., Choi, S., and Pentland, B. T. (2014). The role of transactive memory in the formation of organizational routines. *Strategic organization*, 12(2):109–133.

[Morera, 2002] Morera, D. (2002). Cots evaluation using desmet methodology & analytic hierarchy process (ahp). In *International Conference on Product Focused Software Process Improvement*, pages 485–493. Springer.

[Motwani et al., 2002] Motwani, J., Mirchandani, D., Madan, M., and Gunasekaran, A. (2002). Successful implementation of erp projects: evidence from two case studies. *International Journal of Production Economics*, 75(1-2):83–96.

[Nelson, 2009] Nelson, R. R. (2009). *An evolutionary theory of economic change*. harvard university press.

[Nelson et al., 2018] Nelson, R. R., Dosi, G., Helfat, C. E., and Winter, S. G. (2018). Modern evolutionary economics: An overview.

[Nelson and Winter, 1982] Nelson, R. R. and Winter, S. G. (1982). The schumpeterian tradeoff revisited. *The American Economic Review*, 72(1):114–132.

[Patel and Hlupic, 2002] Patel, N. and Hlupic, V. (2002). A methodology for the selection of knowledge management (km) tools. In *ITI 2002. Proceedings of the 24th International Conference on Information Technology Interfaces (IEEE Cat. No. 02EX534)*, pages 369–374. IEEE.

[Pentland and Feldman, 2008] Pentland, B. T. and Feldman, M. S. (2008). Designing routines: On the folly of designing artifacts, while hoping for patterns of action. *Information and organization*, 18(4):235–250.

[Pereira and Sousa, 2005] Pereira, C. M. and Sousa, P. (2005). Enterprise architecture: business and it alignment. In *Proceedings of the 2005 ACM symposium on Applied computing*, pages 1344–1345.

[Perez and Rojas, 2000] Perez, M. and Rojas, T. (2000). Evaluation of workflow-type software products: a case study. *Information and software technology*, 42(7):489–503.

[Poon and Yu, 2010] Poon, P.-L. and Yu, Y. T. (2010). Investigating erp systems procurement practice: Hong kong and australian experiences. *Information and Software Technology*, 52(10):1011–1022.

[Rerup and Feldman, 2011] Rerup, C. and Feldman, M. S. (2011). Routines as a source of change in organizational schemata: The role of trial-and-error learning. *Academy of Management Journal*, 54(3):577–610.

[Ribes and Bowker, 2009] Ribes, D. and Bowker, G. C. (2009). Between meaning and machine: Learning to represent the knowledge of communities. *Information and Organization*, 19(4):199–217.

[Salvato and Rerup, 2011] Salvato, C. and Rerup, C. (2011). Beyond collective entities: Multilevel research on organizational routines and capabilities. *Journal of management*, 37(2):468–490.

[Schrödl, 2012] Schrödl, H. (2012). Purchasing cloud-based product-service bundles in value networks-the role of manageable workloads.

[Schumpeter, 1934] Schumpeter, J. (1934). The theory of economic development harvard university press. *Cambridge, MA*.

[Şen and Baraçlı, 2010] Şen, C. G. and Baraçlı, H. (2010). Fuzzy quality function deployment based methodology for acquiring enterprise software selection requirements. *Expert Systems with Applications*, 37(4):3415–3426.

[Simon, 1957] Simon, H. A. (1957). *Administrative Behaviour: a Study of Decision Making Prosesses in Administrative Organization*. The Free Press.

[Storteboom et al., 2017] Storteboom, A., Wondimu, P., Lohne, J., and Lædre, O. (2017). Best value procurement-the practical approach in the netherlands. *Procedia computer science*, 121:398–406.

[Stylianou et al., 1992] Stylianou, A. C., Madey, G. R., and Smith, R. D. (1992). Selection criteria for expert system shells: a socio-technical framework. *Communications of the ACM*, 35(10):30–48.

[Swinglehurst et al., 2010] Swinglehurst, D., Greenhalgh, T., Myall, M., and Russell, J. (2010). Ethnographic study of ict-supported collaborative work routines in general practice. *BMC Health Services Research*, 10(1):1–9.

[Teece, 2007] Teece, D. J. (2007). Explicating dynamic capabilities: the nature and microfoundations of (sustainable) enterprise performance. *Strategic management journal*, 28(13):1319–1350.

[Teece et al., 1997] Teece, D. J., Pisano, G., and Shuen, A. (1997). Dynamic capabilities and strategic management. *Strategic management journal*, 18(7):509–533.

[Teltumbde, 2000] Teltumbde, A. (2000). A framework for evaluating erp projects. *International journal of production research*, 38(17):4507–4520.

[Tewoldeberhan et al., 2002] Tewoldeberhan, T. W., Verbraeck, A., Valentin, E., and Bardonnet, G. (2002). An evaluation and selection methodology for discrete-event simulation software. In *Proceedings of the Winter Simulation Conference*, volume 1, pages 67–75. IEEE.

[Thompson, 1967] Thompson, J. (1967). Organizations in action. 1967. *SHAFRITZ, Jay M.; OTT, J. Steven. Classics of organization theory*, 4.

[Thönes, 2015] Thönes, J. (2015). Microservices. *IEEE software*, 32(1):116–116.

[Tran and Liu, 1997] Tran, V. and Liu, D.-B. (1997). A procurement-centric model for engineering component-based software systems. In *Proceedings Fifth International Symposium on Assessment of Software Tools and Technologies*, pages 70–79. IEEE.

[Tripsas, 1997] Tripsas, M. (1997). Surviving radical technological change through dynamic capability: Evidence from the typesetter industry. *Industrial and corporate Change*, 6(2):341–377.

[Turner and Rindova, 2012] Turner, S. F. and Rindova, V. (2012). A balancing act: How organizations pursue consistency in routine functioning in the face of ongoing change. *Organization Science*, 23(1):24–46.

[Tuş and Adalı, 2019] Tuş, A. and Adalı, E. A. (2019). The new combination with critic and waspas methods for the time and attendance software selection problem. *Opsearch*, 56(2):528–538.

[van Weele, 2014] van Weele, A. (2014). *Purchasing and Supply Chain Management*. Cengage Learning.

[Verville et al., 2005] Verville, J., Bernadas, C., and Halingten, A. (2005). So you're thinking of buying an erp? ten critical factors for successful acquisitions. *Journal of Enterprise Information Management*.

[Verville and Halingten, 2003] Verville, J. and Halingten, A. (2003). A six-stage model of the buying process for erp software. *Industrial Marketing Management*, 32(7):585–594.

[Verville and Halingten, 2002] Verville, J. C. and Halingten, A. (2002). A qualitative study of the influencing factors on the decision process for acquiring erp software. *Qualitative Market Research: An International Journal*.

[Wegmann et al., 2005] Wegmann, A., Regev, G., and Loison, B. (2005). Business and it alignment with seam. In *1st International Workshop on Requirements Engineering for Business Need, and IT Alignment (REBNITA2005)*, number CONF, pages 74–84. University of New South Wales Press.

[Wei et al., 2005] Wei, C.-C., Chien, C.-F., and Wang, M.-J. J. (2005). An ahp-based approach to erp system selection. *International journal of production economics*, 96(1):47–62.

[Wiegers and Beatty, 2013] Wiegers, K. and Beatty, J. (2013). *Software requirements*. Pearson Education.

[Winter, 1986] Winter, S. G. (1986). The research program of the behavioral theory of the firm: Orthodox critique and evolutionary perspective. *Handbook of behavioral economics*, 1:151–188.

[Winter, 2000] Winter, S. G. (2000). The satisficing principle in capability learning. *Strategic management journal*, 21(10-11):981–996.

[Winter, 2003] Winter, S. G. (2003). Understanding dynamic capabilities. *Strategic management journal*, 24(10):991–995.

[Winter, 2006] Winter, S. G. (2006). Toward a neo-schumpeterian theory of the firm. *Industrial and Corporate Change*, 15(1):125–141.

[Wright, 1990] Wright, P. (1990). Choosing a computer based instructional support system: An evaluation/selection model. *Computers & Education*, 14(3):217–225.

[Yazgan et al., 2009] Yazgan, H. R., Boran, S., and Goztepe, K. (2009). An erp software selection process with using artificial neural network based on analytic network process approach. *Expert systems with applications*, 36(5):9214–9222.

[Yusuf et al., 2006] Yusuf, Y., Gunasekaran, A., and Wu, C. (2006). Implementation of enterprise resource planning in china. *Technovation*, 26(12):1324–1336.

[Zahra et al., 2006] Zahra, S. A., Sapienza, H. J., and Davidsson, P. (2006). Entrepreneurship and dynamic capabilities: A review, model and research agenda. *Journal of Management studies*, 43(4):917–955.

[Zaidan et al., 2015] Zaidan, A., Zaidan, B., Hussain, M., Haiqi, A., Kiah, M. M., and Abdulnabi, M. (2015). Multi-criteria analysis for os-emr software selection problem: A comparative study. *Decision Support Systems*, 78:15–27.

[Zaini et al., 2015] Zaini, R., Quqandi, E., and Guezguez, W. (2015). A multi-criteria decision making model for software selection to build e-portfolio. In *2015 Fifth International Conference on e-Learning (econf)*, pages 131–134. IEEE.

[Zollo and Winter, 2002] Zollo, M. and Winter, S. G. (2002). Deliberate learning and the evolution of dynamic capabilities. *Organization science*, 13(3):339–351.

[Zott, 2003] Zott, C. (2003). Dynamic capabilities and the emergence of intraindustry differential firm performance: insights from a simulation study. *Strategic management journal*, 24(2):97–125.

Appendices

Nr.	Routine	Subroutine	References
1	Gather requirements	Identify and involve stakeholders	e.g. [Harnisch, 2014]
2	Gather requirements	Analyze affected business processes	e.g. [Ayağ and Özdemir, 2007]
3	Gather requirements	Organize focus day with end-users	e.g. [Howcroft and Light, 2006]
4	Gather requirements	Brainstorming with team	e.g. [Tewoldeberhan et al., 2002]
5	Gather requirements	Define needs and expectations	e.g. [Miller and Yeoh, 2006]
6	Gather requirements	Define requirements	e.g. [Verville and Halingten, 2003]
7	Gather requirements	Review requirements	e.g. [Lin et al., 2007]
8	Research the market	Conduct market research	e.g. [Bandor, 2006]
9	Research the market	Conduct vendor surveys	e.g. [Tewoldeberhan et al., 2002]
10	Research the market	Conduct vendor interviews	e.g. [Wei et al., 2005]
11	Research the market	Research vendor white papers	e.g. [Wright, 1990]
12	Research the market	Research technical specifications	e.g. [Bandor, 2006]
13	Research the market	Representation at IT conferences	e.g. [Tran and Liu, 1997]
14	Research the market	Experiment with trial versions	e.g. [Zaidan et al., 2015]
15	Research the market	Examination by consultants	e.g. [Bernroider and Koch, 2001]
16	Research the market	Create longlist of alternatives	e.g. [Ayağ and Özdemir, 2007]
17	Pre-select vendors	Send and review RFPs	e.g. [Schrödl, 2012]
18	Pre-select vendors	Set first screening (exit) criteria	e.g. [Howcroft and Light, 2006]
19	Pre-select vendors	Identify selection criteria	e.g. [Adhikari et al., 2004]
20	Pre-select vendors	Analysis of prototype	e.g. [Kontio, 1996]
21	Pre-select vendors	Vendor demonstrations	e.g. [Patel and Hlupic, 2002]
22	Pre-select vendors	Contact other users of the package	e.g. [Chau, 1995]
23	Pre-select vendors	Create use case scenarios	e.g. [Lin et al., 2007]
24	Pre-select vendors	Interview end-users	e.g. [Harnisch, 2014]
25	Pre-select vendors	Interview vendors	e.g. [Wright, 1990]
26	Pre-select vendors	Review RFPs	e.g. [Miller and Yeoh, 2006]
27	Pre-select vendors	Review vendor information	e.g. [Lin et al., 2007]
28	Pre-select vendors	Involve process experts	e.g. [Teltumbde, 2000]
29	Pre-select vendors	Recruit external consultants	e.g. [Poon and Yu, 2010]
30	Pre-select vendors	Map core processes	e.g. [Teltumbde, 2000]
31	Pre-select vendors	Analysis using qualitative criteria	e.g. [Colombo and Francalanci, 2004]
32	Pre-select vendors	Discuss pros and cons	e.g. [Kontio, 1996]
33	Pre-select vendors	Create shortlist by elimination	e.g. [Illa et al., 2000]
34	Select vendor	Manage catalogues of packages	e.g. [Grau et al., 2004]
35	Select vendor	Categorize criteria	e.g. [Perez and Rojas, 2000]
36	Select vendor	Weigh criteria	e.g. [Collier et al., 1999]
37	Select vendor	Define hierarchy of criteria	e.g. [Grau et al., 2004]
38	Select vendor	Define relationships among criteria	e.g. [Şen and Baraçlı, 2010]
39	Select vendor	Revise criteria	e.g. [Le Blanc and Korn, 1992]
40	Select vendor	Vendor demonstrations	e.g. [Lin et al., 2007]
41	Select vendor	Obtain package information	e.g. [Verville and Halingten, 2003]
42	Select vendor	Score packages	e.g. [Teltumbde, 2000]
43	Select vendor	Rank packages	e.g. [Howcroft et al., 2010]
44	Select vendor	Consider risk factors	e.g. [Bandor, 2006]
45	Select vendor	Consider intangible factors	e.g. [Bandor, 2006]
46	Select vendor	Filter out marketing claims and hypes	e.g. [Lin et al., 2007]
47	Select vendor	Gather opinions and reviews	e.g. [Illa et al., 2000]
48	Select vendor	Compare packages	e.g. [Howcroft et al., 2010]
49	Select vendor	Discussion within team	e.g. [Lin et al., 2007]
50	Select vendor	Evaluate vendors	e.g. [Schrödl, 2012]
51	Select vendor	Use evaluation tool to evaluate	e.g. [Şen and Baraçlı, 2010]
52	Select vendor	Use AHP to evaluate	e.g. [Wei et al., 2005]
53	Select vendor	Use WSM to evaluate	e.g. [Wright, 1990]
54	Select vendor	Perform sensitivity analysis	e.g. [Miller and Yeoh, 2006]
55	Select vendor	Create list of data and evaluations	e.g. [Lin et al., 2007]
56	Select vendor	Review top-ranked alternatives	e.g. [Miller and Yeoh, 2006]
57	Select vendor	Trade-offs analysis	e.g. [Lin et al., 2007]
58	Select vendor	Create risk management plan	e.g. [Miller and Yeoh, 2006]
59	Select vendor	Present evaluation results	e.g. [Morera, 2002]
60	Select vendor	Re-evaluate packages	e.g. [Lin et al., 2007]
61	Select vendor	Prepare selection proposal	e.g. [Illa et al., 2000]
62	Select vendor	Determine selection approach	e.g. [Poon and Yu, 2010]
63	Select vendor	Select software that was evaluated best	e.g. [Morera, 2002]
64	Select vendor	Static and dynamic investment method	e.g. [Bernroider and Koch, 2001]
65	Select vendor	Use a selection tool	e.g. [Hlupic and Mann, 1995]
66	Conduct pilot	Pilot testing / Prototyping	e.g. [Poon and Yu, 2010]
67	Negotiate contract	Negotiation with vendors	e.g. [Colombo and Francalanci, 2004]
68	Negotiate contract	Make estimation of costs	e.g. [Illa et al., 2000]
69	Negotiate contract	Create implementation plan	e.g. [Illa et al., 2000]
70	Negotiate contract	Create contingency plan	e.g. [Illa et al., 2000]
71	Negotiate contract	Final approval from top management	e.g. [Verville and Halingten, 2003]

TABLE 1: Software selection routines discovered in the literature research

Routine	**Subroutine**	**Subsubroutine**
1. Initiate project	1.1. Identify need	1.1.1. Recognize arising need from the business
		1.1.2. Create base of support to find solution
		1.1.3. Translate need to IT
	1.2 Make reuse, make or buy decision	1.2.1. Check for in-house solutions
		1.2.2. Consider process reengineering solutions
		1.2.3. Consider developing software
		1.2.4. Decide to buy software
	1.3 Set up project organization	1.3.1. Involve projectmanager
		1.3.2. Write first version business case
		1.3.3. Involve right stakeholders in team
		1.3.4. Hire consultants
		1.3.5. Plan project
		1.3.6. Decide procurement strategy
		1.3.7. Kick off project
2. Gather requirements	2.1. Analyze impact of new software	2.1.1. Make a Business Impact Analysis
		2.1.2. Make a Privacy Impact Analysis
	2.2. Draw up requirements	2.2.1. Gather business/IT requirements through different channels
		2.2.2. Collect specialists' requirements
		2.2.3. Draw up requirements
		2.2.4. Store requirements in one document
	2.3. Specify requirements	2.3.1. Distill proofpoints from requirements
		2.3.2. Define knock-out criteria (minimal requirements)
		2.3.3. Define acceptance criteria
		2.3.4. Prioritize requirements
		2.3.5. Determine scoring mechanism
		2.3.6. Refine business case
3. Research the market	3.1. Conduct market research	3.1.1. Browse the internet
		3.1.2. Go to events
		3.1.3. Converse with peers
		3.1.4. Consult software experts
	3.2. Request vendor information	3.2.1. Organize QAs with possible vendors
		3.2.2. Send out RFI
		3.2.3. Conduct reference visits
		3.2.4. Post tender on tender publication website
	3.3. Create longlist	3.3.1. Classify options in Kraljix Matrix
		3.3.2. Create longlist
4. Pre-select vendors	4.1. Conduct additional research	4.1.1. Research vendors
		4.1.2. Talk to peers
		4.1.3. Organize QAs with longlisted vendors
		4.1.4. Organize vendor presentations/demonstrations
	4.2. Enter RFx process	4.2.1. Send out RFx to longlist
	4.3. Assess vendors	4.3.1. Assess offers
		4.3.2. Assess vendors
		4.3.3. Assess solutions
		4.3.4. Score individually
		4.3.5. Apply knock-out criteria
	4.4. Create shortlist	4.4.1. Rank vendors
		4.4.2. Consultants advise the project team
		4.4.3. Discuss offers/assessments with stakeholders
		4.4.4. Create shortlist
5. Select vendor	5.1. Enter RFx process	5.1.1. Send out RFx to shortlist
	5.2. Interact with vendors	5.2.1. Organize QAs with shortlisted vendors
		5.2.2. Brief vendors about expectations
		5.2.3. Organize vendor presentations/demonstrations
		5.2.4. Organize vendor pitches
		5.2.5. Have vendors create a prototype
	5.3. Assess vendors	5.3.1. Assess offers
		5.3.2. Assess vendors
		5.3.3. Assess solutions
		5.3.4. Assess implementation plan
		5.3.5. Score individually
		5.3.6. Make reference visits
		5.3.7. Find a consensus with stakeholders
		5.3.8. Rank vendors based on average scores
		5.3.9. Finalize business case
	5.4. Make a preliminary selection	5.4.1. Process experts/consultants advise the project team
		5.4.2. Project team advises budget holder
		5.4.3. Approve selection
		5.4.4. Inform vendors
		5.4.5. Continue with one or two best vendors
6. Conduct pilot	6.1. Pilot the software	6.1.1. Agree with vendor(s) on acceptance criteria
		6.1.2. Supply vendor(s) with business processes to support
		6.1.3. Set up PoC with vendors (s)
		6.1.4. Organize Hack-a-thon
		6.1.5. Test integration with as-is landscape
	6.2. Make final assessment	6.2.1. Question users on experiences with pilot
		6.2.2. Score PoC individually
		6.2.3. Assess PoC(s) on acceptance criteria
		6.2.4. Test impact on ROI and TCO
		6.2.5. Perform cost-benefit analysis
		6.2.6. Discuss points of improvement with vendor(s)
	6.3. Make selection final	6.3.1. Send out RFP to vendor(s)
		6.3.2. Select highest scoring vendor
		6.3.3. Approve selection
7. Negotiate contract	7.1. Enter negotiations	7.1.1. Enter negotiations with vendor(s)
		7.1.2. Negotiate price and contract details
		7.1.3. Continue with second best vendor if negotiations fail
	7.2. Close the deal	7.2.1. Draw up contract
		7.2.2. Enter an escrow agreement
		7.2.3. Sign contract formally
Post-selection	Select implementation partner	
	Implement software	
	Set up strategic consultations	
	Set up tactical consultations	
	Set up operational consultations	
	Convert data	
	Train users	
	Handover to operational processes	
	Dismantle previous system	
	Manage vendors	
	Manage contracts	
	Manage licenses	
	Conduct maintenance	

TABLE 2: Software selection routines distilled from interviews

Abbreviation	Meaning
IT	Information Technology
Q&A	Question and Answer
RFx	RFI, RFQ, RFP
RFI	Request for Information
RFQ	Request for Quotation
RFP	Request for Proposal
PoC	Proof of Concept
ROI	Return on Investment
TCO	Total Cost of Ownership

TABLE 3: Meaning of abbreviations in table 2

Interview Guide Master Thesis Yorick van der Vorm: Software Selection

Interviewer: Yorick van der Vorm
Organization: Universiteit Leiden

General software selection process:

1. Trigger for software selection project to start
2. Stakeholder and requirement identification
3. Screening of potential software packages/vendors
4. Evaluation of remaining packages/vendors
5. Selection of software

Pitch

Since the digital transformation started, software has become indispensable to every modern organization. But how does one select "the right" software? My name is Yorick van der Vorm and I am a Master student ICT in Business and the Public Sector at the University of Leiden. For my Master Thesis I am researching the software selection capability of organizations. My goal is to gain a deeper and better understanding of software selection in practice.

I am looking for people who have been involved in software selection projects in the past 2 years, and who would like to learn more on how other organizations are practicing software selection. Does this sound like you and are you willing to have a one-hour interview with me? Please comment / reply / message me, so I can reach out to set up an interview. If you know anybody in your network who might be relevant, a referral is also much appreciated.

PS. The interview will be conducted virtually due to the current COVID-19 situation.

Participants

Interviewee	Phone Nr.	Organization	Role

Introduction

First of all, thank you so much for making some time for this interview, I really appreciate it. My name is Yorick van der Vorm, I am 24 years old and I study ICT in Business and the Public Sector at the University of Leiden. For my Master Thesis I am researching Software Selection, more particularly the way in which organizations select their software. I want to map out the process steps and find best practices, which I will use to write a recommendation on how organizations should ideally execute their software selection projects. Since you play a part in software selection projects at your organization, I would love to interview you about that process. Before diving in, I would like to ask you again: do you give your consent to me for recording, collecting, analyzing, and using the data you are giving me in this interview? Intro uitbreiden met wat ik leuk vind aan

I would like to ask you some introductory questions first:

What is your name and for what organization do you work?

What is your function within the organization?

How many software selection projects were you involved in?

What role did you have in those software selection projects?

What type of software packages have been selected in projects you were working on?

Interview

What are the main phases in a software selection project at your organization?

Which stakeholders/roles are involved in a software selection project?

When is a software selection project considered a success?

The following questions will be based on the answer given to the first question but will most likely be in line with the general software selection process mentioned on page 1.

Step 1. Trigger for software selection project to start
What triggers a software selection project to start?

Which process steps are taken in this phase?

How are the steps taken? Go into detail.

Step 2. Requirement identification
Which steps are taken in the requirement identification phase?

How are the steps taken? Go into detail.

Step 3. Screening of potential software packages/vendors
Which steps are taken in the screening phase?

How are the steps taken? Go into detail.

Step 4. Evaluation of remaining packages/vendors
Which steps are taken in the evaluation phase?

How are the steps taken? Go into detail.

Step 5. Selection of software
Which process steps are taken in the phase of making the final decision?

How are the steps taken? Go into detail.

Concluding

Which dimensions of evaluation criteria do you consider in a software selection project? An example could be functionality or vendor reputation

Do you use a standardized methodology for software selection? If so, can you send that to me?

What happens after a software package has been selected? Negotiations? Implementation?

In practice we often see that there is a difference between how a project is supposed to go, and how it goes in reality. Do you recognize this in your organization?

These were all my questions. Do you have anything to add? Any remarks? Anything I forgot to ask?

Thank you so much for your time and the interview. This will truly help me a lot with my research. If you want, I can send you my findings after I finish my Thesis.

Software Selection

Start of Block: Opening message

Dear participant,

Thank you for taking the time to participate in this anonymous survey. This research investigates software selection. We are specifically interested in the process steps your organization goes through when selecting software and the effect it has on your organization.

This survey is meant for people that are or have been involved in software selection projects in recent years. By participating in this survey, you will be given the opportunity to get insights into the results.

This survey is fully anonymous and all data will be treated fully confidential. There are no right or wrong answers, therefore we would like to encourage you to answer all questions. If there are any questions please contact me at y.s.van.der.vorm@umail.leidenuniv.nl .

It is estimated that the survey will take around 15 minutes to answer.

Thank you very much for your participation!
Kind regards,

Yorick van der Vorm
Master student ICT in Business at Leiden University

Prof.dr.ir. J.M.W. Visser
Leiden University

T.D. Offerman MSc
Leiden University

End of Block: Opening message

Start of Block: Introductory questions

What type of organization are you in?

▼ Consumer products ... Other

What is your function within the organization?

_____

How many software selection projects have you been involved in?

- 0
- 1 - 2
- 3 - 5
- 6 - 10
- 11 - 20
- 21+

What is the last time you were involved in a software selection project?

- I am currently doing a software selection project
- 1 year ago
- 2 years ago
- 3 years ago
- 4 years ago
- 5 years ago
- More than 5 years ago
- I have never been involved in a software selection project

End of Block: Introductory questions

Start of Block: Role in software selection

What is/has been your role(s) in software selection projects? Check all applicable answers:

- [ ] Budget holder
- [ ] Problem owner
- [ ] Project manager
- [ ] Enterprise architect
- [ ] Solution architect
- [ ] Business representative
- [ ] IT representative
- [ ] Procurement officer
- [ ] (Key) user
- [ ] Process expert
- [ ] Security expert
- [ ] Privacy expert
- [ ] Risk expert
- [ ] Data expert
- [ ] Legal expert
- [ ] External consultant
- [ ] Internal consultant
- [ ] Other _____

End of Block: Role in software selection

Start of Block: Software selection general

How many people does a software selection team typically consist of?

- 1
- 2 - 3
- 4 - 6
- 7 - 10
- 11 - 14
- 15+

Which of these roles, or comparable roles, are involved in software selection projects? Check all applicable answers:

- [ ] Budget holder
- [ ] Problem owner
- [ ] Project manager
- [ ] Enterprise architect
- [ ] Solution architect
- [ ] Business representative
- [ ] IT representative
- [ ] Procurement officer
- [ ] (Key) user(s)
- [ ] Process expert
- [ ] Security expert
- [ ] Privacy expert
- [ ] Risk expert
- [ ] Data expert
- [ ] Legal expert
- [ ] External consultant
- [ ] Internal consultant
- [ ] Other: _____

How many external consultants or external employees are typically involved in a software selection project?

- 0
- 1 - 3
- 4 - 6
- 7 - 9
- 10 +

*

Typically, what is the duration of a software selection project? (excluding implementation, etc.)

Enter the number of months:

_____

X→

What is the typical Total Cost of Ownership (TCO) of the software packages you have selected? TCO based on three years:

- ○ No idea
- ○ €0 - €10.000
- ○ €10.001 - €50.000
- ○ €50.001 - €200.000
- ○ €200.001 - €500.000
- ○ €500.001 - €1.000.000
- ○ €1.000.001 - €5.000.000
- ○ €5.000.001 - €10.000.000
- ○ €10.000.001 - €20.000.000
- ○ €20.000.001+

What are the reasons for the start of a software selection project? Check all applicable answers:

- [ ] Dissatisfaction with current software systems
- [ ] Current systems end-of-life
- [ ] Contract expiration with current vendor
- [ ] Implementation of new products
- [ ] Implementation of new processes
- [ ] Innovation
- [ ] The need to increase performance
- [ ] Execution of organization's (digitalization) strategy
- [ ] Seeing software opportunities in other places (i.e. the market, competitors)
- [ ] Striving for operational excellence
- [ ] Anticipating market changes
- [ ] Complying to changing laws and regulations
- [ ] Other: _____

In your opinion, what is the maturity level of software selection in your organization?

Curtis, B., Hefley, B., & Miller, S. (2009). People capability maturity model (P-CMM) version 2.0

- ○ Level 1: Initial
- ○ Level 2: Managed
- ○ Level 3: Defined
- ○ Level 4: Quantitatively managed
- ○ Level 5: Optimizing

End of Block: Software selection general

Start of Block: Perceived effects

In your opinion, what effect does a successful software selection have on the following topics?

	Extremely negative	Somewhat negative	No effect	Somewhat positive	Extremely positive
Minimizing implementation risks	○	○	○	○	○
Reduction of incidents	○	○	○	○	○
Adoption rate	○	○	○	○	○
User satisfaction	○	○	○	○	○
Overview of IT Architecture	○	○	○	○	○
IT rationalisation	○	○	○	○	○
System integration	○	○	○	○	○
Reaching the goals of the business	○	○	○	○	○
Profit margins	○	○	○	○	○
Cost reduction	○	○	○	○	○
Relationships with software vendors	○	○	○	○	○
Operational excellence	○	○	○	○	○
Creation/transformation of resources	○	○	○	○	○
Creation/transformation of processes	○	○	○	○	○
Compliance to laws and regulations	○	○	○	○	○
Other:	○	○	○	○	○

Other:	○	○	○	○	○

End of Block: Perceived effects

Start of Block: Routines

The following questions will be aimed at finding out more about the process steps your organization executes in a software selection project.

You will be asked how often your organization typically executes the step and how important you think the step is for the successful completion of a software selection project.

End of Block: Routines

Start of Block: Identify need

Does your organization clearly identify the business need? (e.g. analyze the problem or translate the need to IT)

- ○ Never
- ○ Sometimes
- ○ Most of the time
- ○ Always

X→

In your opinion, how important is identifying the business need in successfully completing a software selection project?

- ○ Not at all important
- ○ Slightly important
- ○ Moderately important
- ○ Very important
- ○ Extremely important

End of Block: Identify need

Start of Block: Reuse, make or buy

Does your organization discuss whether to reuse, make or buy software? (e.g. check in-house solutions)

- Never
- Sometimes
- Most of the time
- Always

In your opinion, how important is discussing the reuse, make or buy decision in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Reuse, make or buy

Start of Block: Project organization

Does your organization set up a project organization? (e.g. involve a project manager or create a business case)

- Never
- Sometimes
- Most of the time
- Always

In your opinion, how important is setting up a project organization in successfully completing a software selection project?

- ○ Not at all important
- ○ Slightly important
- ○ Moderately important
- ○ Very important
- ○ Extremely important

End of Block: Project organization

Start of Block: Impact analysis

Does your organization analyze the impact new software will have on the organization? (e.g. doing a Business Impact Analysis or Privacy Impact Analysis)

- ○ Never
- ○ Sometimes
- ○ Most of the time
- ○ Always

In your opinion, how important is analyzing the impact on the organization in successfully completing a software selection project?

- ○ Not at all important
- ○ Slightly important
- ○ Moderately important
- ○ Very important
- ○ Extremely important

End of Block: Impact analysis

Start of Block: Draw up requirements

Does your organization draw up requirements? (e.g. specifications/criteria/user stories)

- Never
- Sometimes
- Most of the time
- Always

In your opinion, how important is drawing up requirements in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Draw up requirements

Start of Block: Specify requirements

Does your organization specify requirements? (e.g. defining knock-out criteria, prioritizing requirements)

- Never
- Sometimes
- Most of the time
- Always

In your opinion, how important is specifying requirements in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Specify requirements

Start of Block: Market research

Does your organization conduct market research? (e.g. consulting research companies and going to events)

- Never
- Sometimes
- Most of the time
- Always

In your opinion, how important is conducting market research in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Market research

Start of Block: Request information

Does your organization request information from software vendors? (e.g. through reference visits or Q&A sessions)

- Never
- Sometimes
- Most of the time
- Always

In your opinion, how important is requesting information from software vendors in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Request information

Start of Block: Longlist

Does your organization create a longlist of possible software vendors?

- Never
- Sometimes
- Most of the time
- Always

In your opinion, how important is creating a longlist in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Longlist

Start of Block: Conduct additional research

Does your organization conduct additional research on the longlisted vendors? (e.g. reference visits or vendor presentations/demonstrations)

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is conducting additional research in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Conduct additional research

Start of Block: RFx process

Does your organization enter an RFI and/or RFQ and/or RFP process with the longlisted vendors?

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is entering an RFx process with longlisted vendors in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: RFx process

Start of Block: Assess vendors

Does your organization assess the vendors on the longlist? (e.g. scoring vendors/offers or applying knock-out criteria)

- ○ Never
- ○ Sometimes
- ○ Most of the time
- ○ Always
- ○ Not applicable

In your opinion, how important is assessing the vendors on the longlist in successfully completing a software selection project?

- ○ Not at all important
- ○ Slightly important
- ○ Moderately important
- ○ Very important
- ○ Extremely important

End of Block: Assess vendors

Start of Block: Create shortlist

Does your organization create a shortlist?

- ○ Never
- ○ Sometimes
- ○ Most of the time
- ○ Always

In your opinion, how important is creating a shortlist in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Create shortlist

Start of Block: RFx process shortlist

Does your organization enter an RFI and/or RFQ and/or RFP process with the shortlisted vendors?

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is entering an RFx process with shortlisted vendors in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: RFx process shortlist

Start of Block: Vendor interaction

Does your organization interact with vendors on the shortlist? (e.g. vendor demonstrations, conversations or pitches)

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is interacting with shortlisted vendors in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Vendor interaction

Start of Block: Assess vendors shortlist

Does your organization assess vendors on the shortlist? (e.g. scoring offers/products and discussing vendors)

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is assessing shortlisted vendors in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Assess vendors shortlist

Start of Block: Preliminary selection

Does your organization make a preliminary selection? (e.g. selecting one or more vendors to enter a pilot or negotiations)

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is making a preliminary selection in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Preliminary selection

Start of Block: Enter pilot

Does your organization execute a pilot with the (preliminarily) selected software? (e.g. doing a Proof of Concept or prototyping)

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is piloting software in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Enter pilot

Start of Block: Make final assessment

Does your organization do a final assessment based on the pilot? (e.g. questioning users and scoring ability to integrate)

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is making a final assessment in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Make final assessment

Start of Block: Make selection final

Does your organization make the selection final? (e.g. getting approval from all necessary instancies)

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is making the selection final in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Make selection final

Start of Block: Contract negotiations

Does your organization enter contract negotiations with one or more vendors?

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is entering contract negotiations in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Contract negotiations

Start of Block: Close the deal

Does your organization close the deal with a software vendor? (e.g. through a formal signing moment)

- Never
- Sometimes
- Most of the time
- Always
- Not applicable

In your opinion, how important is closing the deal in successfully completing a software selection project?

- Not at all important
- Slightly important
- Moderately important
- Very important
- Extremely important

End of Block: Close the deal

Start of Block: Tools/documentation

Does your organization use a standardized methodology for software selection? If yes, can you briefly describe it?

- Yes _____
- No
- Unknown

Do you use any tools other than Office (Office365, Google Docs) in your software selection? If yes, which ones? (e.g. SelectHub or GetApp)

- Yes _____
- No

Does your organization typically use any documentation in software selection projects? If yes, what and where during the selection?

	Initiating project	Gathering requirements	Orienting on the market	Making a pre-selection	Making a (preliminary) selection	Piloting	Contracting
Business case template	☐	☐	☐	☐	☐	☐	☐
Business Impact Analysis template	☐	☐	☐	☐	☐	☐	☐
Privacy Impact Analysis template	☐	☐	☐	☐	☐	☐	☐
Requirements template	☐	☐	☐	☐	☐	☐	☐
Research reports	☐	☐	☐	☐	☐	☐	☐
Scoring template	☐	☐	☐	☐	☐	☐	☐
RFI/RFQ/RFP template	☐	☐	☐	☐	☐	☐	☐
Contract template	☐	☐	☐	☐	☐	☐	☐
Pricing/TCO template	☐	☐	☐	☐	☐	☐	☐
Other:	☐	☐	☐	☐	☐	☐	☐

End of Block: Tools/documentation

Start of Block: Additional comments

Did we miss anything crucial you want to share in relation to software selection?
Please do not write down any personal information.

_____

End of Block: Additional comments

Universiteit Leiden

Consent form

For the research *The execution of continuous software selection,* it is necessary to use your personal data. To use this data during our research we need your consent.

What data are being used?

Everything that is said during an interview and/or answered to a survey question and/or collected from shared internal documentation.

What happens if I change my mind?

If you change your mind, you can send an e-mail to *y.s.vandervorm@hotmail.com* with a short message indicating that you want your personal data to be removed. Your name will be permanently deleted from the collected data. Any other information that can be traced back to you will also be permanently deleted.

What will be done with my data after the research project Software Selection as a Dynamic Capability?

Your data will be stripped of your name and other information that can identify you, *6 month* after the research is concluded.