AALBORG UNIVERSITET
STUDENTERRAPPORT

CAUSAL MAPPING OF SOFTWARE PROJECT RISKS IN AGILE SOFTWARE TEAMS

Master's Thesis
IT Design and Application Development

David Kinnberg Hein

Supervised by John Stouby Persson

June 15th, 2022

Summary

Denne specialerapport vil gøre brug af action case tilgangen, som den videnskabelige tilgang til at undersøge, hvordan agile software teams kan forklare risici og deres forskelligartede årsagsforklaringer. Action case tilgangen er en hybrid imellem de to videnskabelige tilgange casestudier og aktionsforskning. Dette medfører, at forskeren tilegner sig forklaringskraften fra casestudier og aktionsforsknings egenskab til at intervenere i en organisationskontekst. Igennem udførelsen af dette studie blev der samarbejdet med et agilt software team, der har deres daglige gang i en dansk bank og som arbejder ud fra Scaled Agile Framework (SAFe), hvilket også udgør konteksten for specialerapporten. Samarbejdet har foregået hen over fem måneder, hvor der har været rig mulighed for at indsamle relevant data fra softwareteamet. Der er blevet gjort brug af tre forskellige typer dataindsamlingsmetoder til at udforske problemformuleringen. Disse metoder inkluderer semi-struktureret interviews af 10 holdmedlemmer, der repræsenterer de seks roller i teamet og observation af sprint retrospective og PI-planning møder, samt dokumentation, der omhandler risici som de tidligere har observeret. Der er blevet gjort brug af flere forskellige typer datakilder for at opnå et højere niveau af data triangulering.

Forskeren begyndte samarbejdet med softwareteamet ved at opstille et møde med en repræsentant fra softwareteamet, hvor repræsentanten italesatte, at de havde udfordringer med at lære af risici, som de har oplevet under tidligere projekter. Dette ledte til, at forskeren eftersøgte et værktøj til at forklare og visualisere holdmedlemmernes opfattelser af de risici, som de har erfareret under tidligere projekter. I denne forbindelse blev kausalkortlægning valgt som værktøjet til at forklare tidligere risici. Kausalkortlægning indebærer, at forskellige koncepter, i dette tilfælde risici, bliver forbundet ud fra en antagelse om, at en givet risiko kan forårsage andre risici. Risicienes indbyrdes kausalitet bliver visualiseret igennem pile, der forbinder risiciene med hinanden og antyder at der er identificeret et kausalt forhold imellem to eller flere risici. I forhold til fremførelsen af de kausale kort, blev det fremlagt, at der med fordel kunne blive fremført kausale kort for hver enkelt af de seks roller i softwareteamet, fremfor at fokusere mere holistisk på hele software teamets opfattelser af risici. Dette blev besluttet for at skabe en mere alsidig udlægning af et software teams' risici, hvor alle rollers opfattelser bliver inkluderet og for at kunne udlægge eventuelle forskelle og ligheder imellem rollerne og indbyrdes i rollerne.

Da de seks forskellige kausale kort var blevet konstrueret, afholdte forskeren individuelle feedback møder med seks repræsentanter fra hver rolle i softwareteamet. Her blev der indsamlet

feedback med henblik på indholdet af kortene, det vil sige de identificerede risici og deres indbyrdes kausalitet. Dette reducerede forskerens usikkerhed omkring troværdigheden ved forbindelserne imellem den identificerede risici i de kausale kort. Herefter blev alle de kausale kort præsenteret for alle rolle indehavere i softwareteamet, hvor der blev indsamlet feedback i forhold til anvendeligheden af kortene indenfor software teamets kontekst. I denne forbindelse blev de kausale kort identificeret som værende brugbare værktøjer til vidensdeling, som start på dokumentation af risici, evaluerings - og diskussionsværktøj under sprint retrospective møder, samt som kommunikationsværktøj til at konkretisere specifikke risici på en mere overbevisende måde til ledelsen. Her menes der mere overbevisende end risikolister, der udelukkende fokuserer på risici isoleret fra hinanden, hvor kausale kort fremfører et dynamisk og systematisk overblik over identificeret risici, samt tager højde for risicis potentielle kausale forhold.

Abstract

This study demonstrates an action case study within the context of an agile software development team practicing the Scaled Agile Framework (SAFe), where the researcher collaborated with said software team over a duration of five months. The goal of this thesis is to explain the diverse project risks and risk causes residing in a software team. The causal mapping technique is selected as the technique to explain the identified risks and showcase the interconnectedness between the risks. The researcher constructed a causal map for each of the six roles that exist within the software team, which was decided to illuminate the similarities and differences between the different roles' perceptions of project risks. To increase the level of data triangulation, the researcher made use of semi-structured interviews, observation of a sprint retrospective and PI-planning meeting, as well as documents describing the software team's previously risks from previous projects. The researcher received individual feedback on the content of the maps from a representative from each role and received feedback on the applicability of the causal maps during a workshop. This was to accommodate with the insecurity surrounding the trustworthiness of the causal maps. The researcher argues that the causal maps provide a more dynamic and systematic visualization of identified risks than that of risk lists, where each risk is considered in isolation of each other. The software team identified a potential use of the causal maps, in terms of using them as a post-evaluation tool during sprint retrospectives, knowledge sharing tool, starting point for documentation of risks and as a tool for showcasing the implications of risks more convincingly to management.

Indholdsfortegnelse

1 Introduction

The main goal of this thesis is to understand and explain the causal relationships between project risks and the role owner's perception of why the risk occurred, as well as the patterns among the role owners' perceptions within an agile software team. To achieve this goal the researcher utilizes the causal mapping technique, derived from (Laukkanen, 1994; Ackermann and Alexander, 2016). The application of causal mapping as a means of explaining different role owners' perceptions of causes to project risks in an agile software team, has not been investigated in the IS research, at least not to the knowledge of the researcher. It is also evident that the use of causal mapping in terms of risk management has exclusively been used to visualize a holistic view of a software team's perceptions or isolated to the perceptions of a single role e.g., Product Owner or Scrum Master (Al-Shehab, Hughes and Winstanley, 2004; Ackermann and Eden, 2005; Williams, Ackermann and Eden, 2013). This probes for further examination of the application of causal mapping as an explanatory tool within the context of an agile software team. The focus on different roles' perceptions within an agile software team is adopted from (Ghobadi and Mathiassen, 2014). Moreover, this thesis is the result of a cooperation with an agile software team in an anonymous bank. The software team will be referred to as "Alpha" and the anonymous bank will be referred to as *Estate Bank*, as well as act as the context of the thesis. On account of the goal of this thesis and its context, a problem statement has been formulated:

Problem statement: *How can an agile software development team explain diverse causes of project risks?*

The researcher was motivated by the cooperation with Alpha to conduct this thesis, as the software team identified having an issue with learning from risks they have experienced during previous risks. This thesis will therefore present the use of the causal mapping method to identify and analyze risks for software teams. In addition, the method may prove to hold the potential as a tool for knowledge sharing of risks, documentation of risks, discussion, and evaluation of risks during sprint retrospectives, and as a tool for communicating identified risks in a more dynamic and concise fashion, compared to risk lists that are produced through the traditional approach to risk identification and analysis from Boehm (1991).

As this thesis will primarily be concerned with perceptions and explanations to risks, it may be viable to clarify what a risk is and which definition this thesis adheres to. A risk may be defined as an unintentional event that has a negative impact on a project, when it occurs (Moran, 2014,

18, ; Sommerville, 2016, 644; Schön, Radtke and Jordan, 2020, 2). A risk should not be misunderstood as simply being uncertainty. The differentiation between a risk and an uncertainty lies in risks being knowable in probabilistic terms and uncertainty is perceived to be unknowable randomness (Moran, 2014, 18). Some authors focus solely on uncertainty instead of risks (Ward and Chapman, 2003), and thereby adopts a broader scope (Dingsøyr and Petit, 2021, 76). Uncertainty is understood as an aspect to take into account when estimating the probability of occurrence of a risk and will therefore not focus on uncertainty as a broader concept than a risk (Pressman and Maxim, 2015, 745). The focus is on project-related risks, which concern threats to the project schedule and its resources (Pressman and Maxim, 2015, 746; Sommerville, 2016, 644). This focus address the risk management literature calling for exploration of different types of risks other than technical and financial risks (Ackermann *et al.*, 2014). As well as differentiating from the literature by focusing on risk causes and explanations to risks, in contrast to focusing solely on risk identification.

2 Theoretical backgrounds

This section presents the theoretical background constituting this thesis, which is divided into two subsections: software risk management and causal mapping.

2.1 Risk management in software projects

Risk management is one of the most decisive and advantageous approaches that agile software teams have at their disposal (Odzaly, Greer and Stewart, 2018; Tavares *et al.*, 2021). Failures in software projects are far from unheard of and one way to accommodate these failures is risk management (Chaouch, Mejri and Ghannouchi, 2019; Tavares *et al.*, 2021). Risk management is the process of anticipating risks and to manage events that may cause unwanted changes or impose a negative impact on a project (Sommerville, 2016; Chaouch, Mejri and Ghannouchi, 2019). Boehm, (1991) considered the father of software risk management (Menezes, Gusmão and Moura, 2019), divides the process of risk management into two core activities. The activities are risk assessment and risk control. Risk assessment is further divided into three activities, risk identification, risk analysis and risk prioritization. Risk control is divided into three sub activities as well, risk-management planning, risk resolution and risk monitoring. Risk assessment is generally concerned with documenting potential risks, assessing their likelihood of occurrence and their potential harm to a team or project. Whereas risk control will mainly involve planning strategies on, how to reduce the likelihood of occurrence and potential harm of the risks, as well as monitoring any changes in the risk profile (Boehm, 1991). These key

principles and activities from Barry Boehm are widely used within the software risk management literature. This can be said of researchers concerned with either agile processes or plan-driven processes (Moran, 2014; Pressman and Maxim, 2015; Sommerville, 2016). This aligns with the fact that Boehm did not present his risk management process with a specific agile method in mind (Boehm, 1991).

The risk management literature is predominantly focused on plan-driven processes as opposed to processes in agile contexts (Moran, 2014, 33-35; Odzaly, Greer and Stewart, 2018,) where the methods do not promote formal techniques or any activities related for managing (Tavares, da Silva and de Souza, 2019; Tavares *et al.*, 2021). In agile teams, risk management is often an informal process with a low amount of documentation. The agile methods are more concerned with reducing risk through achieving transparency, inspection and adaptation (Sommerville, 2016, 647; Schön, Radtke and Jordan, 2020).

Tavares, da Silva, and de Souza (2019 propose a list of risk management practices for agile projects to increase agile team's likelihood of success by systematically analyzing a range of existing agile risk management activities. The purpose of their contribution is to combat the aforementioned lack of risk management activities in the agile software development methods (Tavares, da Silva and de Souza, 2019). Moran (2014) proposes a general agile risk management process that encapsulates elements from traditional plan-driven risk management, whilst remaining faithful to the principles presented in the agile manifesto. Their goal is to provide agile teams with a detailed process of managing risks that enables them to continue despite of the risks they might encounter (Moran, 2014, 33).

Others are preoccupied with improving the risk management process within the scaled agile framework (SAFe). Schön et.al conducted a case study of a large-sized ecommerce company using SAFe and investigated how to improve their risk management process. They reveal that in large companies with many teams working on the same project, it is useful to use a more formal risk management process and it is beneficial to combine agile and traditional development methods (Schön, Radtke and Jordan, 2020). Odzaly et.al propose an agile risk management tool where software agents are utilized to assist agile teams in identifying, assessing, and monitoring risks. They demonstrate that using software agents in terms of risk management is a useful tool to minimize the human effort required to manage risks, which is achieved by allowing the team to react dynamically to changes in the environment and in detecting risks (Odzaly, Greer and Stewart, 2018). Some researchers zoom in on specific properties of risks,

such as the factors to why the risks occur. The reason to focus on risk factors is due to the fact that accurate identification and monitorization of the risk factor may be crucial for the success of software development projects and their overall quality (Menezes, Gusmão and Moura, 2019).

Moreover, Perkusich et.al. (2021) demonstrate a knowledge-based risk management framework for Scrum software development projects, where they focus on both opportunities and threats to the product delivery process. They utilized causal mapping as the basis to relate risks to key process factors, which should ensure a causal analysis of risks. In addition, they propose a range of activities and roles to ensure reuse-driven risk management in an agile software development context (Perkusich *et al.*, 2021, 114). It may be deduced from their contribution that the framework has yet to be tested in a software development project e.g., in a real-life context. The authors contributing to the agile risk management literature focus mainly on improving agile teams and projects' risk management process, proposing activities and frameworks and are generally concerned with the identification and analysis of risks (Menezes, Gusmão and Moura, 2019). The focus will now shift to a method that has recently acquired acquisition, but it still requires more exploration in the field of software risk management (Al-Shehab, Hughes and Winstanley, 2004; Ackermann and Alexander, 2016).

2.3 Causal mapping in the IS literature

As asserted by Laukkanen (1998) causal mapping is a type of cognitive mapping in which the units of analysis clarify their causal claims in terms of specific phenomenon or issue relating to a real-life situation. The proposed method for collecting the relevant data enabling the researcher to elicit the causal claims is interview sessions (Laukkanen, 1994, 1998). As opposed to other researchers interested in causal mapping techniques, who prefer to use interactive workshops as the mean of eliciting causal claims (Ackermann and Eden, 2005; Ackermann *et al.*, 2014). This thesis will primarily rely on interviews as the main method of collecting the appropriate data, but as it will be explained in section 3 the interviews will be supplemented by other data sources. Causal mapping displays the patterns of concepts and causal beliefs that reside in definite statements of different groups. It is intended as an appropriate tool for comparative analysis of different types of actors within an organization and as a means of identifying variations and similarities across the perspectives of the relevant actors (Laukkanen, 1994, 1998).

The process of eliciting causal relationships is concerned with finding expressions that reveal: phenomenon A leads to/causes phenomenon B or B is an outcome of A. These are causal assertions and are visualized through graphs, which consists of nodes and arrows that link the nodes together. The nodes denote concepts and phenomena, which their owners, such as managers may seem to perceive in their situations. The arrows symbolize the actors' beliefs about causal relationships among the concepts. These linked concepts can portray the patterns of causal thinking of a person or across a group (Laukkanen, 1994). The causal mapping technique has been selected as it provides a versatile platform for explanatory cognitive studies and presents the researcher with a more practical and powerful tool to analyze data and phenomena, than e.g., a text-based analysis (Laukkanen, 1998; Laukkanen and Eriksson, 2013).

Furthermore, Williams et.al. (2013) conducted a case study with the UK Ministry of defense as the case and client, where the client had put together a project risk register, and the project managers wanted to know how to get the most out of the register. Williams et.al. (2013) decided to convert the register into a causal map to illustrate the relationships between risk items and to gain a deeper understanding of the causality (Williams, Ackermann and Eden, 2013, 343-345). They conclude that the structure of the final causal map could be used with a scenario approach as a useful tool to analysis and planning of possible actions to be tested in relation to possible alternative future scenarios. The causal maps may also be a powerful tool in terms of risk evaluation. In addition, they point out that the causal maps are a highly beneficial tool for managers in relation to risk management. This is due to the fact that they assist in initiating new thoughts on risks and their relationships, enables examination of the interaction between risks, suggests potential danger zones, increases clarity of contradictions and inconsistencies and helps to identify actions that may address more than one risk (Williams, Ackermann and Eden, 2013, 346). Al-Shehab, Hughes and Winstanley (2004) suggest using causal mapping as a post-evaluation framework for software teams, as well. A single causal map can capture the knowledge that a software team has gained during a past-project. It may be used to evaluate lessons learned and facilitate learning within an organization. The issue of using causal mapping as a post-evaluation method is to move from the diagnosis of the sources of previous problems and to become able to predict potential problems in new future projects (Al-Shehab, Hughes and Winstanley, 2004).

To accommodate the utilization of causal mapping in explaining diverse causes to project risks across different software team roles, the proposed steps were adopted: 1) predate and data collection stages (participant selection, interview design, symmetric data collection and follow-up

interview sessions, and 2) analysis and post-data collection stage (establishing coding guidelines and categories, identifying causal relationships, modelling maps based on selected themes, analysis of maps and seeking feedback from experts) (Laukkanen, 1994; Laukkanen and Eriksson, 2013; Ghobadi and Mathiassen, 2014).

3 Research approach

In the following section, the research approach, and the applied methods of collecting data, as well as the process of data analysis will be presented. The specific case and its context will also be introduced in greater detail. Finally, the findings conducted on account of the data collection and data analysis will be presented. This thesis is conducted using an action case study as the overall research approach. The action case study is a hybrid approach consisting of intervention from action research and interpretation from the case study approach (Vidgen and Braa, 1997, 526-528; Braa and Vidgen, 1999, 30-32; Svangren *et al.*, 2021, 4).

The case study approach is characterized by being an empirical enquiry that uses different methods of collecting data to examine one or more instances of a phenomenon within its real-life context (Yin, 2003, 12-13; Host *et al.*, 2012, 12). Action research is concerned with improving or changing a "real-life" problem situation through collaboration with concerned stakeholders in the problem context and to test hypotheses through intervention in the organization (Braa and Vidgen, 1999, 30; McKay and Marshall, 2001, 50). On the one hand, researchers performing case studies contribute to change by questioning incidents and by applying new concepts. On the other hand, a strict use of action research is often inappropriate on the account of organizational constraints or the nature of the subject to be examined. A less strict use of intervention combined with a thorough contextual understanding is one viable way of balancing the previous stated dilemma. This combination constitutes the action case. The utilization of both action research and case study research is conducted at the expense of descriptive capabilities (Vidgen and Braa, 1997, 530; Braa and Vidgen, 1999, 34).

This thesis makes use of the interpretative and explanatory capabilities from case study research to understand and explain the different role owners' perceptions of risks and the causal relationships between them. Moreover, this thesis is also utilizing properties from action research, through the small-scale intervention that occur, when the results are presented to the collaborators and due to the collaborators' participation in a workshop held by the researcher. The researcher is also intervening in the case context by applying new concepts and by asking inquisitive questions to the different role owners in Alpha. The combined aspects of case study

research and action research are therefore appropriate when it comes to the overall research question. This is due to the explanatory and interpretive nature of the research question, where intervention is used to ensure a high degree of validity and triangulation. The conceptualization of combining action research and case study research may be viewed in a more specific and visual sense in figure 1.

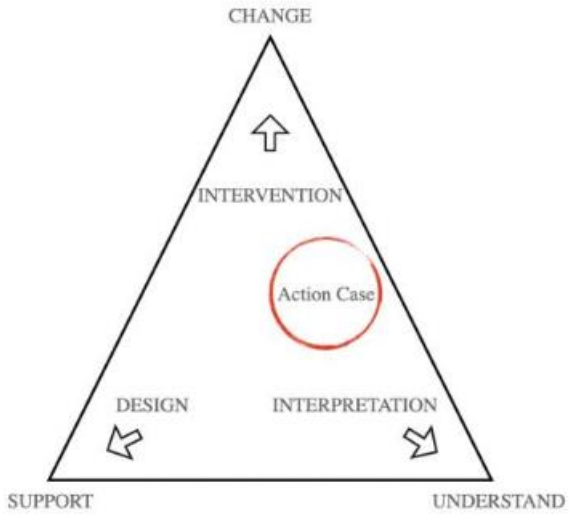

Figure 1: Action case as a hybrid method between action research and case study research, derived from (Braa and Vidgen, 1999) and (Svangren et.al, 2021)

The researcher is likewise aware of the fact that even though action research and case study research are intertwined, an action case might still fall victim to the criticism and weaknesses of both research approaches (Vidgen and Braa, 1997, 534). An action case may suffer from difficulty in maintaining control from the case study approach (Host *et al.*, 2012, 20) and overlooking ethical implications of change from action research (Vidgen and Braa, 1997, 534). As a means of accommodating with the difficulty in maintaining control, a case study protocol was conducted. The case study protocol is recommended by Yin (2018), on account of its ability to enhance the reliability and replicability of the case study and acts as a valuable tool to guide the data collection process (Yin, 2018, 94-99). The case study protocol is included in appendix 1.

Furthermore, a case can be a project, individual or a group of individuals and can be categorized as both a single-case or multiple case study. The single-case focuses on one case within one context, whereas a multiple-case focuses on multiple cases and contexts (Yin, 2003, 14-15; Host *et al.*, 2012, 26-27). This action case will be conducted as a single case, where the previously mentioned software development team is the case and the different roles of Alpha constitute the units of analysis. The context will be their latest project and the *Estate bank* they

reside in. According to Flyvbjerg (2006) the single-case study has been criticized for not providing sufficient evidence for generalization, as opposed to multiple-case studies. Flyvbjerg stresses that, even though a single-case study might not always be able to generalize phenomena across a specific context, it is still valuable in its own right. A single-case study can contribute to the scientific body of knowledge, without providing evidence of generalization and mentions that generalization is overvalued as a source of scientific development, whereas the weight of a single-case is underestimated. In addition, a single-case study can both test theories, methods and hypotheses, as well as generate new ones (Flyvbjerg, 2006, 224-229).

3.1 The case

The specific case applied to examine the overall research question is a software development team that has its base of operation at a large Danish bank in Aalborg. The *Estate bank* has around 4000 employees in total and offers a wide range of products and services in relation to banking transactions and housing investments. Their main activities are mortgage credit and banking. The *Estate bank* is owned by an association of homeowners and corporations, which are also located in Denmark.

The researcher has been in contact with a representative from Alpha and demonstrated an interest in Alpha's risk management processes. The representative possesses the role of UX designer in the team and acted as the personal contact for the researcher. The representative and the researcher had held a meeting about their general risk management practices and whether they had any issues related to risk management. The representative mentioned that they have issues with learning from previously occurred risks. The knowledge gathered from this meeting laid the foundation for the overall research question of this case study. The researcher is therefore interested in investigating whether causal mapping can explain the causes of risks and if Alpha can learn from the explanation.

The software development team consists of 12 team members in total and consists of one Product Owner (PO), one Scrum Master (SM), one Business Analyst (BA), two Architects (ARC), four Front-end developers (FD), one Back-end developer (BD), two UX designers (UX) and one Student Assistant. The team members in Alpha have worked together for a long time and know each other well. The team members in Alpha form a closely connected team. Four out of the thirteen team members are located in Poland, whereas the rest of the team operates in Denmark. The PO's primary objective is to prioritize tasks in the backlog and to ensure that Alpha deliver the most important tasks on time to their users. The SM is responsible for ensuring a

proper use of Scrum in Alpha by following the Scrum guide, facilitating Scrum events, and shielding the team from impediments to maintain the focus of the team. The BA act as the link between the IT-department and the rest of the corporation. The BA is responsible for defining the corporation's requirements and to make sure that the technical aspect of the software solution lives up to the required quality. The UX takes part in analyzing the team's work cycle and the software requirements, as well as defining requirements and preparing tasks for the developers supported by design examples. The FD develops the front-end related tasks and is also responsible for testing the software, whereas the BD is concerned with developing and maintaining API's. Finally, the ARC participates in developing both front-end and back-end related tasks and have a broader responsibility of the development process than that of the developers. Throughout this thesis the two developer roles will be abstracted into one role and will simply be referred to as developers.

Alpha makes use of an agile development method, called the Scaled Agile Framework (SAFE). Within the SAFe development method, a cluster of teams are working towards shared goals and solutions. This is done through the agile release train (ART), where the goal is to deliver a continuous flow of value. The process starts off with a fixed and reliable schedule, which is determined by the chosen rhythm of the program increment (PI). Within each program increment, teams embark on a new system increment every two weeks and all the teams are embedded in the same PI length, which lasts 10-12 weeks. The PI's have common start and end dates and duration. The different teams working on a PI must conduct the most essential event in the SAFe framework, which is the PI-planning event. The agenda of the PI-planning event is to present the business context and vision, what is the most important to focus on and develop in the future.

3.2 Data collection

In this section the different methods for data collection and the rationale behind the selection of the methods will be presented, as well as the actual process of collecting data through the use of the following methods.

The researcher interviewed 10 members from Alpha in total. The researcher took field notes during the interviews, focusing on explaining diverse risk causes. The specific members include: one PO, one SM, one BA, two UX's, two Arc's, two FD's and finally one BD. The team consists of 14 members in total, two of the remaining team members that were not interviewed possessed the role of front-end developer, and the last one occupied the role of student assistant.

The researcher decided not to interview the two remaining front-end developers as the researcher estimated that the likelihood of acquiring new concepts and causal relationships from them was highly unlikely and therefore not worth pursuing. The student assistant was excluded as a candidate because the student assistant would most likely not be able to know much about the topic and was not considered a reliable source to inquire insights from in relation to Alpha's perceptions of risks.

The interviews lasted between 40-65 minutes, depending on how much time the interviewees had available, how much the interviewees had to say during the interview and the researcher's occasional inability to keep track of the time during the interviews. 9 of the interviews were recorded and saved to enable the researcher to backtrack if the need should arise and to enable the researcher to not only rely on the field notes, thus minimizing the degree of uncertainty during data analysis (Patton, 2015, 472). One of the interviewees wished to not have the meeting recorded. 5 of the interviews were conducted online through Teams, due to the PO resided in Copenhagen and two of the developers resided in Poland, whereas the last two interviews were conducted online as it suited the interviewees and the schedule of the researcher better. The other half of the interviews were conducted face-to-face at the location of the *Estate bank*. The researcher is aware of the impediment, when it comes to conducting online interviews. During a face-to-face interview the researcher is able to perceive the non-verbal communication from the interviewee, where during an online interview this ability is lost (Patton, 2015, 494). To combat this impediment the online interviews were held with cameras on, on both ends, although one interviewee preferred not to use the camera during the interview.

The interviews are semi-structured, which entails that the questions are prepared beforehand, but are not necessarily asked in the same order as they are listed. The progression of the conversation throughout the interview can have an impact on the order in which the different questions are asked to the interviewee (Host *et al.*, 2012, 50; Benyon, 2014, 143). Semi-structured interviews enable improvisation and exploration of the emerged issues during the conversation. The semi-structured entails open and closed questions, which allows for reducing both the time and complexity of the analysis through including some closed questions and allows for a broader exploration with some open questions (Host *et al.*, 2012, 51). Moreover, an interview guide was formulated to make sure that the same questions are asked throughout all of the interviews and is also used to ensure that all the relevant questions have been asked. The interview guide aids the researcher in deciding the best use of the limited amount of time available and making the interview process more systematic and comprehensive (Patton, 2015, 439). The

final interview guide was generated through a couple of iterations and may be inspected in the case study protocol in appendix 1.1.

In addition to the utilization of interviews, the researcher also observed one of Alpha's retrospective meetings. The meeting took 1,5 hours and the researcher took field notes regarding explaining risk causes, the mood and environment and designed a causal map of mentioned risks in real-time. The researcher used the data gathered from the retrospective meeting to gain further insights into the residing risks and risk causes in Alpha and to equip the researcher with a sensation of how their retrospective meeting is conducted and structured. This insight was valuable as the researcher wanted to present the results from the analyzed data during an extended retrospective meeting. The researcher was also able to attend Alpha's initial and concluding PI-planning meeting during the second day of the PI-planning event. The researcher attended the initial PI-planning meeting on the second day since this was where Alpha identified and discussed risks in relation to the team and the overall program. Both the initial and concluding meeting lasted around 30 minutes.

These observations act as a supplement to the interviews to achieve a higher level of data triangulation. Triangulation implies using multiple perspectives towards the studied object and thereby obtaining a broader picture. Data triangulation means to use more than one data source or to collect the same data at different locations. Triangulation is essential to enhance the precision and to increase the validity of empirical research (Host *et al.*, 2012, 15-16). The observational method allows the researcher with the opportunity to see aspects that may be subconsciously or unconsciously ignored by the people in the setting. Another valuable aspect of conducting observations is the possibility of learning things that respondents would be reluctant to talk about during interviews (Patton, 2015, 333). During the observation, the researcher acted as an observing participant, where the level of interaction by the researcher is low and the level of awareness of being observed among the participants is high (Host *et al.*, 2012,).

This thesis also makes use of archival data, which will further increase the degree of data triangulation and the ability to obtain an even deeper understanding of the relevant situation and phenomena. The archival data used are documents containing documentation of issues stemming from Alpha's 10 latest retrospective meetings and the identified team and program risks from the three latest PI-planning events. Archival data is a third-degree data source, which has some limitations, such as the fact that the data was not constructed with the purpose of providing data for research in a case study. A document may include much information of lower value

to the researcher and may exclude information relevant to a case study for political or confidentiality purposes. As with all third-degree data sources, they should be used in combination with other data collection methods (Host *et al.*, 2012, 57-58). Especially, because archival data may provide a limited amount of relevant data, by being incomplete or inaccurate.

Combining these different data sources will not only enhance the level of triangulation, but will also allow cross-check between findings and they may complement each other in terms of their strengths and weaknesses (Patton, 2015, 389). Observation is limited in the number of activities observed and the fact that the researcher cannot know what everyone are thinking during the observation process. Whereas interviews are limited to recall-error, reactions of the interviewee in relation to the interviewer and self-serving answers. Observation may provide an opportunity to verify what was said during interviews and interviews complement observation in allowing the researcher to go beyond the capabilities of observation by exploring feelings and behavior (Patton, 2015, 389).

An overview of the data collection methods and different items of collected data may be viewed in table 1 in appendix 2.1. After having presented the chosen data sources and how the data was collected, it therefore seems natural to explain how the collected data was analyzed next.

3.3 Data analysis

The researcher initiated the process of analyzing the collected data by reviewing the field notes and by reviewing the recordings generated from the interviews. The researcher decided to transcribe only the parts of the interviews revolving concrete mentions of risks. This was to accelerate the process of data analysis slightly, whilst ensuring the benefits of transcription. The most noteworthy benefits of transcription in this instance were a complete overview of all the mentioned risks and risk causes, as well as a more structured analysis process. 7/10 interviews were transcribed with this focus on risks, this was due to one of the interviewees preferred not to have the interview recorded. The other two interviews were not transcribed, as the answers from the interviewees were very brief and modest, therefore a review of the recording and the field notes sufficed.

Thereafter, the transcripts and the field notes were analyzed to uncover the different role owners' perception of risks and risk causes. The researcher coded statements referring to risks and strived to find patterns among the role owners themselves and between the different types of role owners. During the coding process the researcher focused on the mentioned risks, the number of times it was mentioned and who it was mentioned by. The identified risks were also

divided into different coding categories to allow the researcher to view the risks in a more nuanced fashion. The coding categories include; actor, structure, task and technology, denoting which aspect a specific risk is related to. Actor constitutes individuals or groups of stakeholders, who may influence the software development. Structure denotes ways of communicating, hierarchical structure and systems behind an organization's work processes. Task means the development tasks, which prescribe how and what the developers should achieve. Technology represents the technological methods, tools and infrastructure required to develop the software system (Lyytinen, Mathiassen and Ropponen, 1998, 236). The data from the coding process has been grouped into a table to gain an overview and may be viewed in appendix 2.1. This categorization of risks and their connectedness was derived from Leavitt, (1946) which also presents a model for categorizing risks. Additionally, this model has been adopted and a model was constructed for each role, where each category in the model has a value denoting the number of times a risk from a specific category causes the creation of a risk belonging to another category. These models may also be observed in appendix 2.2.

A concern mentioned by the interviewees was deemed as a risk when it corresponded to the previously mentioned definition of a risk. A causal relationship between two or more risk was identified, when the interviewee mentioned a risk as a contributor to the occurrence of another risk. The researcher first constructed a generic causal map for each role in Alpha, whilst focusing on the processes of the different risks, where the risk causes are showcased in the bottom of the map, the risks in the middle and the consequences of the risks in the top. This process-focused view on risks in a causal map was specifically inspired from (Ackermann and Alexander, 2016). After having mapped the different processes of each risk and their causes and effects, the researcher constructed a more concise causal map for each role. This time the causal map contained only the risks mentioned by the different role owners and the causal relationships between them. The process of constructing the concise causal maps was relatively manageable, due to the overview and experience gained from the initial process-focused causal maps.

The specific techniques to visualize the conducted causal maps are adopted from (Laukkanen, 1994; Ackermann and Alexander, 2016). The techniques for visualization were selected in favor of those from Ghobadi and Mathiassen (2014), because they were perceived as being simpler to conduct by the researcher and were estimated to be easier to comprehend by the stakeholders. In addition, the issues of comprehending the visual expression of the causal maps from Ghobadi and Mathiassen (2014) by stakeholders are discussed in (Ottosen, 2019). It is revealed

that the respondents had a difficult time comprehending the conducted causal maps. This was due to its complexity and the amount of variables that needs to be explained (Ottosen, 2019, 26-28). The application of the visual expression through the causal mapping technique from Laukkanen (1994) and Ackermann and Eden (2005) in favor of that of Ghobadi and Mathiassen was also, because of the expression's more concrete nature. Ghobadi and Mathiassens' causal maps are focused on generic areas of barriers to knowledge sharing, whereas the ones used in this thesis will be focused on concrete events. The researcher argues that concrete concepts and events are not only more comprehensible to understand, but also more valuable, when it comes to explaining risk causes to a software team. This is due to the fact that the causal maps will showcase specific risks and risk causes, instead of e.g., generic and abstract risk areas. The above adds to the rationale behind choosing the visualization of causal maps from Laukkanen (1994) and Ackermann and Eden (2005).

After having constructed the causal maps, they were showcased and explained individually to six representatives from each role in Alpha with the goal of obtaining feedback and to accommodate uncertainty in terms of each of the constructed causal maps. The researcher chose to only showcase the risk-focused causal maps to the different representatives, as the researcher estimated that this type of causal map would be the simplest to understand and the most valuable to Alpha, due to its concrete nature. The adoption of individual feedback sessions was adopted from Ghobadi and Mathiassen (2014).

The individual feedback sessions were followed by a workshop with Alpha from *Estate bank*, where the causal maps were presented to all of the roles from Alpha. The causes to the risk "inability to comply with the deadline" among the roles, were also presented during the presentation. These causes were selected as the primary focus of the presentation, due to the risk was interpreted as one of the most central risks to Alpha by the researcher. This is since the risk is mentioned by 5/6 of the roles, is the risk with the most causes leading to it and is therefore also the one with the most noteworthy patterns among the roles. The risk and its explanations are likewise interpreted as the most relevant in an academic sense on account of the same rationale as before, and will act as a fitting baseline, throughout the following section. Finally, the workshop provided primarily feedback on the applicability and usefulness of the causal maps. This feedback will be presented and discussed in greater detail by the end of the following section in 3.4.7.

3.4 Findings

To examine the overall research question and accommodate the need expressed by the representative from Alpha to better learn from occurred risks. The researcher set out to use the causal mapping method. The aim was to explain the risks residing in Alpha from the different roles' perceptions of risk causes. However, the first interviews with representatives from each role in Alpha revealed few explanations of risk causes, which resulted in difficulties for the researcher in constructing the causal maps for each role. The researcher's realized that the mentioned risks from the interviews needed a lot of investigation to grasp the interconnectedness between risks. The identified risks (see table 2 appendix 2.2) did not necessarily connect in a meaningful way expressed by the roles in the software team.

The difficulties in constructing a comprehensible and plausible causal map for each role induced substantial self-doubt in the researcher. Thus, the first causal maps were generic, large, and complex maps, of all the concepts mentioned by the roles. Many of the connections between the concepts in the causal maps were made solely on account of the researcher's intuition. The interviewees did not always talk about risks in a way that made it straightforward to establish a causal relationship between the concepts, which further fueled the researcher's uncertainty about the causal maps' plausibility. These maps were also way too complex for anybody to comprehend. It became clear that the level of abstraction needed to increase, and the degree of uncertainty needed to decrease (see appendix 3). In the middle of conducting the interviews, the researcher attended a sprint retrospective meeting with Alpha, to increase the data triangulation, but the uncertainty and low level of abstraction were still persistent, as Alpha did not explain risk causes throughout the meeting.

The researcher sought to overcome the issue, of increasing abstraction in all of the causal maps by focusing solely on mentioned concepts that corresponded with the definition of a risk. In terms of maps with many risks, the researcher either abstracted them into one or even excluded some from the map. The researcher was predominantly satisfied with the level of abstraction, but the level of uncertainty, especially regarding plausibility, continued to concern the researcher. Throughout the data collection process, the use of a case study protocol seemed only to have resulted in a temporary feeling of certainty at the beginning of the data collection. The researcher arranged a feedback session with a representative from each role to overcome the uncertainty. Each role recommended a few adjustments regarding the relevance of the identified risks in the causal map and the plausibility of the causal links between risks. The feedback from the different representatives resulted in an updated version of the initial causal map and,

more importantly, decreased the researcher's uncertainty about the causal maps' plausibility. With a greater confidence in the causal maps, the researcher scheduled a workshop with the software team to present the causal maps for each role, receive collective feedback, and induce mutual learning for Alpha and the researcher. In the presentation, a general narrative of the problem situation arose about Alpha's main concern with meeting their deadlines and avoiding getting delayed. This risk was observed as being the most frequently mentioned risk by the interviewees. All the identified risks from the different roles were contracted into a table, showcasing who mentioned the risk and the assigned coding category, which may be viewed in table 2 in appendix 2.2. A few roles ascribe inefficient communication and the inadequacy of their technological tools as critical causes for potentially not delivering their tasks on time. This narrative will function as the overall frame and will continue to be explained in greater depth, throughout the rest of this section with each role acting as a piece of the puzzle.

3.4.1 Developer

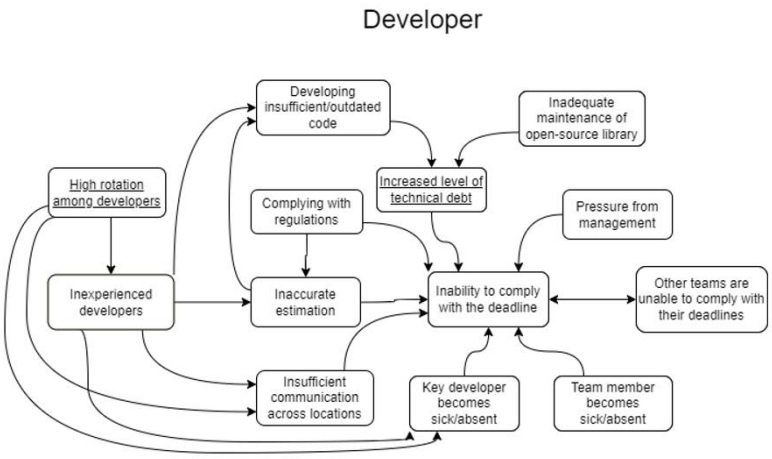

Figure 2: Developer's causal map

The developer explained that a possible reason to being unable to comply with their deadlines was, because of a key developer becoming absent from work. The key developer received this particular role, due to being the only Danish developer and only one able to understand the Danish documentation, they are currently working with. As well, as being the developer with the most experience with working at *Estate bank.* The developer explains, they have previously experienced the key developer suddenly getting sick. This could have resulted in: "*It could have had the outcome, where we had to say, we cannot deliver this task. We have to move the process or our deadline".* The fact that the key developer was not able to translate the documentation for the Polish developers and make use of their experience, could have resulted in delaying their deliveries. This is the rationale behind the arrow connecting the risk "key developer becomes sick" and "inability to comply with the deadline". Although, in this example the key developer was able to finish the tasks that could have been delayed, because the key developer was only sick for five days. The developer mentioned that the risk of missing a deadline, due to the key

developer becoming sick continues to persist, because management has yet to hire more Danish developers. The lack of Danish developers and the management's hesitation to hire them in the team, was perceived to be the result of favoring cheaper and contract-based developers from Poland. The developer explains: "*Management wants Polish developers. I believe it may be, because they are cheaper, but also because they are contract-based, so they are also easier to get rid of again".* It may be likely that this issue continues to exist indefinitely in the future, for as long as management favors cheap manpower.

Furthermore, a developer explained that not only their team, but also other teams might be delayed, if a regular team member becomes sick in their team: "*If the task has not been done, it might be, due to someone getting sick or their kids get sick and must take time off".* This means that any team member becoming sick, may have a consequence for the timely delivery of their tasks. The developer clarified that Alpha has been delayed in the past, because of other teams working on the same program within the SAFe method: "*We are responsible for the front-end. If our back-end teams cannot deliver the tasks we need, then we have to postpone our tasks or code the GUI anyway".* The quote exposes a reasonable critique of the SAFe method that having many teams working on the same program, an issue in one team can easily present consequences for other teams, and the deadline for the entire program increment might exceed. The high dependency on other teams as a cause leading to missing deadlines, was mainly perceived as being related to the structural composition of the IT-department at *Estate bank.*

The same can be said of having to comply with regulations, insufficient communication, and pressure from management. A developer stated management may put pressure on the team: "*Management might pressure us to do the delivery faster, even though they know it is probably not possible*". And:" *Once a data is set you have to meet it, because other teams are dependent on that… the date becomes real at some point*". These statements do not present a direct explanation or link to missing their deadlines during the development of their tasks. Although they do reveal that one explanation to missing their deadlines may be, due to management's initial unrealistic planning of deadlines for their deliveries. The risk of missing deadlines may be inscribed at the beginning of every program increment because management chooses to accept a presumptuous deadline to begin with.

The two other structure related risks presented greater difficulty, when assessing their implications and coherence with other identified risks. Regulations were only mentioned as having an

impact on, what they are working on and how long a production will take. The researcher deduced that it may contribute to some degree, to Alpha missing a deadline and the accuracy of an estimation of a task. The rationale was that it may take up unexpected time from some of the team members. The estimation of tasks could on a rare occasion be estimated inaccurately, which: "*It could have the consequence of having spill-overs, where the tasks surpass to the next sprint*." Inaccurate estimations present a contributing factor in explaining delayed tasks, whereas the insufficient communication across locations' causal relationship with the risk is vaguer in its nature. The Danish developer stated a difference in work schedule and culture, which may present difficulties in communicating across locations: "*If I for example have a task, I find difficult, then I would much rather go to the team members sitting in this office".* This reveals a hesitation of communication or asking for help from a team member across locations but does not reveal any causal relationship with meeting a deadline. An obvious critique of the developer's causal map must be the vague nature behind the connection between those two risks.

The developer mentioned technical debt as an issue at the end of an interview but did not go into greater details or presented any explanations meaningful for the researcher at the time. To accommodate with the plausibility of the explanations between some of the risks and the validity of including technical debt as a risk in the causal map, the researcher sought to receive individual feedback from the developer. The developer agreed to the rationale behind the causal relationships that complying with regulations presents but did not introduce an actual explanation. Although, the developer accepted the causal map overall, the ambiguity of the risk and its causal relationships persisted after the feedback session. The researcher did not receive a possible explanation to how insufficient communication may explain missing a deadline. The researcher did receive confirmation of including technical debt as an explanation to missing a deadline:

" *Technical debt has a significant impact on meeting our deadlines, because if you gain a lot of technical debt, it will take longer to finish a task and then it is not likely you will meet your deadline."*

The developer continued to highlight the significance of the risk of technical debt, by selecting this risk as one of the most essential risks in the causal maps. This risk is highlighted in figure 2 with an underscore. Technical debt was explained to be caused by technological related risks, for instance: "*We currently have some outdated code and I know there are tasks in the next*

PFP (meeting), which is affected by this code and I know, due to this technical debt from this outdated code that the task will take longer to finish in the future, if we do not fix it". And: "*One of the examples of technical debt is actually having these open-source libraries or other dependencies that does not get maintained".* This emphasizes the fact that outdated or otherwise insufficient code and unexpected issues with open-source libraries may lead to the accumulation of technical debt, which inherently impacts the likelihood of meeting their deadlines.

Lastly, the explanation behind the risk of the key developer becoming sick had changed since the interviews were conducted. The team has received another Danish developer, but the developer continued to be perceived as an indispensable part of the team. This was no longer, due to being the only developer able to translate documentation in Danish, but because the risk of high rotation has increased significantly, as well, since the interviews. This meant that the key developer was the one in charge of training new developers: "*A lot of time is spent on it and it means that sometimes there is not enough time to finish your own tasks, because you spent all your time on training someone".* The risk related to the key developer, was not only an issue if the key developer becomes sick, but also currently during their daily operations. The increase of rotation among developers has caused an increase in the amount of new and inexperienced developers in the team to accommodate with resent resignations. This places a pressure on the key developer to both train new developers and finish their own tasks, without letting it influence their ability to comply with their deadlines. The high rotation was explained as being, due to: "*they(management) don't want rotation, but they cannot offer enough money as a raise and do not offer a raise, so employees leave and a new one comes in*". The developer reveals that the rotation within their team was, caused by management's inability to offer a more appropriate salary and the prospects of a higher paying job. The risk of high rotation among developers was explained as being one of the most important risks by two of the developers. The importance of the risk related to the key developer and the high rotation in the team was supported by the fact that, both risks were mentioned during their recent PFP meeting (see table 3-4 in appendix 2.2).

3.4.2 Architect

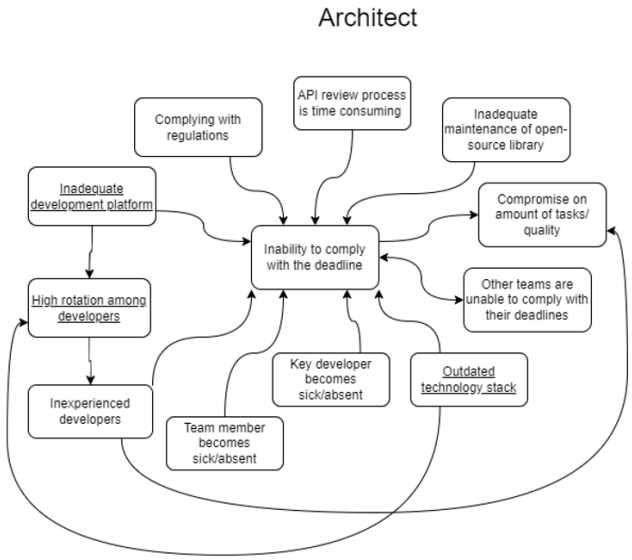

Figure 3: Architect's causal map

There was a consensus between the developer and the architect of perceiving the key developer becoming sick as a potential risk, which may be caused by the lack of Danish developers to translate documentation. The architect refers to the same previous situation as the developer, where the key developer was sick. Although, the role was not as detailed in their explanation of the consequences of the risk as the developer was: "*There may be a challenge, especially when we have a Danish developer, an actual developer, then there may be a challenge, if sickness should occur and you have a critical task*". The risk of the key developer becoming sick from work, was still interpreted as being a possible explanation to missing a deadline. This was, because of the similarity in the description of the past event as the developer, and because the challenge that sickness of the key developer may present for a critical task, was interpreted as being a delay in its delivery. This explains the causal relationship between the two risks, as observed in figure 3. In continuation of sickness, the architect mentioned the risk of other team members becoming sick but did not explain its implications. The researcher reasoned that if the key developer may have implications for the timely delivery of their tasks, then other team members' absence may present a similar cause, if the specific task was sufficiently critical and the team member was absent for a long time.

The architect did not mention the risk of inexperienced developers, as the developer did. Although, the role did share the same concern of rotation among developers. The architect specified that the rotation was mainly an issue for the Polish developers, they leave, due to: "*It is very much, due to the salary, when people are offered 40% more in their salary, it is hard to compete with".* On account of this quote and the fact that this concern has persisted for a long period of time: "*It has existed for almost as long as I have worked at Estate Bank*". This aligns with the fact that that was mentioned at their PFP meeting approximately six months ago. It has become apparent that management has been hesitant to raise salaries for inexperienced developers that have worked at Estate Bank for a bit of time. This leads to the developers

leaving the company and: "*There is a way too high rotation among the developers in Poland and we only get to train them, before they are gone*". They explain that there was a risk of Polish developers leaving before they can fully contribute to the team, so Alpha may have a constant flow of inexperienced developers from Poland coming and going. This emphasizes the inclusion of the risk of inexperienced developers in figure 4 and its causal relationship with high rotation, but the connection with inability to comply with deadlines lacks an explanation.

The architect was also worried about inadequate maintenance of open-source libraries similar to the developer, but identifying causal relationships related to this risk was difficult. This was because the architect did not present an explanation that made it possible to deduce a plausible connection to any of the other identified risks. The architect did not explain its implications for increasing technical debt, as the developer did. The architect did explain that inadequate maintenance of libraries may cause problems for their project: "*It took us a long time to figure out what we should do about it. Should we choose another one and then migrate all of our applications over to this or should we take the lead and help this library on the way?"* The researcher argued that it may contribute to causing the risk of not meeting their deadline if they suddenly must spend additional resources on resolving the issue of inadequate maintenance of libraries. The connection between the two risks may be viewed in figure 3.

Unlike the developer, the architect explains the implications of complying with regulations: "*Then you just have to react immediately. They are so hard to see when they come. You have to take care of it, when it presents itself. It can push to stuff, it really can".* The sudden allocation of manpower to handle regulations imposed by the Danish government, may result in rescheduling other tasks currently in their backlog. Complying with regulations was explained to have a higher priority than many other tasks, which may ultimately lead to a failure to comply with the deadline and getting delayed with the delivery of certain tasks. This explains the connection between the two risks in figure 3. The only risk that the architect explained as having a direct impact on their ability to comply with deadlines, was whether other teams were delayed with tasks that they were dependent on. The architect explained that they are a front-end team dependent on back-end from other teams and: *"Yes, sometimes we have to redefine a task, it cannot be done, because we are missing a delivery from another team".* There was a consensus among the developer and architect that other teams' delays have an impact on Alpha's ability to comply with their own deadlines.

During the individual feedback, the architect highlighted the importance of two technological risks that were not mentioned during the interviews. Inadequate maintenance of development platform and outdated technology stack. This was due to: "*I want the tools that makes me effective. Otherwise, I'll get very irritated. It just makes me happy, having proper tools that just work".* The architect continued to clarify that they may have implications for the high rotation: "*Some of these things may be interrelated fx. this technology stack may constitute a higher rotation among developers because they are working with something they really do not want to".* This explains the connection with the risk of high rotation and adds another layer behind the explanation to the high rotation among developers. The risk may also be caused by technological issues, besides the insufficient salary. This may be observed in figure 3. This focus on technological explanations to risks from both the architect and the developer, may be ascribed to their particular role. The developer was responsible for development of tasks, reviewing and testing others code, as well as selecting tasks from the backlog. The architect may also contribute to the development of code and testing code, but in contrast to the developer, the architect was also responsible for more general decision making in terms of development. In comparison with the other roles, the developer and the architect were the only ones, who mentioned any technological risks, which may be ascribed to the responsibility of their particular role. The researcher made the connection between the two risks with the inability to comply with the deadline, even though the architect did not offer an explanation. The connection was accepted tacitly by the architect, during the feedback session. Furthermore, the architect was one of the only roles to point out that the prospect of missing a deadline, may cause Alpha to compromise on the quality of their tasks. The architect explains:

"*If you do not have the most experienced developers, it may have an impact on the quality they deliver and while that is not very good, then one way or another you will have to spend more time on it. It can also have an impact on what they can deliver, the features they can arrive at get more limited".*

This supports the rationale behind the connection between the risk of inexperienced developers, the inability to comply with deadlines and the following risk of compromising on quality and number of tasks. The inexperienced developers usually spend more time on completing their tasks than the architects and the other experienced developers. The inexperienced developers may then contribute to missing some deadlines, whereas they may suffer the consequence of having to compromise on the quality and number of tasks that the team can deliver.

Overall, the developer focused mainly on technological risks compared to the other roles and the explanations to these risks were more elaborate than that of the identified actor related and structural risks. The causal relationships were vaguer and presented greater problems, when interpreting their implications for the other identified risks. The architect was also not as explicit in their explanations as to why they may be delayed, as the developer was. Which in turn caused more uncertainty, when it came to the validity of the causal map and the plausibility of the causal relationships. They were validated throughout the feedback session with the architect, but the architect chose to focus on explaining the technological risks and their implications, over the other identified risks. This may be ascribed to the responsibility and focus of the role of the architect, which lies on that of development and has a greater responsibility on the decision-making process in terms of development.

3.4.3 UX designer

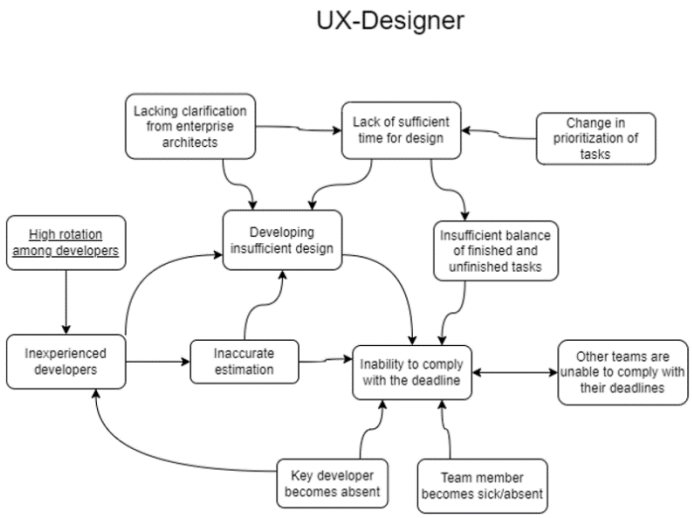

Figure 4: UX-designer's causal map

The UX designer shares the same reasoning behind their rotation among developers as being, due to the unattractive salary for new employees. The UX-designer explains in similar fashion, as the architect and developer that new employees leave the company, when they have become more valuable on the market. This explanation from the three roles indicates a pattern within Alpha. The team continues to lose new developers rather quickly. They then frequently have to spent resources on training new developers, due to the company' inability to retain new developers. This highlights the causal relationship between high rotation and inexperienced developers because this high rotation causes a constant flow of new and inexperienced developers. When asked about why they have inexperienced developers, the UX-designer explained: "*Well, it is purely rotation I think, especially because we have experienced the competition for qualified IT-staff in Poland means there is a high rotation".* When developers leave frequently, they need to hire new and more inexperienced developers to fill in the gap. This reveals that the risk of high rotation was viewed as a significant issue aside from the developers and architects.

The UX-designer attributes the cause of getting delayed and not meeting their deadlines to that of other teams. Similar to the developer and the architect, the UX-designer explains: "*We are the last team in a row of three or four teams that work together. As soon as something starts to get delayed in the early part of the process, we get hit by the sum of all these delays".* It was mentioned earlier that Alpha was a front-end development team, which explains why they were the last team in the process. They get delayed when the back-end teams get delayed. The UX-designer was also aware of the issue of having a key developer in their team, who was the only one able to read Danish documentation. The UX-designer did not offer a direct explanation to the consequences of having a key developer, but did explain: "*Right now, if one of our Danish front-end developers get sick, then we get hit even harder than if it was a Danish developer*". The fact that the UX-designer stated that they get hit hard, if the key developer gets sick reveals that they have a significant responsibility to translate the documentation. This may lead the team to miss their deadlines and postpone certain tasks dependent on the Danish documentation, if the key developer becomes sick for long enough. This clarifies the causal relationship between the risk of the key developer becoming sick and the risk of missing their deadlines, which may be observed in figure 4.

Similar to the developer, the UX-designer perceives inaccurate estimation as an explanation to their inability to meet their deadlines. The UX-designer explained the causal relationship after being asked about causes to delays: "*In reality there are many, but the typical that we meet can for instance be that someone has estimated a task wrongly. It has been estimated that something is easy, and it turns out to be difficult".* As opposed to the developer, who viewed inaccurate estimations as something of lesser importance, due to how infrequent they occur. The UX-designer identified inaccurate estimations as a typical cause to delays and did not mention it something that happens infrequently. This may be interpreted as a slight conflict in the importance of the risk between the developer and the UX-designer. Most likely with the architect as well, as the role did not even identify the risk. There lies a confusion in determining, whether the designer was referring to either the developers, the designers, or the team, when talking about estimating tasks. This makes it appear less apparent, which roles in the team the UX-designer perceives as the ones responsible for estimating inaccurately.

Throughout the individual feedback session, the UX-designer revealed a couple of risks and connections between risks unique to the particular role, which other roles failed to identify. The UX-designer identified lack of sufficient time for design and insufficient balance of tasks and explained their causal relationship: "*I have many tasks and therefore it is hard for me to*

finish enough tasks, so the developers can work with them". The researcher interpreted with the designer that they lack time to finish all of their tasks for the developers, which may lead to a lack of tasks for the developer. Then the developer must wait and cannot work. The UX-designer explained that the lack of time for design may be causes by:" *There may be something that causes a lack of time. It could be reprioritization of tasks shortly before something has to be ready".* This reveals the rationale behind the connection between the two risks in the causal map, in figure 4. The UX-designer stated that they also must receive clarification on certain tasks from their enterprise architects, which lately has resulted in the designer having to help the enterprise architects with the clarification. It usually does not happen very often, but currently it has occurred more rapidly- The UX-designer elaborated: "*This is because we are working with a new technology than previously, so it has contributed to a lack of time for me to design, when I have to help them with a clarification".* The designer must assist with the clarification, which both they and the enterprise architect needs. This may be explained to be, due to both parties need to understand, what they are saying yes and no to, in terms of certain tasks related to the new technology they are using. This takes up a lot of time and removes focus from his responsibility, which was analysis and design. The UX-designer accepted the causal relationship between the risk of lacking clarification from enterprise architects and insufficient design but did not offer an explanation. The researcher reasoned that the rationale behind the relationship may be, due to the lack of clarification on a task may, which may impact the quality of their design.

The designer was the only role to mention the risk of lacking clarification from enterprise architects, change in prioritization, insufficient balance between tasks and the two design related risks. As shown in figure 4. All the risks may in their essence be ascribed to causing an impact on Alpha's ability to comply with their deadlines, because they require Alpha to spend an additional amount of time and resources if the risks occur. The researcher reasoned that the risk of an insufficient design and insufficient balance lead directly to the risk of missing their deadlines. The three other risks were modelled on account on the previously presented explanations to have an indirect impact on meeting their deadlines, because they contribute to the occurrence of insufficient design and insufficient balance between tasks. The UX-designer's unique identification of risks and explanations related to analysis and design may be, due to the responsibility of the role. In contrast to the developer and the architect, the UX-designer mentioned no

technological risks and separates themselves from the other roles with their identification of design and analysis related risks.

Moreover, the UX-designer offered an explanation to the connection between the risk of key developer and the risk of having inexperienced developers. The developer described the inexperienced developers as a contributing factor to, why the key developer possesses the role, hence the connection from inexperienced developers to key developer. Although, the UX-designer explains the influence that the absence of the key developer has: "*It will have the consequence of our less experienced developers being more on their own".* The connection between the two risks were flipped here, so the key developer becoming sick has an implication for the inexperienced developers, where they lack the support of the key developer. The connection may be visualized in figure 4.

3.4.4 Business analyst

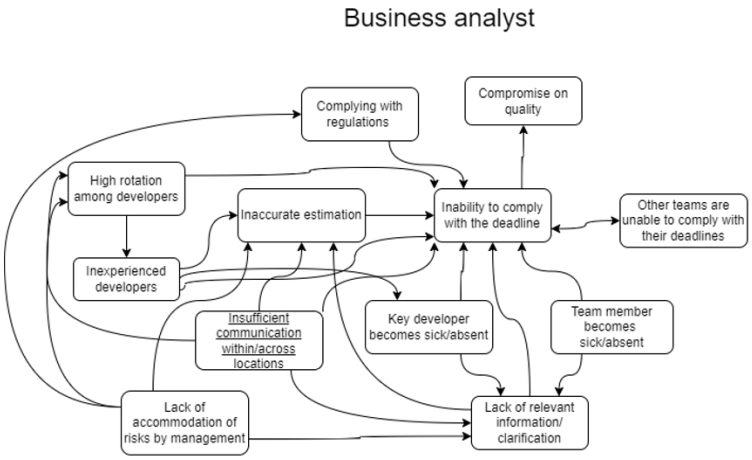

Figure 5: Business Analyst's causal map

The Business analyst shares a range of similar risks with the other roles and the related explanations. Like the developer and the designer, the BA mentioned the risk of complying with regulations. The BA explained, when regulations from the authorities emerge, they have a tendency of postpone complying with them. Complying with regulations usually have a deadline for compliance within a couple of years. The BA stated they handle the regulations, when they become a problem for their development and not before. The BA was the only role to explain the cause from inaccurate estimation very directly: "*It can really push to a project's deadline, as well that compliance and GDPR were not realized".* The BA thereby contributes to the explanation behind the causal relationship, between complying with regulations and their ability to comply with their deadlines. This may be observed in figure 5. The BA also shares the identification and explanation regarding the risk of inaccurate estimation, with the developer and the designer. The BA was asked about the causes to delays and answered: "*…or that we have simply underestimated, how long it will take".* This reveals the connection between inaccurate estimation and missing their deadlines.

It also shows the similarity in explaining the implications of inaccurate estimation in terms of the developer and UX designer.

The BA contributes to the pattern in the team's perception of the risk of high rotation among developers. The BA explained the issue in similar fashion as the previous roles, as being caused by uncompetitive salary and the high rotation implies that Alpha has to spent a lot of time and resources on training new developers. The BA stated: "… *but if someone comes along and offers a 40 % larger salary on the other side, then money quickly means more than having a cool team*". Which results in: "*So we are in a permanent loop, where we are always training people and that of course has a consequence in terms of our capacity".* The BA gave an almost exact replication of the issue as presented by the previous roles. The BA also contributes to the pattern of getting delayed, due to delays from other teams working on the same program. In contrast with the other roles, the BA explains that their own and other teams risk delaying each other, because management has recently decided to decrease the estimation buffer that all teams on a program has available. The BA explains:

"*Management has recently attempted to push it up, as to say that we would actually like to be on 90%, because our business wants more transparency on what you are spending your time on. When we are forced to go higher up, then there is also a risk of not delivering on time and we see that happen".*

This increases management's control of all the teams' time, but at the cost of limiting their freedom to manage their own time. It will inherently make it harder for Alpha and the other teams to act swiftly, especially, if something suddenly unexpected should happen, such as a risk of new regulations or sickness in the team. The researcher wondered whether Alpha's inability might cause other teams to be delayed, as well. The BA confirmed that their inability to comply with their deadlines, may also contribute to other teams getting delayed, which was lately, due to absence. The BA perceived similarly as the previous roles, sickness, or absence among team members as a restraint in reaching their deadlines. The BA differ from the other roles by explaining that absence may also be for a prolonged amount of time, due to maternity leave. It has the impact of: "*There is obviously a risk of overtime, and you have a lot on your plate".* This signifies an increased use of time and resources for the rest of the team members if the BA may be absent for a prolonged amount of time. According to the BA's view of himself as a central role in the team and the fact that taking over the unique tasks from the BA, may cause Alpha to delay their tasks. This explains the continued connection between sickness in

the team and the inability to comply with deadlines. Even though, the connection was not explained directly between the two risks, on account of the previous quote.

The BA shared the same perception of the risk of having a key developer and the key developer becoming sick. The key developer was the only one able to read the Danish documentation and management told the key developer: "*you cannot get sick, because you are the only one, we have and the next day she got sick".* The BA did not offer a specific explanation to, how the key developers' absence may affect their ability to meet their deadlines. The fact that management has told the key developer to not get sick implies management's concerns. The BA did also reveal that mostly management experiences the consequences of delays because they must take full responsibility. Management must make sure that their stakeholders depending on Alpha and the other teams' products to be delivered on time. This signifies that the BA illuminated that management and therefore in part also Alpha, were worried that the absence of the key developer might cause delays. This supports the pattern of Alpha being worried of the key developer's absence causes missed deadlines and the connection between the two risks. This may be observed in figure 5.

The BA was the only role to highlight the importance of certain consequences from insufficient communication. The BA perceived the issue of communication to mainly be across the different teams in the program. There was not a shared emphasis on insufficient communication across locations within the team with the developer. It was revealed: "*I often have issues with communication with other teams, if I want a quick answer on something, then I'm told to send a carrier pigeon".* The BA clarified that he could end up waiting a week for an answer that should take to minutes to answer and: "*It has been communication problems that has caused things to not be delivered on time or have a high enough quality".* This showed that the BA perceives insufficient communication as a risk with significant consequences for delivering on time and further supports the connection with the risk of lacking clarification/information. This difference in focus in terms of insufficient communication may be, because of the difference in responsibility and who the two different roles mainly communicate with. The BA works as the link between the corporation's wishes and the team, which entails a lot of communication with other teams within the program. Whereas the developer works closely with the Polish developers seated in Poland and the BA might not be as dependent on communicating with them. The BA did not experience the same issues with communication across locations, when asked: "*A little bit, I will say we are lucky in my team that we have been together for so many years, so I will say we are good at talking with each other in the team and be transparent".*

This difference in who both roles mainly communicate with, may explain the difference in their experience of insufficient communication.

The BA continued to emphasize the importance of insufficient communication, throughout the individual feedback session. Insufficient communication may also lead to: "*It can also lead to inaccurate estimation, because if I have explained something poorly to those that have to estimate it or we lack information, then we get bad estimates".* Estimation of tasks may also suffer from insufficient communication, although this was explained as being within Alpha and not in terms of other teams. This means the BA perceives the issue of insufficient communication to exist between teams and within the team, but not across locations. In contrast to the other roles in Alpha, the BA points out another explanation to the occurrence of high rotation among developers: "*It may lead to high rotation among developers, one can simply get frustrated, and we have experienced that several times"*. The role specified that developers have often experienced informing management about issues with for instance redundant IT-systems but experienced that management did not do much about it. This frustration in insufficient communication may lead to team members leaving the team. The issue of insufficient communication, especially with management revealed another risk. The risk of management ignoring issues and not wanting to accommodate with the risks. This was explained to potentially cause team members to leave the team and lacking information and clarification from management, which may: "*It can lead to bad estimations, it can lead to complying with regulations, all over. Because then we have this loop".* The risk of management not taking risks seriously and lacking clarification from them, may cause the occurrence of several other risks. Similar to that of insufficient communication.

The BA identified insufficient communication as the most important risk and the risk may also be viewed as the most central, according to the BA: "*For me in my position, then it is usually lacking communication or the lack of listening to communication that causes many of these problems".* As shown in figure 5, insufficient communication leads to three risks in the causal map, but according to the BA it may contribute to the occurrence of many other risks. This signifies the BA's unique perception of insufficient communication compared to the other roles in Alpha, which may be explained aside from the fact that communication was a central part of the role's responsibility. Communication may be further ingrained in the role: "*I have a background in communication, and as far as I can see that is usually, why something goes wrong".* The role was most likely talking about their educational background, which further

contributes to their personal bias in perceiving issues with communication as the root to most of Alpha's concerns.

Aside from being concerned with insufficient communication and lacking clarification, the BA offered a similar explanation as the developer and the designer, when it came to inexperienced developers. The inexperienced developers contribute to inaccurate estimates, but the BA clarified another cause to inaccurate estimates, aside from the other roles: "*Are we talking about management level, where someone estimates? Or is it inexperienced developers, who has to estimate further down on story level. Both are equally relevant".* The role uniquely identified management as a contributing factor in terms of inaccurate estimates. This also revealed, the issue of inaccurate estimates may be an issue for Alpha on more levels than one. The BA identified the issue of estimation on low level user story tasks, but also when management estimated more general tasks. This increases the power of the concern. The fact that the BA was the only role to explain this other layer, may be ascribed to the role's function and responsibility. The BA was responsible for communicating with different levels of management within the bank, which makes the role more inclined to know of inaccurate estimations made by management. The role may therefore be more concerned with management's part in inaccurate estimates, but also when it comes to lacking clarification from management and the fact that management tends to ignore risks.

Moreover, the BA and was one of the only roles, who explained how the prospect to missing deadlines may cause a compromise on their products' quality. The BA stated: *"If we are pressured on time, because of all the reasons we have mentioned here, then it may be that we say, we have to get this out and then we have to cut some of the things that are more nice-to-have".* This supports the rationale behind the connection between the risk of missing deadlines and compromise on quality. It was clarified by the BA that part of their work was to make sure their deliveries live up to the requirements, which means the expected quality. On the one hand, it therefore makes sense that the BA was the only role concerned with compromising on quality. On the other hand, the other roles may perceive the fact that they can never reach a perfect level in expected quality as a condition, when developing IT solutions.

Even though, there was only one person occupying the role of BA, the BA managed to identify a significant number of risks and related explanations, compared to other roles with the same number of role owners. The BA identified an equivalent number of risks compared to roles with more than one role owner, such as the three previous roles, but offered more explanations

compared to all the roles in Alpha. This had an unfortunate consequence for the complexity of the causal map and the observer's ability to comprehend all the causal relationships in figure 5. Compared to all the other roles in Alpha, the BA was the only role to identify the most risks related to the structures within the corporation, whereas the other roles were mainly concerned with issues caused by the actors in the team or the difficulty with their technologies.

3.4.5 Product Owner

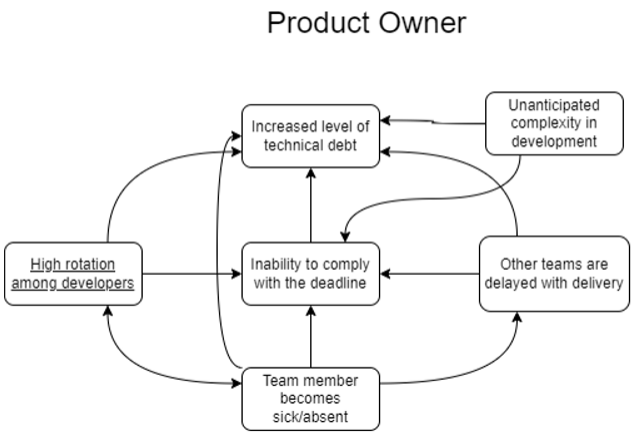

Figure 6: Product Owner's causal map

The PO separates himself from the other roles by being the role that managed to identify the least number of risks and related explanations. The PO did contribute to the pattern of being delayed by other teams. The PO placed a greater emphasis on delays caused by other teams: "*That is our primary risk that other teams have to deliver to us, before we can do our part".* The risk of being delayed by other teams was perceived as a central risk by the PO and Alpha's own delays as well: "*A potential delay is where our biggest risk is".* These risks were explained as causing the team to miss their deadlines and postpone their tasks to the next sprint, which was also explained by many of the other roles. This causal relationship may be observed in figure 6.

The PO's greater emphasis on delays compared to the other roles, may be attributed to the PO's primary focus: "*And that is what my job consists of, make sure that these priorities are in place, make sure that we always have the best plan as possible, so we can deliver what we have promised on time".* The PO revealed that his role's primary concerns were, due to his role's job function. The fact that the PO's primary tasks primarily consist of prioritization and planning, may also explain the role's more general view on risks, compared to the other roles. Each of the previous roles have mentioned risks similar to the others and risks uniquely essential for their specific role. The two aspects may contribute to explaining that the PO was only concerned with general team risks and not risks uniquely important for each role in the team. It was simply not part of the PO's job description to focus on risks unique to each role, where the PO's focus lies with planning and prioritization of tasks.

Furthermore, the PO did contribute to the pattern of identifying the risk of team members becoming sick and its causal relationship with missing deadlines. The PO identified the risk of

team members suddenly becoming sick, which was identified by all the previous roles as well. The PO similarly identified the risk's causal relationship: "*Yes, you can say the negative impact is that our project as a whole gets delayed".* The PO explained the same implication sick team members has for meeting their deadlines, as the previous roles. This indicates a consensus within the team of team members becoming sick, which may imply that the risk could be an inevitable reoccurring risk for Alpha. The team cannot do much to change the fact that people get sick, but Alpha should be aware of its potential implications for meeting their deadlines.

Throughout the individual feedback session with the PO, it became apparent that the PO identified the risk of technical debt, which the developer also did. Although, the PO has a reversed perception of the causal relationship between technical debt and meeting deadlines, when compared to that of the developer. The PO explained: "*When we are under pressure, we have to make cuts somewhere and deadlines are typically the most important, so it is often just technical debt we build up. This is actually what we are seeing at the moment.".* This reveals that the PO perceives missing their deadlines as a cause to increasing their technical debt, instead of the causal relationship being the other way around. This stands in contrast to the developer perceiving the generation of technical debt as one of the most central causes to missing their deadlines. The two roles may agree fundamentally on the same connection between risks, but the developer was arguably more concerned with the increase of technical debt causing delays on the project sometime in the future. The PO focused on the nearest deadline, if they were to potentially miss their deadline, they may have to accept generating technical debt to meet their forthcoming deadline. Thereby they avoid getting delayed and postponing the tasks for the next sprint. Their difference in their perception of the implications of technical debt, may be attributed to a differing focus on the time frame of the project. The developer may also have a more specific view of which kind of technical debt they may have accumulated, compared to the supposed knowledge of the PO. This would be, due to the developer was responsible for the development of functions and can see which specific functions that may have been neglected and potentially cause an impact on their available time and resources in the future. The technical debt must be dealt with at some point and the PO may not be able to always be fully aware of the nuances of their current level of technical debt.

The PO continued to provide a unique cause to the generation of technical debt and their ability to comply with their deadlines, compared to the other roles. The PO explains the causes after they were proposed by the researcher: "*It naturally could that is obvious. It is the one with whether our deadlines cause the technical debt directly*". The researcher proposed a connection

between high rotation and team member becoming sick with missing their deadlines. This causal relationship may be observed in figure 6. The rationale was as the PO explained, that they may generate technical debt to reach their deadlines, but their ability to meet their deadline may have been influenced by team members leaving the team. They will then experience a decrease in their work capacity, which may have an impact on their ability to comply with their deadlines. The same can be said of the generation of technical debt, although according to the PO, rotation among developer's impact on technical debt, may be more indirect than on their ability to meet their deadlines. The PO identified the risk of high rotation much like all the other roles in the team, and attributed its existence similarly to be, due to the uncompetitive salary offered to more inexperienced developers in the team.

The PO was the only role to identify the risk of unanticipated complexity in development, which seemed more like a risk that the developer or architect would have identified, because of its technological nature. The PO stated: "*There is always this risk of some tasks being more complex. It will cause us to not meet our deadlines, or the alternative is that we build up more technical debt".* This unexpected complexity may lead to a task taking longer time to complete than anticipated, which may cause Alpha to be unable to comply with their deadlines or accumulate more technical debt instead. When looking at the PO's causal map and compare it to the developer and the architect's map, some similarities across the maps materialize. The developer explained that the generation of technical debt was caused by redundant or insufficient code and lack of maintenance of open-source libraries. The architect agreed to inadequate development platform, outdated technology stack, lack of maintenance of libraries and the time-consuming API review process as contributing factors to their inability to comply with their deadlines. All of these risks were related to development.

This implies that the unexpected complexity in development the PO identified, may be an aggregation of the technological risks identified by the developer and the architect. Taking the PO's role into consideration, the PO would not be concerned with the exact details of what the unexpected complexity in development specifically entails. As mentioned earlier, the PO was preoccupied with prioritization and has a more general view on risks compared to the other roles, who are responsible for developing the prioritized tasks. This implies that the PO may not be more limited in their perspective on risks, but simply should not be concerned with the specific details of, which technological issues or unexpected complexity that the developers may encounter. This is part of the developers' job and not the PO. Additionally, the PO lacked more specific team risks and the risks that may more indirectly lead to the risk of missing

deadlines and generating technical debt, compared to most of the other roles in Alpha. The PO did in fact identify the risk of high rotation as the most important risk, which four of the other roles also did. This revealed that the PO may only focus on the most essential risks for the team, which may be communicated through their PI-planning meetings and their sprint retrospectives. The risk of high rotation was also identified during the PI-planning meeting that the researcher attended during the process of data collection.

3.4.6 Scrum Master

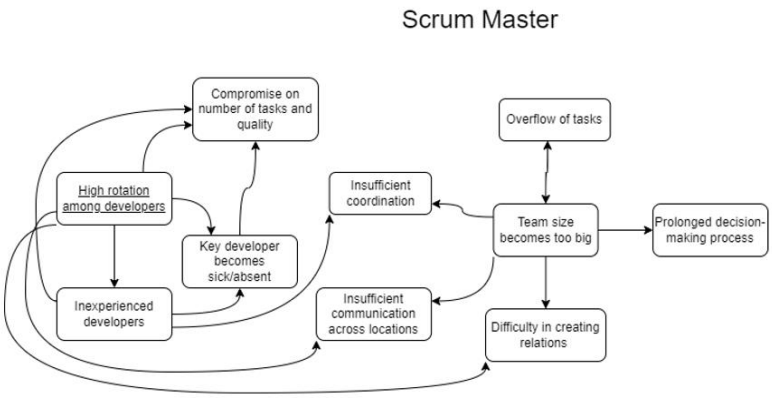

Figure 7: Scrum Master's causal map

The SM differentiates from the other roles by being the only role, who did not identify the risk of missing deadlines and getting delayed. This is, due to: *"We just take less into the PI. Then management may have set some deadlines, they must move, but it is not something we experience as a team".* The SM contrasts with all of the other roles, as they did not perceive delays as a concern for the team at all. The SM was solely concerned with their team becoming too big. The SM explained the cause to potentially becoming too big in the future: "*In my opinion it is caused by the fact that there are so many tasks on our table… we can see there are many tasks coming our way".* This emphasizes, there is currently an overflow of tasks that Alpha has to take care of and: "*Therefore we need to take in more developers into the team to solve it".* The overflow of tasks entails Alpha to increase their number of developers in the team, thereby potentially becoming too big. At the same time: "*The more you put in, the more tasks you receive".* This means that an increase in the team's size, may further increase the number of tasks in their table and likewise, an increase of tasks may increase the team size. In continuation, the SM is worried whether their team size increasing, may lead to even more tasks on their table.

Alpha is the only front-end team in their program, which means all the front-end tasks have to be solved solely by Alpha. This may contribute to explain the risk of Alpha potentially becoming too big, because an overflow of front-end tasks may further contribute to further increasing the capacity of developers in the team. The former explanations constitute the rationale behind,

the two-sided connection between the risk of the team becoming too big and overflow tasks, they are mutually related. This may be observed in figure 7.

The team consists currently of 14 team members and part of the SM's job is consulting the Scrum Guide, which points to: "*I think they say 10 or fewer and our team has reached a size, which is quite large, and may be difficult to manage".* Alpha is approximately four people more than what the Scrum Guide recommends and may explain the source of the SM's concern with the team becoming too big. The Scrum Guide implies that the larger the team becomes, the more ineffective will the team become. The SM is worried that increasing the team's size by hiring more developers, may not result in Alpha becoming more effective.

The SM clarified uniquely that the overflow of tasks and increase in team size, may have an impact on many of the other roles' effectivity. The SM mentioned, the developers may end up blocking the way for each other, especially during the review and test of code. This may be very likely to happen, as the architect already identified the issue of their review process of API's as being time-consuming. The SM continued to clarify: "*But because, when you put more developers into a team, then you also get more tasks and there is therefore a lot the PO has to be able to manage".* The PO may end up losing the overview of what the everybody is working on in the team and how they are progressing on their tasks. This contributes to the previous argument, stating that the PO has a more general view of the team's concerns. If the team size increases, the PO will likely also have an even harder time being up to date with the team's concerns and potential impediments for their deliveries. The SM continues to explain that the BA is also challenged if the team receives more developers and more tasks. The BA is only one person occupying the role and the role will inevitably have to work much faster to continue to produce enough work for all of the developers. The SM also explained this in terms of the PO: *"A PO is just one person. You do not have two PO's in a team".* And about oneself: "*I am also just one person and there won't be coming anymore Scrum Masters into the team".* According to the SM, the PO will have greater issues with sustaining an overview of the team's work and progress in terms of meeting their deadlines. This is if the team potentially get bigger and they receive an overflow of tasks. Whereas the SM is more concerned with the potential issues of communication and decision making.

The SM explained certain aspects, where Alpha's effectiveness may be affected. The Scrum Master explained the hiring of more developers, who most likely would be seated in Poland, may lead to: "*…and that of getting a close connection with them I think can be difficult, when*

you are seated in another location". It may become even harder for the team members and the SM to create relations with the developers in Poland, especially if management decides to hire more developers to accommodate with their current overflow of tasks. The fact that management may hire more developers from Poland, may create a state, where Alpha consists of two more separate teams with separate cultures. One in Denmark and one in Poland, who still has to function as a single unified team, which may end up creating more distance between each location of the team. The SM continued to explain that the team may also suffer from having an even harder time with coordination and making a final decision. This is highlighted through this quote: "*We can already now sense that we may get more, but I can feel that of coordination. Agree on things. We are many chefs, who has to agree on things, when you reach 14 people in the team".* Alpha may experience difficulties with coordination, when they become more people, who may also be seated in Poland. It will also gradually become more difficult to agree on things and make a timely and satisfactory decision. This signifies the arrows pointing towards the two risks with the same names in figure 6.

The SM's unique focus on risks related to other roles and the unique identification of the issues in the team potentially becoming too large, may be attributed to the SM's unique function. The SM explained that she is responsible for protecting the team from impediments and makes sure that Alpha is following the guidelines from the Scrum Guide as good as possible. This means the SM is most likely the only team member, who is comparing their team's work and composition with that of the Scrum Guide, which may explain the unique focus on team size. It may also explain the role's unique lack of focus on getting delayed and missing deadlines. The SM is responsible for making sure that all the other roles have everything they need, so the team can operate to the fullest of their ability and is not responsible for solving tasks that may be delayed. This is primarily, what the other roles are responsible for, so it makes sense that the risk of missing deadlines is not part of the SM's main concerns.

Furthermore, after the researcher received feedback from the SM, it became apparent, the role was now more concerned with the team's high rotation and not the risk of the team becoming too big. Until the feedback session the SM was the only role, who did not identify the risk of high rotation. There has recently been a couple of resignations in the team, which has made high rotation of more immediate importance. The risk of the team becoming too big has obviously decreased in its likelihood of potentially happening in the future, when Alpha has just lost a couple of colleagues. The likelihood of the team size becoming too large, may regain its similar importance as before, when the gap in their workforce has been filled. The SM

explained in contrast to all the other roles that high rotation, may also have an impact on the team members' ability to create relations in the team: "*If rotation happens all the time, then you do not get to know them and then you do not become a team. That takes time".* The SM mentioned earlier that a larger team, may cause the team to potentially have difficulties with creating relations in the team, especially across locations.

After the high rotation has increased in its importance, the SM realized that this high rotation also makes it difficult for Alpha to create trustful and comfortable relations to new employees. They constantly have to get to know new people, which makes it harder to have a coherent team. This highlights the causal relationship between high rotation and difficulty in creating relations, which may be observed in figure 7. This also showcased the fact that difficulty in creating relations is caused, both by an increase and a decrease in Alpha's size. The same can be said of insufficient communication. The SM identified it as a risk, and clarified that it too, may be caused by an increase and a decrease in Alpha's size. The SM reflected over it and said: "*It is both with too big and too small. That is why the theory is also looking for this ideal number for the capacity to function optimally".* The SM explained this through the Scrum Guides recommendations. There is a reason, why it recommends a team size of approximately 10 people, when the team size becomes lower than recommended or higher than recommended the same risks may potentially occur.

The SM now places the same heavy emphasis on high rotation, as all the other roles, besides the BA. The SM also shares many of the same perceptions of its consequences and its contribution to the occurrence of other risks, as many of the other roles. For instance, the high rotation may lead to hiring new inexperienced developers, as mentioned earlier, and that the inexperienced developers must be trained by the key developer. As the SM explained: "*It is the developer working with the code on a daily basis, who has to train the new developers".* This places a pressure on the key developer to not get sick, because it may not only put the key developer's tasks on hold, but also that of the inexperienced developers, dependent on help. Where many of the other roles identified the risks; high rotation, inexperienced developers and key developer becomes sick as contributing factors to missing their deadlines and getting delayed. The SM did not place much emphasis on delays as mentioned earlier and would only accept the risk of compromising on the number of tasks and their quality, because the three risks cause: "*But definitely the capacity and the resources decrease and therefore we can maybe not come as far down the prioritized list, as the ones further up might want us to.".* They limit Alpha in terms of their capacity and resources, which cause Alpha to neglect tasks of lesser importance and to

compromise on the quality of the tasks of high priority. The risk of compromising on the quality of tasks and its related cause, was also identified by the BA.

3.4.7 Summary

The researcher has constructed the causal maps in cooperation with the different roles in Alpha. The process of constructing the maps presented a handful of difficulties, especially in assuring the reliability of the interconnectedness between the identified risks for each role. This caused a feeling of insecurity in the researcher, which was partly alleviated after the conduction of the individual feedback sessions.

Throughout the analysis, it was revealed that Alpha was mainly concerned with their tasks getting delayed and the risk of high rotation. There was a consensus of the cause of high rotation among the roles in the team. All the roles identified the risk and provided the same explanation to its existence, which was due to the uncompetitive salary for inexperienced developers. Most of the roles identified missing their deadline as a risk, but the roles differed in their explanations. The developer and especially the architect, were more prone to attribute missed deadlines as being, due to technological risks, compared to the other roles. The PO had a more general view on risks compared to the other roles but proved to be generally aware of the unexpected complexity that the developers, may experience during development. The BA was mostly concerned with the lack of and insufficient communication across teams and within Alpha, which the BA perceived as the main cause to his and Alpha's concerns, including missing their deadline. The UX-designer shared a large amount of identified risks and the related explanations with the other roles but was mostly unique in their focus on design and task related risks.

The SM stood out from the other roles by being the only one to not perceive the risk of missing their deadlines, as an essential risk. The SM was concerned with the team's impediments, primarily the risk of Alpha becoming too big. Alpha was generally unanimous, when it came to their identification of risks and their explanations to the relationships between them, with a few differences in their perceptions of risks, which was primarily attributed to the focus and function of the different roles.

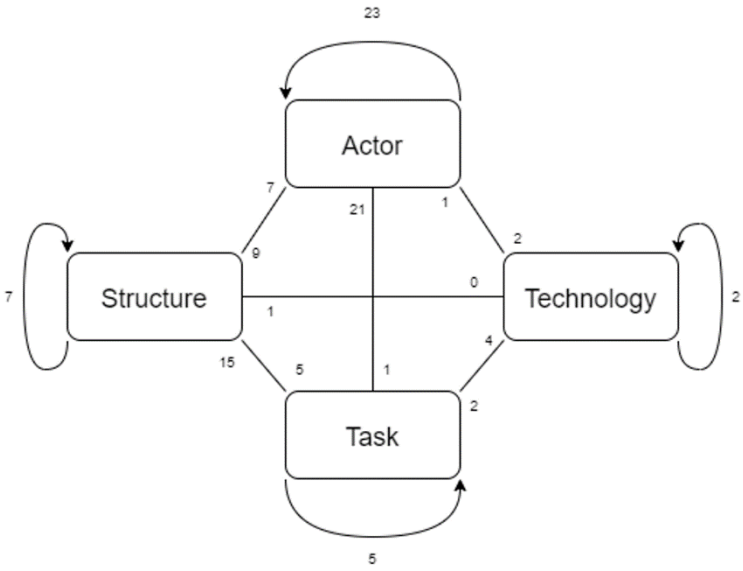

Figure 8: Leavittian model used to categorize connections between identified risks across all roles

Furthermore, to gain a clearer overview of the identified risks and their interconnectedness, the leavittian model will be used for categorization. Figure 8 depicts the categorization of Alpha's concerns, which was an accumulation of all of the leavittian models made for each role. These models may be viewed in appendix 2.2. The arrow pointing to its own category denotes, when a risk from a specific category contributes to the creation of a risk belonging to the same category. When observing figure 8, it becomes apparent that Alpha perceives most of their concerns to be related to their own limitations as team members in Alpha and the structural concerns were of secondary priority. Technological risks were perceived as the category of least concern and were only identified by the two technical roles. Whereas Alpha was predominantly concerned with the identified risks causing the creation of task related risks, which may be a rational priority for a software team. This may be explained, through the fact that Alpha was predominantly concerned with their tasks being finished on time and in a satisfactory manner and was at the same time primarily concerned with their own limitations causing the task related risks.

Alpha's shared focus on their own limitations as a team may be, due to the fact that Alpha has an active influence and greater overview of their own limitations as team members. The same cannot be said in relation to the structures within Estate bank and their current technologies, which Alpha may not realistically be able to influence. Decisions regarding the structural hierarchy in Estate bank, as well as which technology they were using was decided by management. These types of risks may therefore not be perceived as having high priority when it was not generally part of Alpha's responsibility as a team. It may be rational for Alpha to primarily focus their concerns on the aspects, they can realistically have an impact on and make sure that these risks may not contribute in the team getting delayed or they have to comprise on their deliveries. Looking back at the types of risks primarily identified by the PO, SM and BA, as well as their responsibility as a role, it makes sense that they also were concerned with structural risks. Whereas the architect and the developer were the only roles to identify technological

risks. Alpha's focus in terms of their concerns as a team and their different roles may simply be attributed to the responsibility and goals of a software team and the different roles within the team. Management of Estate bank would most likely be pleased to know that Alpha was mainly concerned with their own limitations as individuals and not allowing it to impact their ability to deliver the promised quality on time.

Finally, the different causal maps were explained and presented to Alpha during the workshop. The feedback generated from the workshop contributed to alleviate a significant part of the researcher's experience of uncertainty when it comes to the plausibility of the causal maps. The different roles within Alpha were overwhelmingly positive towards the maps. Moreover, Alpha found the causal maps to be interesting and were able to identify a range of uses for the maps in their team. One of the developers mentioned that it was interesting to be able to see the differences between the identified risks among the roles. The same developer clarified, why it was interesting: "*How we see some risks differently, because they directly touch one role and not necessarily another".* Where the developer found the concrete differences between the roles interesting, the BA was interested in another difference: "*I also think it's interesting to see the different levels of the same type of risks, as you mentioned this sort of abstraction".* The BA found the differences in the levels of abstraction between the roles interesting. The statements from the developer and the BA reveal that the different roles in Alpha, may have learned something new about the concerns of the different roles in their team. The roles have thereby broadened their perspective and knowledge of each other's limitations, throughout their latest project.

This may be further supported by the statement from a UX-designer: *"For me I think it would just be interesting to get these pictures and shove them up on the wall and have people walk by, ask about them and we can explain what we have been a part of".* The UX-designer goes a bit further than the two previously mentioned roles, by stating that the team can benefit from the maps by hanging them on the wall and utilize them as reminders of the different roles' concerns. In addition, the designer revealed that not only Alpha, may learn from the causal maps, but also colleagues passing by the maps. One of the architects continued to explain the use of the maps in relation to other teams: "*It could be interesting to interview maybe another team to see, if they have the same profile".* The statement from the architect implied that Alpha could see value in comparing their concerns with that of other teams, which may provide them with a greater insight into other teams' concerns. The comparison with other teams may also contribute with a more holistic view of all the risks belonging to all of the teams working on

the same program. One of their new developers, who has previously worked at another software team within *Estate Bank*, explained: "*I agree on all of them. The only one I have not met before is of course the open-source library. I think that is a Front-end thing. But the rest of it I think you would find that most of the teams are kind of hit by".* The new developer verified that the identified risks may very well also exist in the other teams in *Estate bank,* except issues with open-source libraries, which was a unique concern for a front-end team. This unambiguously imply the new knowledge that Alpha has acquired from the maps and the value in using them as a knowledge sharing tool between the different teams. They can identify the risks that have existed for a long time across all of the teams and thereby gain a better overview of, which risks that needs to be addressed first.

Several roles identified the use of the causal maps as a tool for knowledge sharing within Alpha, a designer mentioned: "*Yeah, that was my thought also that this could be used as an input to a risk management discussion".* Alpha can use the causal maps to set the stage for a meeting on their approach to risk management, where they can narrow down what is the most severe risk and what they should attempt to address. The SM continued to explain more specifically: "*But also, we could use it for a retrospective and talk about it inside the team, as well".* The designer was reflection on the maps' use in more general terms in relation to risk management, whereas the SM identified the applicability of the causal maps, when they have sprint retrospective meetings. The SM was also interested in discussing the most important risks for each role during the retrospective meeting. This may reveal that the causal maps for each role can act as a tool for discussing the changes in the severity of different risks, which may ease the process of prioritizing the risks that needs to be addressed.

The architects continued to elaborate on the use of the causal maps across teams as a knowledge sharing and documentation tool for their PI-planning meetings. An architect explained: "*I also think that this whole root cause analysis of these impediments that we have identified, where you can use this map to identify, which are connected and that's also a thing we haven't been very skilled at doing and we have stopped doing it for several PFPS".* The architect revealed that the causal maps, may be valuable in providing the different teams with an overview of the interconnectedness between the different risks, instead of only considering risks in isolation. The designer continued to clarify on the use of the maps as a knowledge sharing tool: "*I also think this history perspective can help warrant and investigate in handling a risk, so if we see the same risk over and over again it should actually be an argument that we can spend some money and time on addressing the risk".* This highlights the fact that the maps can act as

documentation of previous risks and as a knowledge sharing tool for management. Alpha can showcase the severity and frequency of a risk in a more explicit and dynamic manner using the causal maps, compared to that of a risk table. A risk table ignores the connectedness between the identified risks and presents the risks in a static way, which may present an oversimplification of a software team's identified risks. It has been established that the causal mapping method potentially has a long range of practical contributions for a software team. This was especially apparent, when it came to utilizing the method as a tool for knowledge-sharing, making the implications of certain risks more explicit for management, for discussion of risks during sprint retrospectives and as a comprehensive documentation of past project risks.

4. Discussion

This section will be divided into four subsections and will discuss the 1) contribution 2) limitations 3) implications for practice and 4) relevant future work concerning this thesis.

4.1 Contributions

Besides the previously mentioned contributions for practice, this study managed to contribute methodologically as well. This thesis contributes to extending the IS research regarding utilizing causal mapping as a method of identification and analysis of risks, when it comes to software teams' perceptions of risks from past projects. The literature focuses extensively on eliciting users' and stakeholders' perceptions of barriers or risks in terms of a technology or system, (Ackermann and Eden, 2005; Ackermann *et al.*, 2014; Leonhardt Kjærgaard and Blegind Jensen, 2014; Ackermann and Alexander, 2016) whereas exploring the residing perceptions of project risks within a software team are virtually none existent, except (Al-Shehab, Hughes and Winstanley, 2004, 2005).

This study addresses the scarcity within the literature on software teams' perceptions of project risks. Previous research by Al- Shebab et.al adopt a generic view on a software team, focusing solely on project managers and groups the roles of the team members into one. Additionally, they group the perceptions of risks stemming from, both the project managers and the team members into one single causal map. Their holistic view on risks within a software team leaves out the essential perceptions from the different individuals in the team and most importantly the different roles within a software team. As it was shown in the findings section, there are similarities as well as differences between the roles, which reveal the danger of leaving out the perceptions among the different roles. This study thereby addresses the research gap regarding

the lack of adopting a role-based view on software teams' perceptions of project risks, in contrast to the holistic view on software teams.

The study also provides unique insight into three Scrum roles, when it comes to their perceptions of risks as the three Scrum roles have not been examined in this manner before. These include the PO, SM, and developer, whereas the last three roles, BA, architect, and UX-designer are more unique to the industry of software development and the *Estate Bank.* Additionally, this thesis extends the notion from Al-Shebab et.al that the causal mapping technique may assist software teams as a post-evaluation method after the duration of a project. The study of the Alpha team shows its usefulness, as a post-evaluation method during sprint retrospective meetings.

Furthermore, the scholars using the causal mapping method within the IS literature tend to use case studies as their research approach, (Al-Shehab, Hughes and Winstanley, 2004; Ackermann *et al.*, 2014; Ghobadi and Mathiassen, 2014) whereas the adoption of the action research approach is more scarce (Ottosen, 2019). Determining the causal mapping researchers' exact research approach allegiance was a cumbersome task, as many of the scholars utilizing case studies seem to adopt some properties from the action research approach (Ackermann and Eden, 2005; Ackermann *et al.*, 2014; Ghobadi and Mathiassen, 2014). This entails the final contribution of this thesis. The researcher decided on the action case study approach as the research approach, which in contrast to other causal mapping scholars, makes it clear from the beginning, that both elements from case studies and action research will be utilized. If hardly any scholars use the action research approach, when utilizing the causal mapping method, then scholars making use of the action case study is non-existent. Thus, this thesis contributes with a unique research approach, when using the causal mapping method and accommodates the research gap in the use of the method. The adoption of the action case approach presented some difficulties and limitations throughout the execution of this thesis, which among other things will be discussed in the next subsection.

4.2 Limitations

The action case approach contributed with the explanatory power of the case study approach and that of intervention from action research. Although, the action case study lends the strengths from both approaches, it may similarly adopt the weaknesses of both approaches. This thesis was therefore limited in utilizing the full potential of the properties from the action research approach, which limited this thesis in being able to examine Alpha's adoption of the

causal maps. Given the lack of testing the applicability of the approach with Alpha, this thesis is limited to answering the question of, whether causal mapping is an effective documentation and knowledge sharing tool for expressing risks and their implications explicitly and as a post-evaluation tool during the sprint retrospective etc. for Alpha and other software teams. This alludes to a sense of uncertainty regarding the actuality and efficiency in adopting causal maps and causal mapping in software teams.

Moreover, this study was also limited when it came to the creation of the causal maps. The different role owners from Alpha constructed their own subjective perception of risks and their causal implications. These were articulated to the researcher primarily through interviews with the role owners, where they were subject to the influence and framed by the questions asked by the researcher. The constructed causal maps signified a socially contrived perspective of how different roles in a software team perceived past project risks and represent a subjective explanation to risks created, through the interaction between the role owners and the researcher. This limitation of causal mapping falling subject to subjectivity and influencing the trustworthiness of the causal maps, was also identified by other causal mapping researchers within IS, such as (Leonhardt Kjærgaard and Blegind Jensen, 2014). Kjærgaard and Jensen (2014) asks if the causal maps can be trusted to not leave out any relevant issues to the participants, which may not be achieved as a true representation of all of the participants' perceptions is not possible. We as researchers claim trustworthiness and representativeness over the causal maps on account of systematically examining the issues mentioned by the participants (Leonhardt Kjærgaard and Blegind Jensen, 2014). In this thesis, the credibility of the constructed causal map was ensured by conducting individual as well as collective feedback meetings with the role owners from Alpha.

4.3 Implications for practice

This thesis has implications for agile software teams concerned with risk management. When agile software teams engage in identifying and analyzing it is essential to illuminate the perspective from the different roles within the team, instead of focusing solely on software teams' risks holistically. This process typically produces a prioritized list of risks, where each risk is viewed individually without considering the implications and relationships between risks. Agile software teams may benefit from the utilization of causal mapping to provide a more dynamic and detailed presentation of risks than risk lists. As change is inevitable during a software project (Sommerville, 2016, 61), causal mapping may also provide agile software teams with a method to anticipate potential changes and further implications of risks. The dynamic

and systematic illumination through causal mapping fits with the iterative and changing nature of agile software teams and projects. The risk lists do not embrace change in the way as causal maps, which highlights its value for agile software teams.

If causal mapping is to be adopted by agile software teams, it may be beneficial to assign the responsibility of constructing the causal maps to the SM. This same is due to the SM's existing responsibility as the software team's facilitator of Scrum events and is interested in finding methods that may improve the effectiveness of their team. As the causal maps may assist Alpha in discussing and identifying risks during sprint retrospectives, where the SM act as the facilitator of the meeting, it may be appropriate to have the SM construct the maps as well. The SM should be aware of the time consuming process of using and constructing the causal maps (Leonhardt Kjærgaard and Blegind Jensen, 2014,). The researcher acknowledges the experience noted by Kjærgaard and Jensen (2014) that even though the process of constructing and using the causal maps is time consuming, the investment is worthwhile for practice.

4.4 Future work

The researcher was reminded by one of the participants during the workshop feedback that the causal maps were a snapshot in time. Alluding to the fact that the researcher may experience a difference in the results if the causal maps were made a year from now or in the past. Although, some of the identified risks and their causal relationships most likely will continue to be present in the future, as they may also be perceived as conditions of software development. It would undoubtedly be profitable to conduct more longitudinal studies in terms of not only Alphas, but other software teams' perceptions of risks. Whilst the causal maps were static snapshots of Alpha's perceptions of risks in their latest project, some risks may also change throughout the course of one project and future projects, as priorities change during software development.

It may likewise be interesting to dive further into the software teams within *Estate Bank* and examine a second software team, acting as team Beta. This would have provided this study with the ability to compare the perceptions of risks and explanations of their causality across the roles between team Alpha and Beta. This may have contributed with a greater sense of representativeness and a higher level of trustworthiness of the causal maps. This is achieved through improving the degree of data triangulation, when the number of units of analysis is increased and the case becomes a multiple case instead of a single case. It may also be interesting to compare the results from the software teams in *Estate Bank* with software teams from similar organizations and organizations outside of banking. The identified risks and the related

explanations of their causes may be compared to software teams sharing a similar context and a different one.

Additionally, the usefulness and effectiveness of the causal maps in a practical context may be further investigated and it may be deduced whether other software teams perceive the maps as being useful as a knowledge sharing, discussion, documentation, and post-evaluation tool. It may also be alluded whether the software teams adopt the causal maps and for how long, as well as which issues, they may encounter, whilst using them. This probes for more action research in the use of the causal mapping method as a way of identifying and analyzing risks, so the applicability and of the causal maps may be further examined. In this case longitudinal studies utilizing action research may also contribute to evaluating, whether the resources spent on constructing the causal maps is proportional with the value it provides for a software, such as Alpha.

As a final notion in terms of future work, the researcher excluded the upper management layers of the *Estate Bank* from the scope of this thesis, which may be relevant to consider in a future study. This was also identified by the participants during the workshop, where they pointed that it would be interesting to showcase management one or more causal maps to probe for answer to why a certain risk is accepted. Interviewing one or more managers from *Estate Bank* may have provided a more thorough validation of the function and effectiveness of causal mapping, when it comes to using them as a knowledge sharing and discussing tool. Including managers as a role in the study would also allude to further inquiry of the effectiveness of causal mapping as a tool for management as well, as the roles in a software team.

5. Conclusion

This thesis was concerned with examining the problem statement: "*How can an agile software development team explain diverse causes of project risks?"*. The researcher collaborated with a software team from *Estate Bank* called Alpha, which constituted the context of this study. The problem statement was formulated on account of the identified issue that Alpha communicated during the first meeting, between the researcher and the software team. Alpha stated they had an issue, when it came to learning from risks they have encountered during previous projects. The action case study was chosen as the overall research approach to investigate the presented problem formulation, which entailed the explanatory power of case studies and the intervening nature of action research. To uncover the risks and the related causes, the researcher interviewed team members from Alpha, observed a retrospective meeting and a PI-planning

meeting, and reviewed archival data documenting previously identified risks. The three different data collection methods were used to obtain a higher level of data triangulation.

Moreover, the causal mapping technique was selected as the specific tool to explain and showcase diverse causes of projects risks. The researcher focused on constructing causal maps for each role in the software team, which included PO, SM, BA, developer, architect, and UX-designer. This focus on roles induced the possibility of showcasing similarities and differences between the roles' perceptions of risks, rather than focusing on risks holistically. The researcher argued that a holistic view on a software team causes software teams to overlook some relevant risks and perceptions. Whereas the construction of software roles' identified risks and analysis thereof, through causal mapping provided a more dynamic and systematic view on risks, where the causality between the identified risks were included. This perspective on risks stands in contrast to the production of risk lists, where identified risks are considered in insolation of each other. The causal mapping approach offers an overview of identified risks, where the causality and interconnectedness between the risks are also considered.

The researcher experienced a sense of insecurity when it came to the trustworthiness of the constructed causal maps. This was accommodated through receiving individual feedback from a representative from each of the six roles and a workshop held during a sprint retrospective meeting. During the individual feedback sessions, the researcher focused primarily on the content of the causal maps, where the workshop primarily provided feedback in terms of the applicability and usefulness of the technique within the context of Alpha. Alpha managed to identify a potential use of the causal maps as a knowledge sharing tool, a starting point for documentation of risks, post-evaluation, and discussion tool during sprint retrospectives, as well as a tool to communicate risks more detailed and convincingly to management.

References:

Ackermann, F. *et al.* (2014) 'Systemic risk elicitation: Using causal maps to engage stakeholders and build a comprehensive view of risks', *European Journal of Operational Research*, 238(1), pp. 290–299. doi:10.1016/j.ejor.2014.03.035.

Ackermann, F. and Alexander, J. (2016) 'Researching complex projects: Using causal mapping to take a systems perspective', *International Journal of Project Management*, 34(6), pp. 891–901. doi:10.1016/j.ijproman.2016.04.001.

Ackermann, F. and Eden, C. (2005) 'Using Causal Mapping to Support Information Systems Development: Some Considerations', in Narayanan, V.K. and Armstrong, D.J. (eds) *Causal Mapping for Research in Information Technology*. IGI Global, pp. 263–283. doi:10.4018/978-1-59140-396-8.ch011.

Al-Shehab, A., Hughes, R. and Winstanley, G. (2004) 'Using causal mapping methods to identify and analyse risk in information system projects as a post-evaluation process: 11th European Conference on Information Technology Evaluation (ECITE 2004)', *11th European Conference on Information Technology Evaluation (ECITE 2004)* [Preprint].

Al-Shehab, A.J., Hughes, R.T. and Winstanley, G. (2005) 'Facilitating Organisational Learning Through Causal Mapping Techniques in IS/IT Project Risk Management', *Professional Knowledge Management*, p. 145.

Benyon, D. (2014) *Designing interactive systems: a comprehensive guide to HCI, UX and interaction design*. 3. ed. Harlow; Munich: Pearson Education.

Boehm, B.W. (1991) 'Software risk management: principles and practices', *IEEE Software*, 8(1), pp. 32–41. doi:10.1109/52.62930.

Braa, K. and Vidgen, R. (1999) 'Interpretation, intervention, and reduction in the organizational laboratory: a framework for in-context information system research', *Accounting, Management and Information Technologies*, 9(1), pp. 25–47. doi:10.1016/S0959-8022(98)00018-6.

Chaouch, S., Mejri, A. and Ghannouchi, S.A. (2019) 'A framework for risk management in Scrum development process', *Procedia Computer Science*, 164, pp. 187–192. doi:10.1016/j.procs.2019.12.171.

Dingsøyr, T. and Petit, Y. (2021) *Managing layers of risk: Uncertainty in large development programs combining agile software development and traditional project management*, *75-96*. De Gryuter. doi:10.1515/9783110652321-005.

Flyvbjerg, B. (2006) 'Five Misunderstandings About Case-Study Research', *Qualitative Inquiry*, 12(2), pp. 219–245. doi:10.1177/1077800405284363.

Ghobadi, S. and Mathiassen, L. (2014) 'Perceived barriers to effective knowledge sharing in agile software teams', *Information Systems Journal* [Preprint]. doi:10.1111/isj.12053.

Host, M. *et al.* (2012) *Case Study Research in Software Engineering: Guidelines and Examples*. Hoboken, UNITED STATES: John Wiley & Sons, Incorporated. Available at: http://ebookcentral.proquest.com/lib/aalborguniv-ebooks/detail.action?docID=818522 (Accessed: 11 February 2022).

Laukkanen, M. (1994) 'Comparative Cause Mapping of Organizational Cognitions', *Organization Science*, 5(3), pp. 322–343.

Laukkanen, M. (1998) 'Conducting Causal Mapping Research: Opportunities and Challenges', *Managerial and organizational cognition* [Preprint].

Laukkanen, M. and Eriksson, P. (2013) 'New designs and software for cognitive causal mapping', *Qualitative Research in Organizations and Management*, 8(2), pp. 122–147. doi:http://dx.doi.org/10.1108/QROM-08-2011-1003.

Leonhardt Kjærgaard, A. and Blegind Jensen, T. (2014) 'Using Cognitive Mapping to Represent and Share Users' Interpretations of Technology', *Communications of the Association for Information Systems*, 34. doi:10.17705/1CAIS.03457.

Lyytinen, K., Mathiassen, L. and Ropponen, J. (1998) 'Attention Shaping and Software Risk— A Categorical Analysis of Four Classical Risk Management Approaches', *Information Systems Research*, 9(3), pp. 233–255.

McKay, J. and Marshall, P. (2001) 'The dual imperatives of action research', *Information Technology & People*, 14(1), pp. 46–59. doi:10.1108/09593840110384771.

Menezes, J., Gusmão, C. and Moura, H. (2019) 'Risk factors in software development projects: a systematic literature review', *Software Quality Journal*, 27(3), pp. 1149–1174. doi:10.1007/s11219-018-9427-5.

Moran, A. (2014) *Agile risk management*. New York: Springer (Springer briefs in computer science).

Odzaly, E.E., Greer, D. and Stewart, D. (2018) 'Agile risk management using software agents', *Journal of Ambient Intelligence and Humanized Computing*, 9(3), pp. 823–841. doi:10.1007/s12652-017-0488-2.

Ottosen, M.H. (2019) 'Aktionsforskning: Praktisk Anvendelse af Kausalitetskortlægning i Softwareudvikling'. Aalborg Universitet.

Patton, M.Q. (2015) *Qualitative research & evaluation methods: integrating theory and practice*. Fourth edition. Thousand Oaks, California: SAGE Publications, Inc.

Perkusich, M. *et al.* (2021) 'A knowledge-based risk management approach for Scrum software development projects', in Engemann, K.J. and O'Connor, R.V. (eds) *Project Risk Management*. De Gruyter, pp. 97–116. doi:10.1515/9783110652321-006.

Pressman, R.S. and Maxim, B.R. (2015) *Software engineering: a practitioner's approach*. Eighth edition. New York, NY: McGraw-Hill Education.

Schön, E.-M., Radtke, D. and Jordan, C. (2020) 'Improving Risk Management in a Scaled Agile Environment', in Stray, V. et al. (eds) *Agile Processes in Software Engineering and Extreme Programming*. Cham: Springer International Publishing (Lecture Notes in Business Information Processing), pp. 132–141. doi:10.1007/978-3-030-49392-9_9.

Sommerville, I. (2016) *Software engineering*. 10. ed., global ed. Boston Munich: Pearson (Always learning).

Svangren, M. *et al.* (2021) 'Infrastructuring In Digital Transformation: An Action Case Study Of District Heating', *ECIS 2021 Research Papers* [Preprint]. Available at: https://aisel.aisnet.org/ecis2021_rp/29.

Tavares, B.G. *et al.* (2021) 'A Risk Management Tool for Agile Software Development', *Journal of Computer Information Systems*, 61(6), pp. 561–570. doi:10.1080/08874417.2020.1839813.

Tavares, B.G., da Silva, C.E.S. and de Souza, A.D. (2019) 'Practices to Improve Risk Management in Agile Projects', *International Journal of Software Engineering and Knowledge Engineering*, 29(03), pp. 381–399. doi:10.1142/S0218194019500165.

Vidgen, R. and Braa, K. (1997) 'Balancing Interpretation and Intervention in Information System Research: The Action Case Approach', in Lee, A.S., Liebenau, J., and DeGross, J.I. (eds) *Information Systems and Qualitative Research: Proceedings of the IFIP TC8 WG 8.2 International Conference on*

Information Systems and Qualitative Research, 31st May–3rd June 1997, Philadelphia, Pennsylvania, USA. Boston, MA: Springer US, pp. 524–541. doi:10.1007/978-0-387-35309-8_26.

Ward, S. and Chapman, C. (2003) 'Transforming project risk management into project uncertainty management', *International Journal of Project Management*, 21(2), pp. 97–105. doi:10.1016/S0263-7863(01)00080-1.

Williams, T., Ackermann, F. and Eden, C. (2013) 'Project risk : systemicity, cause mapping and a scenario approach', in Artto, K.A. and Kahkonen, K. (eds). London: E&FN Spon, pp. 343–352. Available at: https://strathprints.strath.ac.uk/43728/ (Accessed: 27 February 2022).

Yin, R.K. (2003) *Case study research: design and methods*. 3rd ed. Thousand Oaks, Calif: Sage Publications (Applied social research methods series, v. 5).

Yin, R.K. (2018) *Case study research and applications: design and methods*. Sixth edition. Los Angeles: SAGE.

Appendix 1.

1.1 Case study protocol

1. Overview of the case study

A protocol is proposed by Yin (2018), when a case study is going to be made. The case study protocol is recommended, because it enhances the reliability and replicability of the case study and acts as a valuable tool to guide the data collection process. The protocol is limited to the following sections:

- Section 1: Overview of the case study
- Section 2: Data collection procedures
- Section 3: Case study questions
- Section 4: Outline of the case study report

1.1 Case study objectives

The case study in question is part of my master's thesis with the preliminary title "Agile Risk management: Explaining and managing diverse causes of project risks". The general research question in terms of my thesis is:

How can an agile software development team explain and manage diverse causes of project risks?

The case study will also be working with the following assumptions:

- The team does not talk about previously occurred risks or how to improve the management of them in the future.
- The team does not talk about how to prevent a risk from happening again and to reduce its impact.
- Alpha does not have a similar view on which risks are important.
- Alpha does not have a similar view on how to manage risks.
- Software team has difficulties with explaining and maintaining the causes to their risks.
- Alpha will benefit from causal mapping in terms of explaining and maintaining the causes to their risks.
- The team will benefit from the holistic focus on all the roles' perceptions within an agile software team as it increases the level of representativeness opposed to focusing primarily on project managers.

To enable the researcher to answer the general research question a thorough inspection of relevant literature was conducted as the first step. The themes for the relevant literature include: risk management/agile risk management, causal mapping and the sprint retrospective. The variation in the scope of the relevant literature themes is the result of the researcher's inability to find research with a similar focus in terms of the overall research question.

1.2 Case study issues

The specific case applied to examine the previously mentioned research question is a software development team that works at a large Danish bank in Aalborg. The researcher has been in contact with a representative from Alpha and stated an interest in risk management. The representative and the researcher talked about their general risk management practices and whether they had any issues related to risk management. The representative said that they have issues with learning from previously occurred risks. The issue of learning from previously occurred risks grasped the interest of the researcher and laid the foundation for the overall research question of this case study and the use of causal mapping as a tool to explain and manage risks.

The software development team consists of thirteen members in total and consists of one Product owner, one Scrum master, one Business analyst, two architects, four front-end developers, one back-end developer, two UX designers and one student assistant. Four out of the thirteen team members are located in Poland, whereas the rest of the team operates in Denmark. Alpha makes use of an agile development method, called the Scaled Agile Framework (SAFE). More specific details regarding the organization in question is omitted to favor the organization's wish to remain anonymous.

2 Field procedures

In this case study, three types of data sources are used. The first data source is historical data, specifically documents that reveals information about their risk management practice and sprints. The second is interviews with different role owners of the software development team. The selected roles for the conducted interviews include, 1 PO, 1 SM, 1 BA, 2 architects, 1 front-end developer, 1 back-end developer, 2 UX designers. The researcher will be conducting 10 interviews in total. The third data source is observation of one of their sprint retrospective meetings.

2.1 Archival data

The researcher is in possession of archival data in the form of documents containing documentation of identified risks from recent PI planning meetings and the identified problems they have talked about during Alpha's recent sprint retrospective meetings.

2.2 Interview session

The researcher will be conducting interviews to be able to understand the different role owners' perception of risks and in order to explain and manage the causes to risks. The interviews are estimated to last approximately 30 minutes. The interviews will be semi-structured to enable the researcher to ask follow-up questions, if the need should present itself. The interviews will be structured by 1) introducing the interviewees to myself and the project. 2) Mention the approximate duration and the structure of the interview as being semi-structured. 3) Mention that the interviewee will be anonymous in the case study. Followed by the content of the interview guide which will be presented later in the protocol.

2.3 Observation of sprint retrospective meeting

The researcher has been allowed permission to observe one of the software development team's sprint retrospective meetings. The researcher will not be a participant in the meeting but will be a passive observer and will be taking field notes during the retrospective meeting. The field notes will be taken in accordance with the overall research question and the previously presented hypotheses. The researcher chose to observe one of Alpha's retrospective meetings, because this is where an agile software team reflects on what went wrong and what can be improved in relation to future development. Observation as a method of data collection has been selected as a supplement to the interviews and to achieve a higher level of data triangulation.

3 Case study questions

In the following section the questions guiding the different data sources is introduced.

3.1 Interviews and interview guide

Most of the interviews will be conducted in Danish, except the interview with the back-end developer. This is due to the fact that the back-end developer is located in Poland and does not speak Danish. The interview will therefore be conducted in English. An interview guide was conducted based on ensuring replicability between the questions for the different interviewees and as a structure for the researcher to use during the interviews. The questions that constitute

the interview guide will be presented in Danish. The questions in the interview guide are consistent with the first level within Yin's (2018) five levels of questions for the case study protocol. This is due to the fact that these questions are verbalized to the interviewees. The interview guide may be consolidated below:

3.1.1 Interview guide:

- Hvad er din rolle i teamet?
    - Hvad laver du i teamet?
    - Hvilke ansvarsområder har du i teamet?
- Hvilke risici har du fokus på lige nu med din rolle taget i betragtning? En risiko skal forstås som en potentiel uønsket hændelse, der medfører en negativ konsekvens for jeres projekt (Sommerville, 2016). Hvorfor?
    - Hvis de ikke forstår, så spørg ind til hvad de har fokus på kan gå galt?
- Hvilke risici er opstået under jeres foreløbige projekt?
    - Hvis de ikke forstår, så sig det er det de arbejder på lige foreløbig.
    - Hvorfor opstod de?
- Hvem oplevede konsekvensen af risiciene? Hvorfor oplevede de den konsekvens?
- Hvordan blev de opdaget? Hvorfor blev de opdaget?
- Hvordan analyserede du dem? Hvorfor blev de analyseret på den måde?
    - Nævnes parametre ikke, så spørg ind til disse (f.eks. sandsynlighed og konsekvens).
- Hvordan prioriterede du de risici? Hvorfor prioriterede I de vigtigste risici som I gjorde?
- Hvor ofte har de fundet sted før?
    - Hvis en risiko har fundet sted mange gange → spørg hvorfor denne risiko opstår ofte, gerne i sammenligning med andre risici?
- Hvilken negativ påvirkning havde risiciene for projektet?
    - Hvorfor havde risiciene den negative påvirkning som de havde?
- Hvilken strategi blev taget i brug for enten at undgå eller reducere påvirkningen af de tidligere nævnte risici?
    - Hvorfor blev strategierne valgt?
        - For at reducere sandsynligheden?
        - For at reducere konsekvensens påvirkning?
        - På baggrund af erfaring eller andet?

- Hvordan monitorerede du de vigtigste risici? Hvorfor blev de monitoreret sådan?
    - Hvis de er i tvivl: Hvordan holdte du øje med, at risicienes karakter og tilstand ændrede sig?
- Hvilke udfordringer var der generelt forbundet med at håndtere de risici, der opstod under jeres seneste projekt? Hvorfor? Hvorfor ikke?
    - Forstår de ikke: så spørg ind til de udfordringer de har haft med at håndtere det, der gik galt?
- Nogle sidste kommentarer til det vi har snakket om indtil nu?
- Husk at takke respondenten for deres tid – nævn anonymitet og NDA – kan jeg kontakte dig, hvis der bliver behov for at stille opfølgende spørgsmål?

Interview guide in English

- What is your role in your team?
    - What do you do in the team?
    - Which areas are you responsible for?
- Which risks do you currently focus on considering your role in the team? A risk is to be understood as a potential undesired event that has a negative impact on your project.
    - Why do you focus on these risks?
- Which risks have occurred during your current project?
    - Why did they occur?
- Who experienced the consequences of the risks, when they occurred during your current project?
    - Why did they experience it?
- How were the risks discovered in your team?
    - Why were they discovered?
- How did you analyze the risks?
    - Why were they analyzed in that way?
- How did you prioritize the risks?
    - Why were they prioritized in the way they were?
- How often have the risks occurred before in the past?
    - If a risk has occurred often in the past, ask why that risk has occurred often
- Which consequences did the risks produce in your current project?
    - Why did the risks produce the consequences they did?
- Which strategy did you use to avoid or reduce the impact of the risks?

    - Why did you choose the strategy you did?
- How did you monitor the risks?
    - Why were they monitored in the way they were?
- Which difficulties did you experience in general, when managing risks during your current project?
    - Why did you experience these difficulties? Why not?
- Some last remarks or something we did not cover?
- Can I contact you if further questions should come up?

The main goal of the interviews is to map the causal relationships between risks and the role owner's perception as to why the risk occurred and why it was managed in the way it was. The overall structure of the interviews and phrasing of the interview questions are inspired from Patton (2015). The logic behind the questions is derived from the causal mapping literature, Laukkanen (1998) and Ackermann & Eden (2005) Ghobadi & Matthiessen (2014). The questions' focus on specific risk management areas stems from Sommerville (2010), Moran (2014) and Boehm (1991).

Given that analyzing causal relationships is the main goal of the data analysis, "why" questions were included as a necessity in the interview guide, even though Patton (2015) generally recommends omitting "why" questions, unless the researcher's goal is to deduce causal relationships (Patton, 2015, 454-455).

3.2 Individual feedback

Individual feedback guide

- Tak, fordi du vil tage dig tid til at snakke med mig igen.
- Jeg har i min dataanalyse kortlagt de risici som du har nævnt under interviewet. Risiciene er forbundet med pile, der repræsenterer en kausalsammenhæng imellem risiciene. Det vil sige, at kortet ikke kun giver et overblik over de nævnte risici, men også om risiciene bidrager til andre risicis opståen.
- Del skærmen og vis kortet.
- Forklar kortet, rationalet bag risiciene og deres sammenhæng.
- Nu vil jeg stille dig nogle spørgsmål til kortet og jeg starter optagelsen.
- Er der noget du er i tvivl om eller noget du godt kunne tænke dig at få uddybet?
- Forklarer risiciene i kortet dine bekymringer?
    - Er der nogle risici, der mangler i kortet (f.eks. de løse risici)?

    - Er der nogle risici, der ikke burde være der (f.eks. fordi det ikke er noget, der er værd at have fokus på)?
- Hvad er de mest betydningsfulde risici for dig?
    - Er du kommet i tanke om nogle nye vigtige risici siden sidst vi talte sammen?
- Synes du, at forbindelserne/pilene forklarer sammenhængen mellem risiciene i kortet?
    - Er der nogle forbindelser/pile, der mangler?
    - Er der nogle forbindelser/pile, der bør blive fjernet?
- Nogen sidste kommentarer eller spørgsmål til kortet?
- Tak for din tid og dine svar. Dine svar og identitet vil ligesom sidst blive anonymiseret.

Appendix 2.

2.1 Overview of data collection and analysis

Table 1: Data collection methods and items

Interview	Observation	Archival data	Individual feedback	Workshop
Interview with PO	Sprint retrospective meeting	Documentation of mentioned issues from 10 sprint retrospective meetings	Feedback from UX	Presentation of results followed by collective feedback (sprint retrospective)
Interview with SM	Initial PI-planning meeting (morning)	Documentation of team risks from three PI-planning meetings	Feedback from ARC	
Interview with BA	Final PI-planning meeting (afternoon)		Feedback from DEV	
Interview with UX1			Feedback from BA	
Interview with UX2			Feedback from PO	
Interview with DEV1			Feedback from SM	
Interview with DEV2				
Interview with DEV3				
Interview with ARC1				
Interview with ARC2				

Table 2: Identified risks from interviews

Risk	Number of occurrences	Mentioned by	Coding category
1. Inability to comply with the deadline	7	PO, BA, ARC2, DEV1, DEV2, DEV3 and UX1	Task
2. Other teams are unable to comply with their deadlines	6	DEV2, UX1, ARC2, PO and DEV3	Structure
3. Team member is sick/absent	6	BA, PO, DEV2, UX1, ARC2 and DEV3	Actor
4. Inaccurate estimation of task	5	DEV2, DEV1, DEV3, BA and UX1	Task

5. Pressure from management	4	PO, BA, DEV2 and UX1	Structure
6. Complying with new regulations	4	BA, DEV2, UX1 and ARC2	Structure
7. High rotation among developers	4	UX1, ARK2, DEV2, ARC1	Actor
8. Key developer is absent	4	BA, UX1, DEV1 and ARC2	Actor
9. Insufficient communication across locations	3	BA, SM and DEV1	Structure
10. Inexperienced team members	3	DEV2, UX1 and ARC2	Actor
11. Lack of information on tasks	3	DEV1, BA and ARC2	Task
12. Inadequate maintenance of open-source library	3	DEV1, ARC1 and ARC2	Technology
13. Developing insufficient code	2	DEV3 and DEV1	Actor
14. Increased level of technical debt	2	PO, DEV1	Structure
15. Developing insufficient design	2	UX1 and UX2	Actor
16. Prolonged decision-making process	1	SM	Structure
17. Insufficient coordination	1	SM	Structure
18. Team size has become too big	1	SM	Structure
19. Insufficient balance of finished and unfinished tasks	1	UX1	Structure
20. Overflow of tasks	1	SM	Structure
21. Lacking clarification from other BA	1	BA	Structure
22. Sudden occurrence of technical issue	1	UX1	Technology
23. Difficulty in creating relations across locations	1	SM	Structure

Table 3: Identified program risks from PI: Planning-meeting

Program (6 teams in total) risks (affects everyone)	Owner	Resolve	Mitigate
Low bus factor on front-end developers (lack of experienced developers and flexibility when developers are absent)			
Minimal progress on risks mentioned in pfp7 (security assessment)		Accepted	
Front end devs in Poland are leaving			
NRP technology stack is becoming old (concerning back-end and efficiency)		Accepted	
Windows laptop support is not good enough for a fully managed installation (installation takes too long, no computer = no work)		Accepted	

Table 4: Identified team risks from PI: Planning-meeting

Team risks (impact only on team)	Owner	Mitigated	Resolved

Training new developers		Training new developers (only one person training)	
Slow recruitment/not started			
Lack of time for maintenance			
Half amount of time for BA (assigned two teams)			
Emilie key employee (extremely important)			
Cloud team issue		Cloud team issue (have not send documentation way over due and cannot work on the feature in	

Table 5: Identified risks after individual feedback session

Risk	Mentioned by	Coding category
1. Inability to comply with the deadline	PO, BA, ARC2, DEV1, DEV2, DEV3 and UX1	Task
2. Other teams are unable to comply with their deadlines	DEV2, UX1, ARC2, PO and DEV3	Structure
3. Team member becomes sick/absent	BA, PO, DEV2, UX1, ARC2 and DEV3	Actor
4. Inaccurate estimation of task	DEV2, DEV1, DEV3, BA and UX1	Task
5. High rotation among developers	UX1, ARK2, DEV2, ARC1 and SM	Actor
6. Complying with regulations	BA, DEV2, UX1 and ARC2	Structure
7. Pressure from management	PO, BA, DEV2 and UX1	Structure
8. Key developer becomes absent	BA, UX1, DEV1, SM and ARC2	Actor
9. Insufficient communication across locations	BA, SM and DEV1	Structure
10. Inexperienced team members	DEV2, UX1 and ARC2	Actor
11. Lack of information on tasks	DEV1, BA and ARC2	Task
12. Inadequate maintenance of open-source library	DEV1, ARC1 and ARC2	Technology
13. Developing insufficient code	DEV3 and DEV1	Actor
14. Increased level of technical debt	PO, DEV1	Technology
15. Developing insufficient design	UX1 and UX2	Actor
16. Compromise on amount of tasks/quality	SM and BA	Task
17. Prolonged decision-making process	SM	Structure
18. Insufficient coordination	SM	Structure
19. Team size becomes too big	SM	Structure
20. Insufficient balance of finished/unfinished tasks	UX1	Actor
21. Overflow of tasks	SM	Task
22. Lacking clarification from other BA	BA	Structure
23. Difficulty in creating relations across locations	SM	Structure
24. Lacking clarification from enterprise architects	UX1	Structure
25. Change in prioritization of tasks	UX1	Task

26. Lack of sufficient time for design	UX1	Task
27. Inadequate development platform	ARC1	Technology
28. API review process is time consuming	ARC1	Technology
29. Outdated technology stack	ARC1	Technology
30. Lack of acceptance and management of risks by management	BA	Structure
31. Unanticipated complexity in development	PO, Dev	Task

Table 6: Overview of the number of risks related to each category from the leavittian model

Category	Number of risks
Structure	11
Task	8
Actor	7
Technology	5

Table 7: Overview of the number of risks causes from each category

Category	Number of risk causes
Actor	48
Structure	31
Task	13
Technology	6

2.2 Leavittian models

2.2.1 Developer

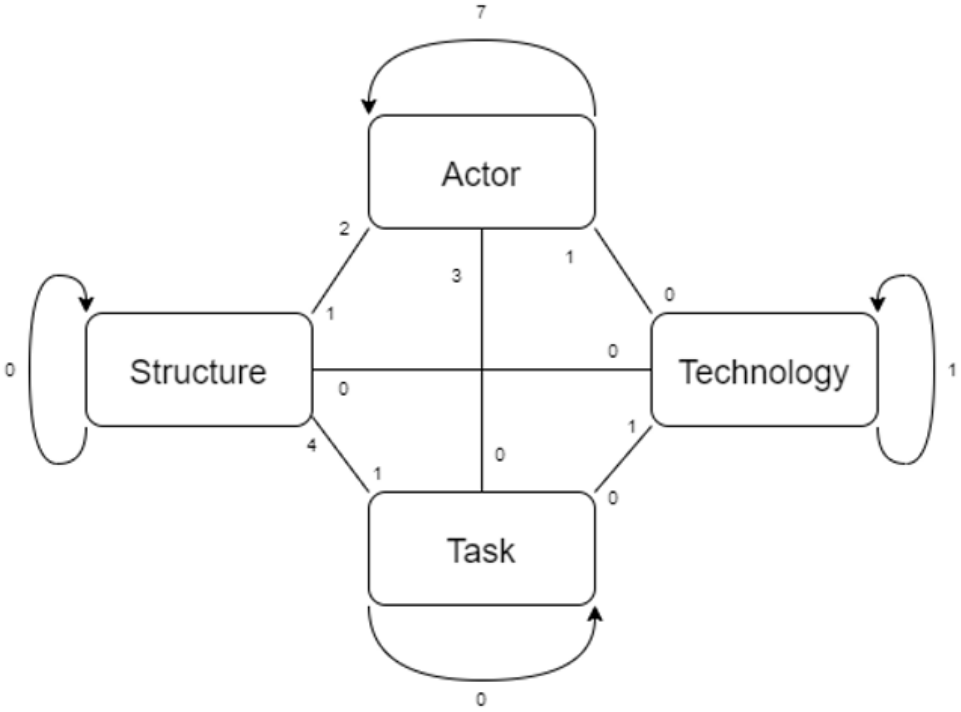

Figure 9: Developer's leavittian model for categorization of risks

2.2.2 Architect

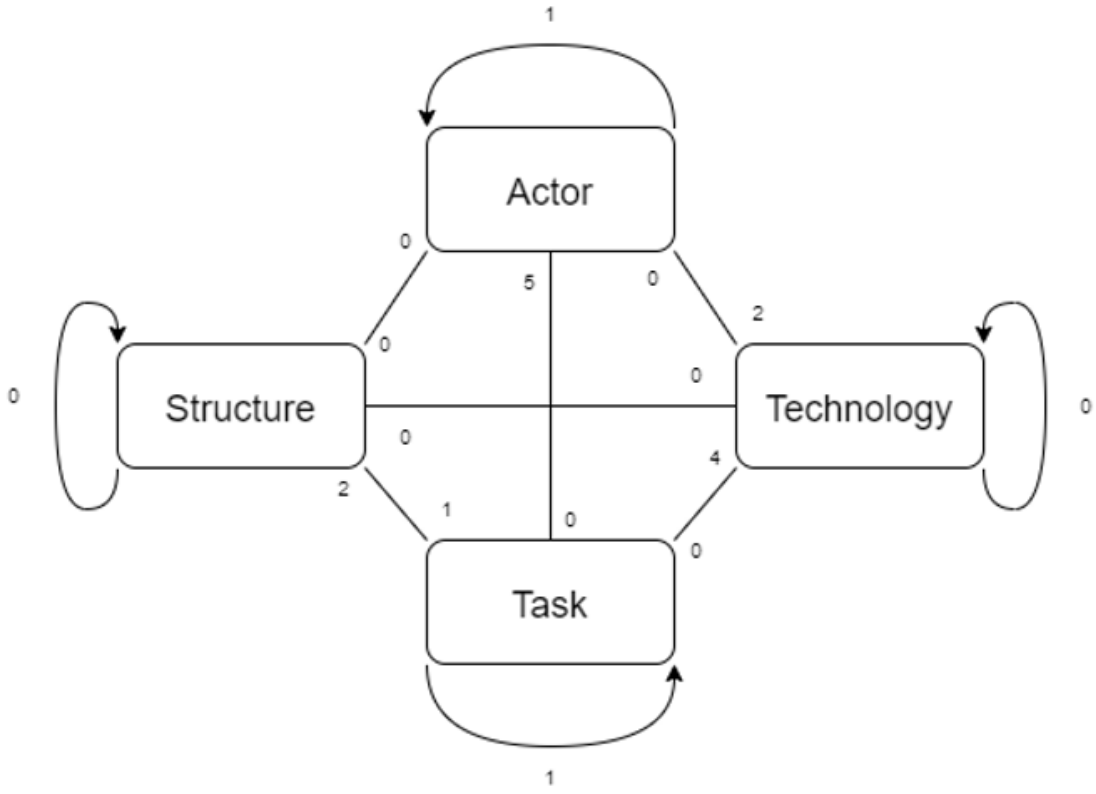

Figure 10: Architect's leavittian mode for categorization of risks

2.2.3 UX-designer

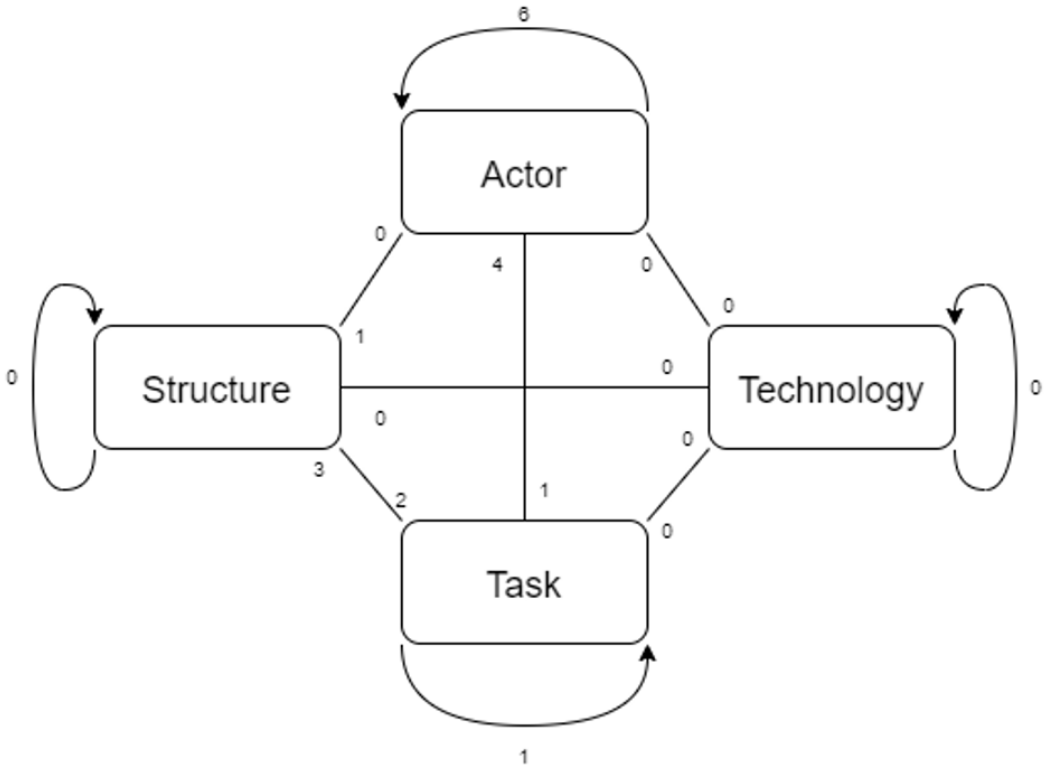

Figure 11: UX-designer's leavittian model for categorization of risks

2.2.4 Business Analyst

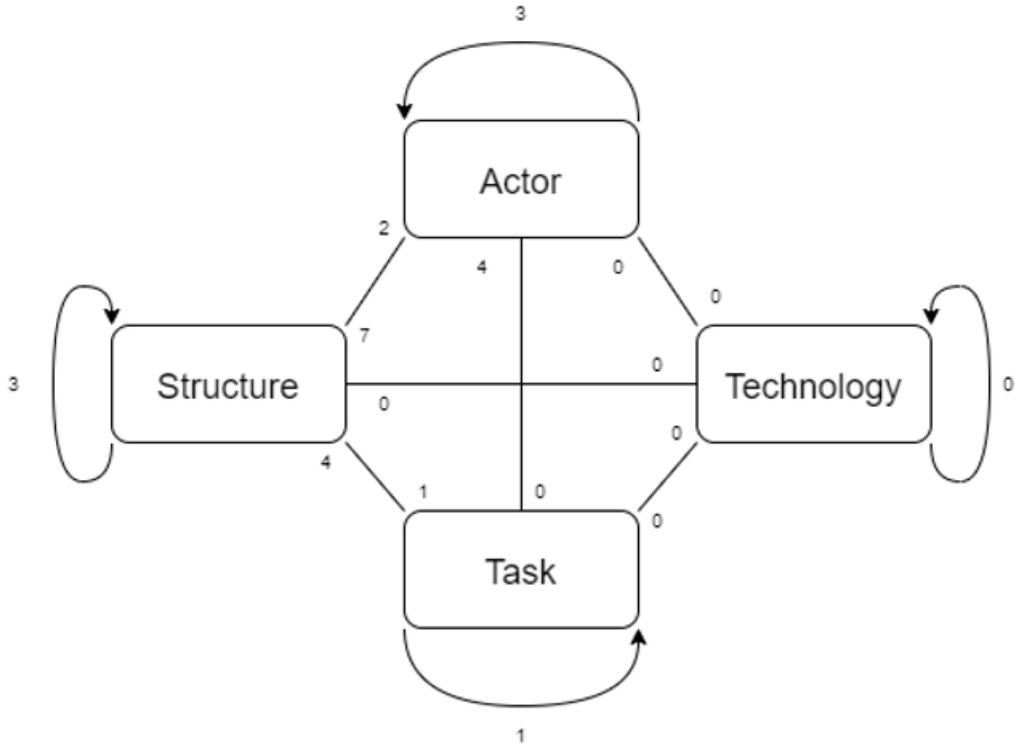

Figure 12: Business analyst's leavittian model for categorization of risks

2.2.5 Product Owner

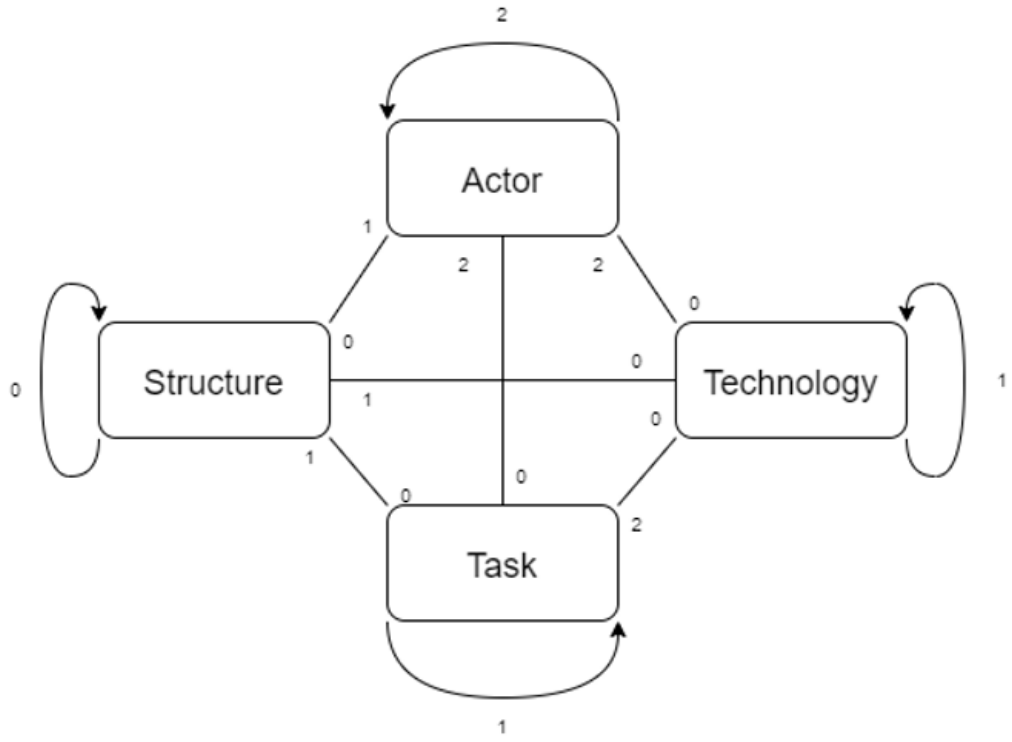

Figure 13: Product Owner's leavittian model for categorization of risks

2.2.6 Scrum Master

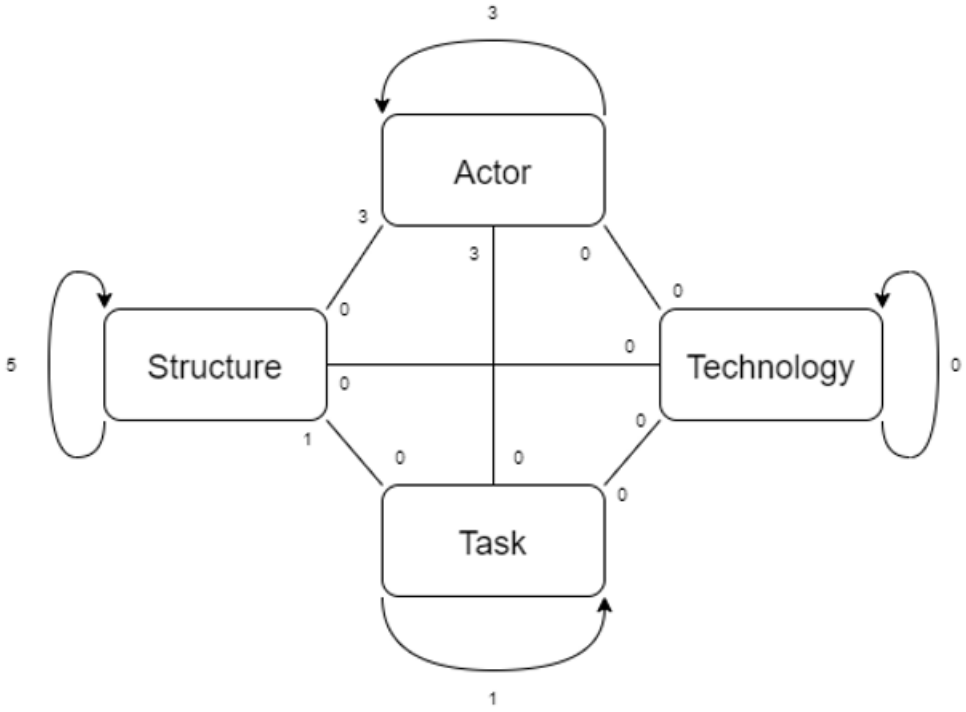

Figure 14: Scrum Master's leavittian model for categorization of risks