Author version. Final version to be available through https://www.sciencedirect.com/

Causal Mapping of the Risks of Using Generative AI in Software Development

David Kinnberg Hein
Aalborg University, Department of Computer Science, Selma Lagerlöfs vej 300, 9000, Aalborg, Denmark
davidkh@cs.aau.dk (corresponding author)

John Stouby Persson
Aalborg University, Department of Computer Science, Selma Lagerlöfs vej 300, 9000, Aalborg, Denmark
john@cs.aau.dk

Victor Vadmand Jensen
Aarhus University, Department of Clinical Medicine, Jens Chr. Skous Vej 4, 8000, Aarhus, Denmark & University Clinic for Interdisciplinary Orthopaedic Pathways, Regional Hospital Silkeborg, Falkevej 1P, 8600, Silkeborg, Denmark
vvj@clin.au.dk

Anders Rysholt Bruun
Aalborg University, Department of Computer Science, Selma Lagerlöfs vej 300, 9000, Aalborg, Denmark
bruun@cs.aau.dk

Martin Gilje Jaatun
SINTEF Digital, Software Engineering, Safety and Security, Strindvegen 4, 7034 Trondheim, Norway
martin.g.jaatun@sintef.no

Abstract

Context:

Generative AI tools can enhance the productivity and effectiveness of software development. These tools are evolving rapidly, as are the associated risks. Therefore, software organizations adopting these tools need to understand their risks, why they occur, and how they evolve over time.

Objective:

While prior research has identified risks associated with generative AI in software development, limited attention has been given to how these risks are causally interconnected and how organizations explain and revise them over time. This study addresses this gap by examining how software organizations explain the causal relationships between generative AI risks and how these explanations evolve following initial adoption.

Methods:

In a multi-case study of three Danish software organizations in the early stages of generative AI tool adoption, we interviewed software developers and managers to uncover causal explanations for the risks associated with generative AI. We employed a causal mapping approach to understand and visualize the differences and similarities across software organizations' causal explanations of risks. A year later, we validated the risks and causal maps with representatives from each software organization.

Results:

Our study shows that the software organizations exhibit circular reasoning regarding a perpetual uncertainty in the validity of AI-generated code, with the three risks: *Mistrust of AI, Insufficient validation of AI output*, and *Insufficient/insecure AI output.* They are also concerned about similar root risks, which solely cause other risks without being caused by any risks themselves, such as *Lacking AI competencies*. While they differed in emphasis on tail-end risks, which are understood as not causing any additional risks.

Conclusion:

The causal mapping approach proved appropriate for illustrating similarities and differences in software organizations' perceptions of risks and causal explanations regarding the use of generative AI in software development. The same applies to visualizing how the emphasis on risks can change over time.

Keywords:

Risk Management, Software Development, Generative AI, Causal Mapping, Multi-case Study.

1 Introduction

Recent developments in artificial intelligence (AI) have led to the proliferation of generative AI tools such as ChatGPT and GitHub Copilot. Generative AI tools are trained on large amounts of data, such as text or images, and can use this training data to generate new content based on users' prompts (Ebert and Louridas, 2023). With prospects of increased productivity and time savings (Imai, 2022; Peng et al., 2023), software organizations have now adopted generative AI in software development. However, the introduction of AI into software engineering is not entirely new. In 2012, Harman argued that software engineers were becoming increasingly interested in AI due to the growing complexity and interconnectedness of software systems, and a desire to optimize the processes and products that underlie such systems (Harman, 2012). As such, traditional AI techniques, like machine learning (ML) and deep learning (DL), have been applied to many areas of software engineering. A tertiary study of ML for software engineering (ML4SE) identified 11 knowledge areas in the ML4SE literature, with the five most frequently addressed being software quality, testing, processes, management, and requirements (Kotti et al., 2023). Similarly, a systematic literature review found that ML and DL had been applied in over half of the reviewed studies to analyze defects or support the maintenance and evolution of software (Wang et al., 2023). However, researchers have also highlighted the possible risks of AI in software engineering. (Feldt et al., 2018) argued that AI becomes riskier the closer it is applied to software's runtime due to costlier consequences. Furthermore, end-users, experts, and other stakeholders may experience different negative impacts from AI use (Lockey et al., 2021), underscoring the variety of potential risks. For example, using ML for software bug reports risks coordinators losing the ability to diagnose and manage bugs (Borg et al., 2022).

The adoption of generative AI in software development has similarly introduced a range of risks for software organizations, including inaccurate AI output, a lack of trust and control (Sergeyuk et al., 2025), and inappropriateness for more complex tasks (Tian et al., 2023). Consequently, researchers have emphasized the need to raise awareness among software engineers regarding the risks of using generative AI (Nguyen et al., 2023) and to provide guidance for mitigating these risks (Russo, 2024). However, to effectively address the risks posed by generative AI in software development, we first need to understand its underlying causes. Attempts have been made to understand the root causes of the risks in software development qualitatively (Bjarnason et al., 2012; Yu et al., 2015) and quantitatively (Chatzipoulidis et al., 2015; Rique et al., 2025). However, generally, qualitative research on machine learning practices remains underexplored (Ardic et al., 2025). Software organizations are increasingly adopting generative AI tools in software development, yet they often struggle to manage expectations regarding their capabilities and implications. Prior work has shown that organizations hold diverse and evolving expectations about how AI tools will impact development practices, many of which change or are not sustained once the tools are used in practice (Jensen et al., 2025). This highlights a broader

challenge in which organizations not only face uncertainty about the benefits of generative AI but also about the risks and how these relate to their existing practices. Understanding how such risks are perceived, connected, and revised over time is therefore essential when integrating generative AI into software development. Against this backdrop, we will address the following research question:

- How do software organizations explain the risks associated with using generative AI in software development?

To answer this question, we conducted an in-depth multi-case study of the causal reasoning for the risks associated with using generative AI in three Danish software organizations. A risk, in itself, can be understood as an entity comprising an explanatory causal relationship between a "risk object" that poses the threat and an "object at risk" that represents the value being threatened (Boholm and Corvellec, 2011). Our study aims to expand on the systemic view of risks as interconnected; when one risk occurs, it can lead to the occurrence of another (Ackermann et al., 2014; Carlo et al., 2008). To unfold how software organizations see causal connections between the risks of using generative AI, we apply the causal mapping approach. Causal mapping has been used in previous research to understand and visualize causal explanations of project risks (Hein et al., 2025), barriers to knowledge sharing (Ghobadi and Mathiassen, 2014), and technology adoption (Kjærgaard and Jensen, 2014). This case study extends this line of research regarding the risks associated with using generative AI in software development. Nevertheless, the fast technological evolution of generative AI (Acharya et al., 2025; Carleton et al., 2024) also shapes software organizations' explanations of the associated risks. Concurrently, existing research primarily considers generative AI risks in isolation (Derner et al., 2024; Sergeyuk et al., 2025), offering limited insight into how these risks are causally related or how organizations reason about their emergence and evolution over time. Against this backdrop, we returned to the three software organizations after they had employed generative AI for a year to assess whether their explanations of risks had shifted in emphasis, to address the second research question:

- How do the explanations of risks associated with using generative AI in software development evolve after software organizations' initial adoption?

The paper detailing our response to these two research questions is organized as follows: First, we review the current research on generative AI and risk management in software development. Second, we introduce and justify the study's research approach, followed by an outline of our findings and causal maps. Finally, we discuss the paper's contributions to research, implications for practice, threats to validity, and future research directions.

2 Related Work

2.1 Generative AI in Software Development

Generative AI developments in recent years have led to this technology being widely considered for professional software engineering. Researchers have highlighted the breadth of AI applications in software engineering across a wide range of activities (Ebert et al., 2024; Tsintzira et al., 2020). As potential AI pair programmers (Bird et al., 2022; Imai, 2022; Moradi Dakhel et al., 2023) in software development, AI-generated code is poised to significantly influence the overall quality of the software project. A survey study found that software engineers perceived GitHub Copilot-generated code as shorter and more accurate, indicating higher quality (Zhang et al., 2023). Similarly, using GitHub Copilot may lead software engineers to recognize that suggestions from generative AI tools are distinctive and may include elements that might otherwise be overlooked (Bird et al., 2022). In cases of quality issues in AI-generated code, it has been shown that ChatGPT can utilize static analysis outputs to correct some of the code itself (Liu et al., 2024). Furthermore, generating code with GitHub Copilot can result in fewer vulnerabilities than human-written code (Asare et al., 2023), thus limiting security issues that attackers can exploit. Researchers have also suggested that user experience (UX) designers may employ generative AI to identify usability issues in their designs, thereby making applications more straightforward to use (Stige et al., 2023). Researchers have underscored the potential of using generative AI to enhance the experiences within the software engineering process, typically by

automating mundane tasks (Ernst and Bavota, 2022; Jaworski and Piotrkowski, 2023; Knearem et al., 2023) and allowing more time for the more creative aspects of software engineering (Bird et al., 2022). Additionally, researchers contend that software engineers may find it easier to learn and improve their programming skills without frequent context switching (Ernst and Bavota, 2022).

However, while researchers have highlighted generative AI's potential to improve code and productivity (Ebert and Louridas, 2023), there is also research suggesting challenges in using generative AI to improve software. (Weber et al., 2024) did not find quality differences in the code generated with either ChatGPT or GitHub Copilot, and they did not decrease code quality compared to code produced by a human. (Moradi Dakhel et al., 2023) demonstrated that buggy solutions from GitHub Copilot need to be managed by expert software engineers rather than inexperienced engineers, which is corroborated by other findings that human supervision of GitHub Copilot increases code quality (Rasnayaka et al., 2024; Yetistiren et al., 2022). Generative AI's potential here is also challenged by the fact that "ChatGPT-generated code is prone to various code quality issues, including compilation and runtime errors, wrong outputs, and maintainability problems" [p. 21] (Liu et al., 2024). In continuation, recent studies have found that AI-generated code does not create the same vulnerabilities (Asare et al., 2023) or types of bugs as human-written code (Tambon et al., 2025). At the same time, various research efforts (Kou et al., 2024; Li et al., 2024) indicate that generative AI and human developers may prioritize different aspects of the prompt when addressing coding challenges. Further, software engineers may exhibit uncertainty toward the copyright of AI-generated code (Bird et al., 2022), rely too eagerly on generative AI without understanding its limitations (Cheng et al., 2023), or have security concerns about generative AI (Jaworski and Piotrkowski, 2023).

Moreover, concerns have been raised about leveraging generative AI in the software engineering process. For instance, UX designers may be apprehensive about AI in their design tools, fearing the perpetuation of existing biases in their design work (Knearem et al., 2023). Others have pointed out the risk that generative AI might lead to negative experiences in the software engineering process, such as overreliance on its outputs (Bird et al., 2022). Furthermore, software engineers may struggle to grasp AI-generated code fully (Bird et al., 2022). This challenge is compounded by the necessity for existing software engineering knowledge to use generative AI effectively for coding (Rasnayaka et al., 2024; Zhang et al., 2023). The literature emphasizes both the opportunities presented by generative AI in software engineering and the associated risks of utilizing these tools. However, these risks related to generative AI are usually considered in isolation. (Nguyen et al., 2024) described how software engineering management must critically assess the role of generative AI to prevent employees and organizations from facing the consequences of risks tied to this technology. Therefore, understanding how the risks of generative AI are interconnected is essential for effective risk management to address the appropriate risks and prevent risks from cascading (Alpcan and Bambos, 2009) and resulting in multiple coinciding issues. Especially as these generative AI tools are evolving at a tremendous rate (Carleton et al., 2024), as evidenced by the recent introduction of Agentic AI (Acharya et al., 2025).

Recent studies, including Zhou et al. (2025), have examined the causes of Copilot usage problems. They found that the most frequent causes are "Copilot Internal Error," "Network Connection Error," and "IDE Compatibility Issue" (Zhou et al., 2025). Other researchers examined why software engineers choose to use generative AI. It was discovered that the compatibility of generative AI with existing development workflows and organizational values is the primary driver of its adoption. Although these generative AI tools may align with organizations' requirements, it remains essential for organizations to recognize the potential risks associated with generative AI use (Russo, 2024). Sergeyuk et al. (2025) found that the most frequent reasons for developers to avoid using generative AI are "Lack of need for AI assistance", "AI-generated output is inaccurate," "Users' lack of trust and the desire to feel in control" (Sergeyuk et al., 2025). Despite attempts to investigate the causes of generative AI adoption, the literature on the risks and challenges associated with its use in software development remains scarce. While prior research has identified risks associated with generative AI in software development and explored aspects of risk management, these risks are typically considered in isolation. Limited attention has been given to how such risks are causally interconnected and how software organizations explain and revise them over time. To address this gap, we examine how software organizations explain the risks

associated with using generative AI in software development and how these explanations evolve following initial adoption.

2.2 Risk Management in Software Development

Risk management in software engineering has existed for many years and can be traced back to the introduction of risk management approaches by the prominent figure, Barry Boehm (Boehm, 1989, 1991). Traditionally, Boehm defines a risk as “the possibility of loss” [pp. 33] (Boehm, 1991), while others merely view it as an undesirable future event (Pressman and Maxim, 2015; Sommerville, 2016). Simplistically, a risk can be viewed as the sum of the likelihood that the event will occur and the consequences of the event (Pfleeger, 2000; Pressman and Maxim, 2015). There still seems to be some ambiguity regarding the definition of risk. In recent years, attempts have been made to unify risk management concepts, presenting terminology to address the aforementioned ambiguity and inconsistencies surrounding the definition of risk (Abioye et al., 2020; Masso et al., 2022).

The argument for integrating risk management in software development has traditionally been to combat high software project failure rates and cost overruns, leading to a cascade of frameworks for identifying and addressing project risks (Fairley, 1994; Schmidt et al., 1999; Williams et al., 1997). The issue of project failures and cost overruns is still prevalent in modern-day software projects (Flyvbjerg et al., 2022; Kula et al., 2022). Kula et al. (2022) noted that factors like refining requirements, task dependencies, organizational alignment, and politics are seen as significantly influencing on-time delivery. In contrast, proxy measures including project size, the number of dependencies, past delivery performance, and team familiarity can account for a substantial portion of scheduling deviations (Kula et al., 2022).

Furthermore, the rise of agile methodologies (Beck et al., 2001) has led the risk management literature to focus on risk management within agile development, examining risks specific to software projects that follow agile principles. Several frameworks have been created to integrate risk management into the iterative nature of agile development (Lopes et al., 2021; Muntés-Mulero et al., 2019; Odzaly et al., 2018; Tavares et al., 2019), and frameworks tailored for addressing the challenges of distributed agile development (Adzgauskaite et al., 2025; Persson and Schlichter, 2015; Shrivastava and Rathod, 2017). Although research suggests that adopting risk management activities is an adequate way to reconcile traditional waterfall software development with agile development (Boehm and Turner, 2005). The agile risk management frameworks face the issue of conflicting with a core value within the agile manifesto, “individuals and interactions over processes and tools,” (Beck et al., 2001) as adopting risk management activities introduces new processes and tools for an agile team. Agile software teams face new challenges in implementing risk management practices, alongside noticeable distinctions compared to the key risks traditionally identified by Boehm (1991), which include "personnel shortfalls,” “unrealistic schedules and budgets,” and “developing the wrong functions and user interface." In contrast, the primary risks in agile projects are recognized as “Technical debt,” “Lack of knowledge retention,” and “Separation of development and IT operations” (Elbanna and Sarker, 2016). Recent research has also revealed risks associated with data management in agile software development, including challenges in managing data integration, gathering various data types, automating data collection, and satisfying real-time analysis demands. To mitigate these risks, they suggest strategies such as adopting appropriate automation tools, decentralized data management practices, and ontology-based methods (Fawzy et al., 2025).

Clearly, the risk management literature is not scarce in proposing risk management frameworks. However, in the field of software engineering, little attention has been devoted to exploring the concept of risk in greater detail, particularly its causality. Apart from a few qualitative studies on root causes of risks (Bjarnason et al., 2012; Yu et al., 2015), and the abundance of statistical studies predicting the causes of risks (Brændeland et al., 2010; Chatzipoulidis et al., 2015; Rique et al., 2025). This suggests an inherent objective view on risk in the software engineering field. Beyond the realm of software engineering, Boholm and Corvellec (2011) describe risk as a concept shaped by cultural and social contexts, representing a harmful possibility for an actor. This concept comprises an interrelated "risk object" and an "object at risk." The risk object signifies the potential danger, whereas the object at risk

denotes the value in jeopardy. The notion of risk does not arise by itself; instead, it is formed through a semantic connection between these objects. The identities of the risk object, the object at risk, and their interrelation are established only through their connections. A risk object is meaningful only in relation to an object at risk, which, in turn, is identified by its link to a risk object (Boholm and Corvellec, 2011).

In 1998, Lyytinen et al. (1998) introduced an attention-shaping framework for software risks, grounded in Leavitt's (1964) socio-technical model. They propose that software risks are part of a system comprising four interconnected elements: actor, technology, task, and structure. A change in any of these components or their relationships can result in changes to the others and the software development process (Lyytinen et al., 1998). Therefore, software risks in development can be traced back to any of these four elements. In continuation, the idea of systemic risk has been introduced to overcome the complexities and challenges associated with risk emergence, emphasizing that systemic risks are interconnected and that when one part fails, it can cause failures in another (Carlo et al., 2008). In closing, extant research has explored the causes of project risks by employing the concept of systemic risks and the causal mapping technique to illustrate the systemic network (Hein et al., 2025). They showed that risks in software projects within agile teams are interconnected, suggesting that causal mapping is suitable for treating these risks collectively instead of individually. This has similarly been found to be the case for barriers to knowledge sharing in software projects (Ghobadi and Mathiassen, 2014) and visualizing the causality of perceptions of adopting a new technology (Kjærgaard and Jensen, 2014). Based on the effective use of causal mapping in the risk management literature, we adopt this approach to address the gap in research on understanding risk causes in the risk management literature.

3 Research Approach

We adopted a multi-case study approach (Runeson et al., 2012), collaborating with three Danish software companies to uncover the risks and causes of using generative AI in software development. This procedure provided both richness and depth in comprehending each software organization’s risks (Runeson and Höst, 2009; Yin, 2018) while allowing for comparisons among them (Eisenhardt and Ott, 2017). The software organizations in this study were chosen based on their early stage of adoption of generative AI tools to examine the evolution of the risks of using generative AI a year after our initial investigation. To allow for comparison among the case organizations, they were selected to reflect variation in their contexts (Eisenhardt and Graebner, 2007; Flyvbjerg, 2006) across industry, size, customer segment, and software development process. This enabled us to investigate the causes of risks across various types of software organizations and to compare the causes of the identified risks based on the specific context of each organization. We employed the causal mapping technique (Laukkanen, 1998) to visualize the causes of risks and facilitate more straightforward comparison of risks and their associated causes among software organizations (Laukkanen, 1994). All the researchers work for a university and have no proximity to the case companies we collaborated with. None of the researchers work for the case companies, and neither the researchers nor the case companies solely dictated the end goal of this study; instead, it was developed in unison. We anonymized the collected data, signed an NDA with the companies, and sent the completed article to one representative from each case.

3.1 The Cases

The Bankers is a leading Danish bank with a workforce exceeding 4,000 employees, offering a variety of banking and housing investment products and services. They focus on providing mortgage credit and banking operations. The *Bank* is owned by a Danish association of homeowners and corporations. It employs around 400 staff members dedicated to developing software used both internally and by other banks that provide mortgage products. Among these, 100 employees are part of a multi-team effort to create a mortgage platform. The teams work together within the Scaled Agile Framework (SAFe), collaborating towards shared objectives and solutions. This is accomplished through the Agile Release Train, which enables continuous value delivery. The process adheres to a fixed and consistent schedule guided by the program increment rhythm. Each program increment lasts 10 to 12 weeks, with teams initiating a new system increment every two weeks. All teams are synchronized within the same program increment, with established start and end dates. A key event in SAFe is the Program Increment

Planning session, where teams align on the business context and vision, prioritize essential development areas, and identify risks that could impact the program increment's success. In addition to the program planning session, the *Bankers'* software teams use the Scrum event known as the "sprint retrospective" every second week to discuss past challenges and future risks in the software project and identify solutions.

The Insurers is a Danish non-life insurance company recognized for providing affordable, customer-focused insurance solutions. Established over 100 years ago, it employs more than 500 people. The company has a long history of offering various insurance products, including auto, homeowners, accident, and travel insurance. The company was initially owned by Danish trade unions, with a strong emphasis on ensuring fair and accessible insurance for union members. They have recently been acquired by one of the largest insurance providers in Scandinavia, yet they continue to operate under their own brand. They are noted for their competitive pricing, efficient digital services, and high customer satisfaction ratings in the Danish insurance market. The *Insurers'* software development department follows the agile methodology "Scrum," enabling iterative development, flexibility, and continuous feedback loops to enhance internal systems and digital services. By implementing Scrum, the Insurers facilitate collaboration across development teams, accelerate software delivery, and align their digital solutions with evolving customer and business needs. The *Insurers'* software teams follow the typical Scrum activities outlined in the Scrum Guide, including daily scrums, sprint retrospectives, and sprint reviews.

The Automaters is a software vendor focused on logistics visibility and automation. Established in the early 1950s, the company expanded its automation solutions from Denmark to a global clientele in the 2000s, opening offices in Canada and Romania and employing over 200 staff. Approximately 100 of these are directly involved in software development. Utilizing RFID (radio-frequency identification), it tracks complex assets in both private and public sectors. Its clients range from healthcare providers to airports, airlines, postal services, and libraries. RFID technology uses radio waves to identify and track items via tags that store data, which are read by an RFID antenna. The company employs a homegrown development lifecycle that primarily follows plan-driven principles for its automation solutions. At the *Automaters*, software development is divided among specialized teams, each focused on different software architecture components, such as a team dedicated to Internet of Things (IoT) devices to build expertise in the field. The sales team plays a vital role early in the development process, collecting client requirements and facilitating approval. The required approvals are then forwarded to the software development teams for execution. The organization adheres to a plan-driven development model rooted in traditional waterfall methodology, emphasizing structured development stages with clear checkpoints. The development process ensures systematic progress and thorough validation throughout.

3.2 Data Collection and Analysis

As shown in Table 1, we initiated our data collection by conducting interviews with a wide range of software practitioners across our three case companies to identify the risks associated with using generative AI in software development. We conducted 15 initial semi-structured interviews, each lasting between 32 and 67 minutes. The interviews were all recorded and transcribed with Whisper. We aimed to interview participants with diverse backgrounds and roles in the cases to capture the richness of various perspectives and enhance data triangulation (Runeson et al., 2012). Participants were selected based on their experience with, or interest in, generative AI tools such as ChatGPT and GitHub Copilot. To capture diverse perspectives on the risks associated with these tools, we deliberately included participants with different roles within the software development process, including developers, architects, UX designers, managers, and security specialists. This diversity enabled us to explore how risks were perceived across both technical and managerial roles with varying levels of exposure to generative AI tools. During the analysis, we employed cross-case coding, in which the first and second authors produced codes that were mutually reviewed to enhance data triangulation (Yin, 2018). We conducted a cross-case analysis using NVivo to ensure consistency in the risks and causes across the three cases and to identify emerging differences among them.

Participant	Role	Case	Interview	Validation Interview	Workshop
P1	Security Director	The Bankers	62 min.		
P2	IT-Operations Manager	The Bankers	61 min.		
P3	Architect	The Bankers	58 min.		
P4	Front-End Developer	The Bankers	47 min.	16 min.	
P5	Back End Developer	The Bankers	60 min.	16 min.	
P6	Back-End Developer	The Bankers	52 min.		
P7	UX-Designer	The Bankers			120 min.
P8	Front-End Developer	The Insurers	43 min.		
P9	Security Architect	The Insurers	53 min.	18 min.	120 min.
P10	Business Analyst	The Insurers	55 min.		
P11	UX-Designer	The Insurers	67 min.		
P12	Back-End Developer	The Insurers	62 min.		
P13	IT-Manager	The Insurers	32 min.		
P14	Scrum Master	The Insurers		18 min.	
P15	Senior Product Manager	The Automaters	56 min.	35 min.	120 min.
P16	Director of Technology & Platform	The Automaters	58 min.	35 min.	120 min.
P17	System Design & Software Architect	The Automaters	41 min.		
P18	Senior Software Developer	The Automaters		35 min.	

Table 1: Overview of our participants, the three cases, and data collection procedures. The presence of a time duration indicates the participant's presence at a specific data collection procedure.

The data analysis proceeded in five overall steps: (1) identification of risks, (2) coding across cases, (3) identification of causal relationships, (4) construction of the causal maps, and (5) validation of the maps. The analysis was initiated by reviewing the transcripts to identify risks that adhere to the definition of "the possibility of loss." Throughout the data analysis, we noticed a pattern in which participants explained risks that caused other risks; therefore, we adopted the causal mapping method to illustrate the systemic implications of these risks. Causal mapping enables comparative analysis of different organizational actors and reveals both differences and similarities in their perspectives (Laukkanen, 1998, 1994). Additionally, we selected this method because it has proven to be an appropriate tool for visualizing the causes of risks in similar research beyond software engineering (Ackermann and Alexander, 2016; Ackermann and Eden, 2020; Ghobadi and Mathiassen, 2014; Hein et al., 2025), particularly regarding perceptions of technology adoption (Kjærgaard and Jensen, 2014).

Causal mapping enables practitioners to articulate their causal reasoning regarding specific concepts within a real-world context. This method visually illustrates patterns of concepts and causal beliefs, capturing the explicit statements of various stakeholders. The process of eliciting causal relationships emphasizes identifying instances where one concept (A) is seen as leading to another concept (B) or where B is understood as a result of A. These causal connections are depicted through models utilizing nodes and arrows, with nodes signifying concepts within a specific context and arrows depicting the actors' perceived causal relationships among these concepts (Laukkanen, 1998, 1994). Using the causal mapping approach, we constructed three causal maps in total, one for each of the case companies. We linked risks in the causal maps when two statements, each describing a risk, suggested a cause-and-effect relationship between them, indicated by phrases such as "because," "due to," or "as a result of". The first author initially inferred the causal links, which the second and third authors then reviewed. As a specific example, the following quote: "I have no idea what it bases its knowledge on or how it stores the data you sent it to. It's somewhat hidden in terms of ChatGPT," alludes to the risk of inadequate data management by AI, as the statement shows ambiguity in how generative AI tools handle their users' data. This quote further explains the risk posed by *Inadequate data management of AI*: "The use of large language models introduces a security risk because we are not entirely in control of what it sends elsewhere," thereby causing the risk of *Lack of control over AI*. The word "because" acts as the link connecting these two risks. No additional preset criteria, frameworks, or metrics were employed to

develop the causal links. Our focus was on ensuring the trustworthiness and reliability of our findings through validation with our participants.

Following the creation of the maps, we validated these with representatives from the case companies to increase their trustworthiness and reliability (Runeson et al., 2012; Yin, 2018). We explained the rationale for the risks and their causal relationships to the representatives, continuously seeking their feedback. In the validation interviews, we were particularly interested in confirming the risks and causal relationships that were ambiguous during analysis. For instance, the *Automaters* sought to establish a relationship between the two risks: *Lack of AI competencies* and *Insufficient validation of AI output.* Additionally, we conducted these validation interviews a year after the initial interviews to enable insights into the evolution of risks and causal relationships. Each of the companies provided feedback on risks that had become less relevant or increased in relevance, while also identifying a few new risks. Finally, after validating and finalizing the causal maps, we conducted a workshop with representatives from the case companies. This workshop aimed to ensure the maps' cohesiveness and to facilitate engagement among the case companies, exploring whether their causal maps would support discussion or reflection on their own. The workshop enabled us to conduct a combined causal map capturing all the risks and causes from the case companies. The risk nodes and relationships were not determined by frequency, salience, or co-occurrence. Instead, we included all the risks and causal relationships identified in the initial causal maps. We used participants' feedback during validation interviews and workshops to assess the salience of these risks and relationships. This process helped decide whether risks or causal links should be added or removed to accurately reflect the risks associated with using generative AI in software development. As a result of the feedback, the causal maps changed slightly (e.g., risks were dropped, or new causal relationships were added) for each case. Examples of these changes will be provided throughout the findings section.

4 Findings

Risks represented in blue indicate that they were noted to have decreased in emphasis during the validation interview or workshop one year after initial adoption. In contrast, risks represented in red denote that they have increased in emphasis. We distinguish between root risks, tail-end risks, chaining risks, and isolated risks in organizations' causal reasoning regarding the use of generative AI in software development. Root risks are defined as risks that solely cause other risks without being caused by any risks themselves, serving as the root of a causal reasoning chain in software organizations' perceptions of risk. Tail-end risks, conversely, do not cause any additional risks, representing the tail-end of a causal reasoning chain. Chaining risks are part of a reasoning chain involving risks that are neither root nor tail-end. In other words, chaining risks are those risks that cause them and also cause other risks. A chain of risks within a causal map can form a circular reasoning loop, without a tail-end risk; in such cases, there is no definitive conclusion in the causal rationale, creating an explanation that may appear to be perpetually ongoing. Finally, isolated risks are neither caused by other risks nor do they cause additional risks; they represent standalone concerns in the organization's perception of risk.

4.1 The Bankers

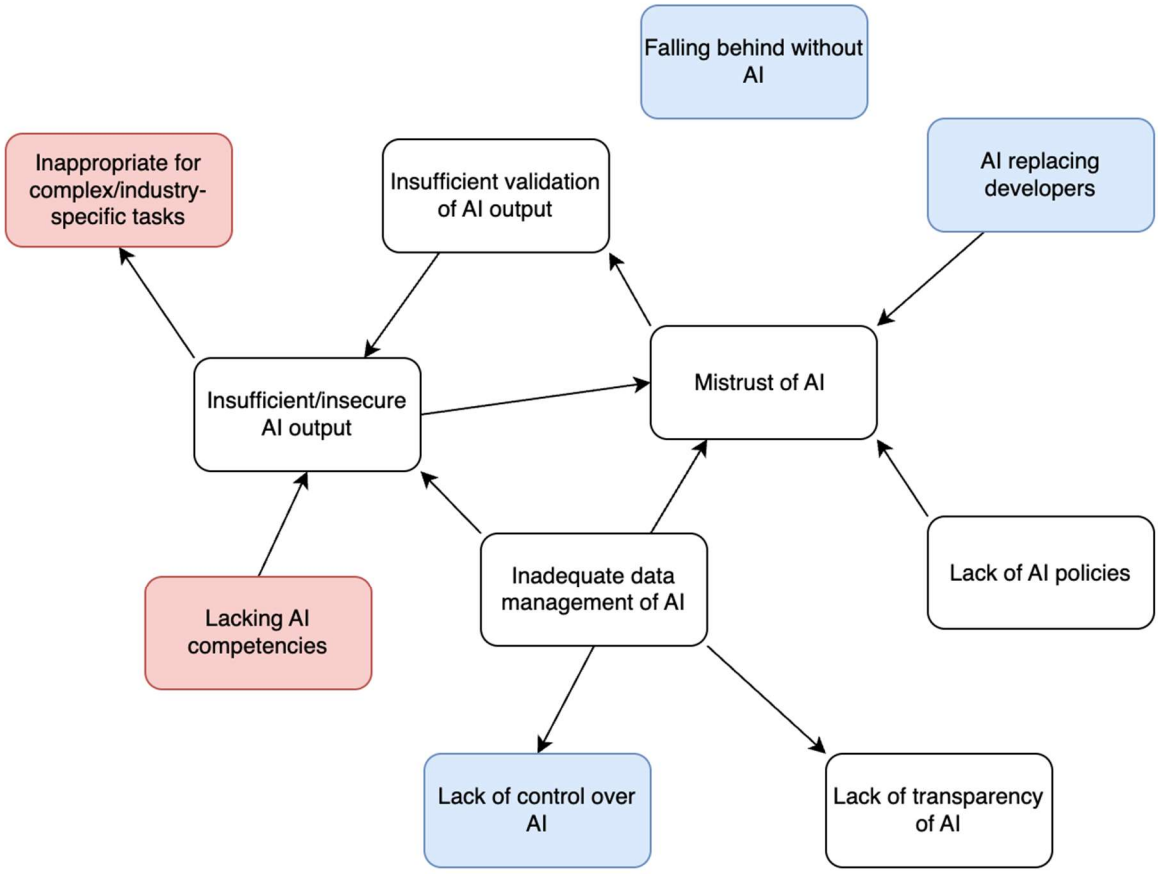


Figure 1: Causal map of the Bankers' risks and associated causes of using generative AI in software development [Blue risks show a decreasing emphasis, and red risks show an increasing emphasis after 1 year of use].

At the start of our investigation, the *Bankers* had just begun using generative AI tools, with ChatGPT and GitHub Copilot as the primary tools. During the validation interview, they decided to remove an initial concern about bypassing usage protocols, as the risk was deemed not relevant to their software development.

Figure 1 shows that *Inadequate data management of AI* is a root risk, with no other risks identified as its causes. This risk pertains to the uncertainty surrounding how AI tools store and manage data, and it is considered a contributing factor to four other risks. E.g., the tail-end risk, *Lack of control over AI*, which has seen a decreasing emphasis after the *Bankers* started hosting their generative AI locally. Nevertheless, the IT-operations manager noted: "The use of large language models introduces a security risk because we are not entirely in control of what it sends elsewhere" (P2). The *Bankers*, also see *Inadequate data management of AI* as the cause for the tail-end risk, *Lack of transparency*: "I have no idea what it bases its knowledge on or how it stores the data you sent to it. It's somewhat hidden in terms of ChatGPT" (P4) … "it remains a black box how ChatGPT truly works" (P1).

Mistrust of AI is a central chaining risk in the *Bankers'* causal explanations of their concerns regarding generative AI, especially for the technology's ability to manage users' data. One of the developers explained: "We own our software, and since I do not know how or where the data I sent to it is stored, I wouldn't use it to transfer code to it in that manner. It's a matter of trust" (P5). During the validation interview, one year after our initial interviews, two developers explained that the risks of *Lack of control over AI* and *Inadequate data management of AI* have diminished in emphasis. Since the *Bankers* are now hosting their version of generative AI internally, they mentioned that the risk of *Inadequate data management of AI* has taken on new emphasis. Managing the data internally has proven to be

challenging, as they have never attempted anything like this before. Therefore, the *Bankers'* causal map only indicated that *Lack of control over AI* has decreased in emphasis, not *Inadequate data management of AI.*

The *Mistrust of AI* is further seen as caused by other risks, such as the root risk, *Lack of policies*: “It clearly must be used with caution, no matter what. I think people are a bit nervous about using it right now because we do not have any policies for using it internally yet” (P5). A year after implementing generative AI tools at the *Bankers*, they now have internal policies for using generative AI. However, the risk was not seen as having decreased emphasis in the validation interview. Moreover, the root risk, *AI replacing developers*, is also cited as causing *Mistrust of AI*: “I have considered potential unemployment and being replaced by AI. I do not believe that is where we are currently, but considering how rapidly these tools are evolving, I am quite concerned about how my future will look" (P4). During the validation interview, the *Bankers* noted that they are not nearly as worried about being replaced by generative AI. After using generative AI tools for a year, they have repeatedly observed that they are unsuitable for more complex tasks and, therefore, cannot always assist them. Thus, the tail-end risk, *Inadequate for complex tasks/industry-specific tasks,* is marked with red to indicate an increased emphasis in the *Bankers'* causal map.

Consequently, this *Mistrust of AI* underscores the emphasis on validating the output of generative AI. It thus causes the chaining risk *Insufficient validation of AI output*: “It may be beneficial to use, but it also presents a threat because who made this (code)? What is it we do not see at first glance in the code, and that’s why we must validate the code very thoroughly before we use it” (P2). One of the developers presents an example where code generated by generative AI can lead to insecure code being integrated into their codebase if it is not adequately validated: “I asked ChatGPT for help with some regex code, and I committed the code, but our testing software flagged several warnings for potential issues introduced by the AI. Then, I asked it to revise the code to eliminate those security threats, and the code successfully passed through the build process without any warnings” (P4). The developer and the *Bankers* are fortunate to have capable testing tools. If the output is improperly validated, it results *in Insufficient or insecure AI output,* causing *Mistrust of AI* and necessitating revalidation. The three chaining risks: *Mistrust of AI, Insufficient validation of AI output,* and *Insufficient/insecure AI output* form a circular chain of reasoning, as it is difficult to determine when the output from the AI is sufficiently validated.

Moreover, *Insufficient/insecure AI output* was explained to be caused by the root risk, *Lacking AI competencies:* “I am not very experienced in using ChatGPT or similar tools. It’s unclear to me how I use it properly and generate the output that I want” (P6). The representative from the *Bankers* at the workshop stated that the risk continues to be an issue and likely will persist in the future. In an effort to minimize this risk, they have invested heavily in addressing the risk: “We are inviting all of our developers to an event soon, where we have also invited representatives from IBM to teach us more about using the tools and how to properly generate the output you want” (P7). The risk is viewed as something that needs to be addressed, as they are utilizing numerous resources to bring together their developers and IBM in a collaborative effort to reduce it. Consequently, we have marked the risk in red to indicate its increased emphasis. In addition, the risk of *Inadequate management of AI data* was noted to cause hallucinations, causing the chaining risk *Insufficient/insecure AI output.* They explained, "With these AI tools, you need to be aware that they often misrepresent information due to how they handle the data on which they base their output" (P3). This highlights the need to validate the output from generative AI to avoid integrating insecure code or solutions.

Lastly, the *Bankers* identified the isolated risk *Falling behind without AI,* which did not have any adjacent causes. They insisted on its inclusion in their causal map, as one manager explained in the initial interview: “I can’t imagine anything other than if generative AI tools become a market standard and we do not use them, we will fall behind our competitors” (P2). When generative AI tools were first introduced, the *Bankers* were quite concerned about missing potential improvements and being outclassed by their competitors if they did not adopt these tools as fast as possible. However, during the validation interview, they explained that this risk is not as concerning as it once was because: “What do we really miss out on if we do not use generative AI?” (P5). A year after implementing generative AI

tools at the *Bankers*, the hype surrounding productivity increases and promises of replacing developers has diminished. As the closing remark from one developer at the validation interview eloquently states: “The fear of the unknown has significantly decreased since we started using it” (P5). Generative AI is no longer perceived as a magical black box technology; like all technologies and people, for that matter, they have flaws that need to be considered and addressed.

In summary, the *Bankers’* causal map revealed a circular chain of reasoning among *Mistrust of AI, Insufficient validation of AI output*, and *Insufficient/insecure AI output*. Meanwhile, the tail-end risks of *AI replacing developers, Lack of control* over, and the isolated risk of *Falling behind without AI* have diminished in emphasis since the initial adoption of generative AI tools. The latter risk has been downplayed, as they opted to host generative AI internally, thereby gaining greater control over it. In addition, the two former risks have decreased in emphasis as they now have a greater understanding of these tools' limitations.

4.2 The Insurers

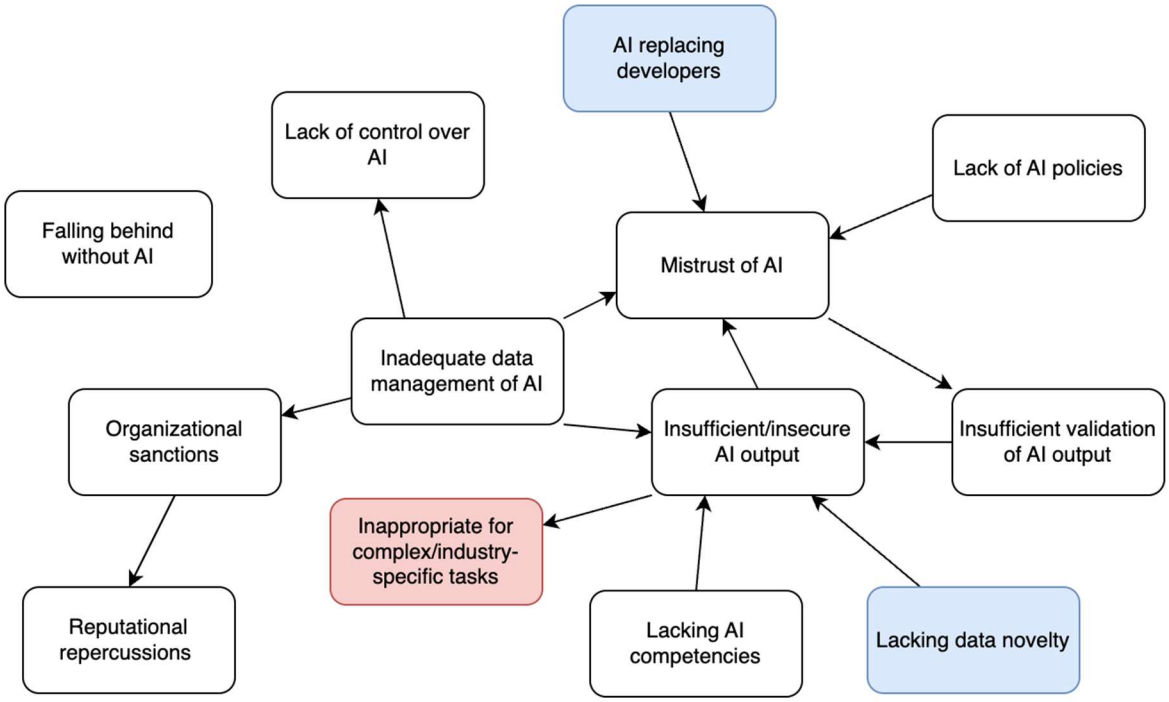


Figure 2: Causal map of the Insurers’ risks and associated causes of using generative AI in software development

They experimented with various tools, primarily using ChatGPT and GitHub Copilot. The *Insurers* discarded the risk of extensive AI maintenance, noting during the validation interview that it was more related to machine learning than generative AI. Additionally, they discarded insufficient AI performance, which was considered part of the *Insufficient/insecure AI output* risk.

The chaining risk of *Insufficient/insecure AI output* was explained to be caused by a few of the other risks in the causal map, such as the root risk, *Lacking data novelty*. One participant noted, "Currently, you cannot use it for everything because it relies on older data”. (P8). However, *Lacking data novelty* was later discussed during the validation interview a year later, where it was reported to have decreased in emphasis: “ChatGPT and other similar tools are now using the latest data available, so it’s really not as important as before” (P9). The participants in the validation interview were hesitant to rule out the risk entirely, as they were uncertain whether this applied to all the tools they were using and if it was also applicable to topics they had not yet explored. Additionally, the root risk *Lacking AI competencies* was similarly highlighted as a cause for *Insufficient/insecure AI output*: “I do not know enough about it to use it properly” (P10) and: “I believe that a lack of AI competencies can lead to Insufficient/insecure

AI output because you need experience to use the tools optimally" (P14). Moreover, the chaining risk *Insufficient validation of AI output* was most frequently mentioned by the Insurers, explaining that it causes *Insufficient/insecure AI output*: "Some of this AI-generated code can be really good, but it can also be of very poor quality, so we have to ensure that we review the code or use our integrated testing tools" (P9).

Unreliable generative AI outputs breed mistrust, highlighting the need for validation, as shown in Figure 2. This quote captures the link between *Insufficient/insecure AI output* and *Mistrust of AI*: "That would make it more trustworthy if you were guaranteed to get the right answer. It's something that undermines my trust in it" (P12). The same is true for the chaining risk, Inadequate data management of AI, causing mistrust: "It's a black box how it works in the machine room… Then, it becomes a matter of trust in those maintaining the AI that they don't look at all our code. That's where the problem arises if you send the AI all your code to optimize it or otherwise transfer sensitive data" (P9). The uncertainty surrounding generative AI functionality and the insecure use of technology foster mistrust. Developers also fear being replaced by AI: "Because it's just AI that should handle those junior tasks like generating classes… In general, I think our work could become quite irrelevant... I'll be out of a job" (P8). During validation, it was noted that the risk of *AI replacing developers* had decreased as they recognized the limitations of generative AI for complex, industry-specific tasks: "As the concern for AI replacing developers has decreased in emphasis, the risk of being inappropriate for complex and industry-specific tasks has increased. They are not only opposites but also contradictory, which seems clear to me" (P14). Thus, the two risks are marked in red, indicating a shift in emphasis since the initial experimentation.

Moreover, the risk of *Insufficient/insecure output of AI* was highlighted as causing the tail-end risk, I*nappropriate for complex/industry-specific tasks* from various angles. For instance, a developer explained: "I was using Copilot and wanted to work on an insurance policy for a specific data object. If I wanted a method that removes, adds, adjusts, or handles errors for something that is suddenly domain-specific and highly tailored to our solution, it wasn't much help" (P8). The AI also proved inappropriate for complex design tasks: "I would not use it to generate an entire design prototype. I might risk that it's based on a different major insurance company's design" (P11) or tasks specific to an insurance company: "I would be hesitant to use it for something related to insurance as it does not understand our business or customers" (P10). Thus, the risk of I*nappropriate for complex/industry-specific tasks* was a persistent concern for the *Insurers* both during and a year after their initial experimentation with generative AI tools.

Furthermore, the risks of *Insufficient/insecure AI output* and *Inadequate data management of AI* were raised as concerns that could cause the chaining risk. *Organizational sanctions* should the generative AI wrongfully receive sensitive data or produce insecure code: "A directive or a reprimand from the Financial and Data authorities, which often leads to negative press coverage" (P13). They are concerned about the potential for sanctions, which could have a significant financial impact on them. Still, they are especially concerned about the poor publicity: "It can result in some severe fines, but the negative coverage really damages our image" (P13). Hence, we connected *Organizational sanctions* with the tail-end risk, *Reputational repercussions.* Finally, the *Insurers* were also concerned about the isolated risk, *Falling behind without AI*: "If you look at the traction that AI-generated code has gained, we need to keep up. If we don't, we will fall behind on many fronts. We'll see what kind of impact it has, but if that impact is significant, we'll be far behind our competitors" (P9). We were unable to identify any causes related to the risk, nor could our interviewees at the validation interview. However, the *Insurers* remarked at the validation interview that they are not as worried about this risk as they once were since the impact of AI has not been as substantial as they initially believed. However, unlike the *Bankers,* they do not believe the risk has shifted in emphasis simply because generative AI tools have become very popular among developers: "We want to attract employees, and it is important to stay on top of the current trends. We are not concerned about the customer side, but it is crucial to appear attractive by demonstrating that we utilize AI to draw in new employees" (P9). Even though they are no longer worried about being outperformed by competitors or their ability to attract customers, it is vital not to fall behind in attracting new developers who want to engage with the latest technologies.

The *Insurers'* causal maps revealed that the risks of *AI replacing developers* and *Lacking data novelty* have decreased in emphasis. The latter risk has become less significant since generative AI tools now have access to the newest data, and the former has diminished in relevance due to awareness of the tools' limitations, particularly their inappropriateness *for complex/industry-specific tasks*. The *Insurer's* causal map also repeats the same chain of reasoning between the risks: *Mistrusts of AI, Insufficient validation of AI output,* and *Insufficient/insecure AI output,* as the *Bankers*.

4.3 The Automaters

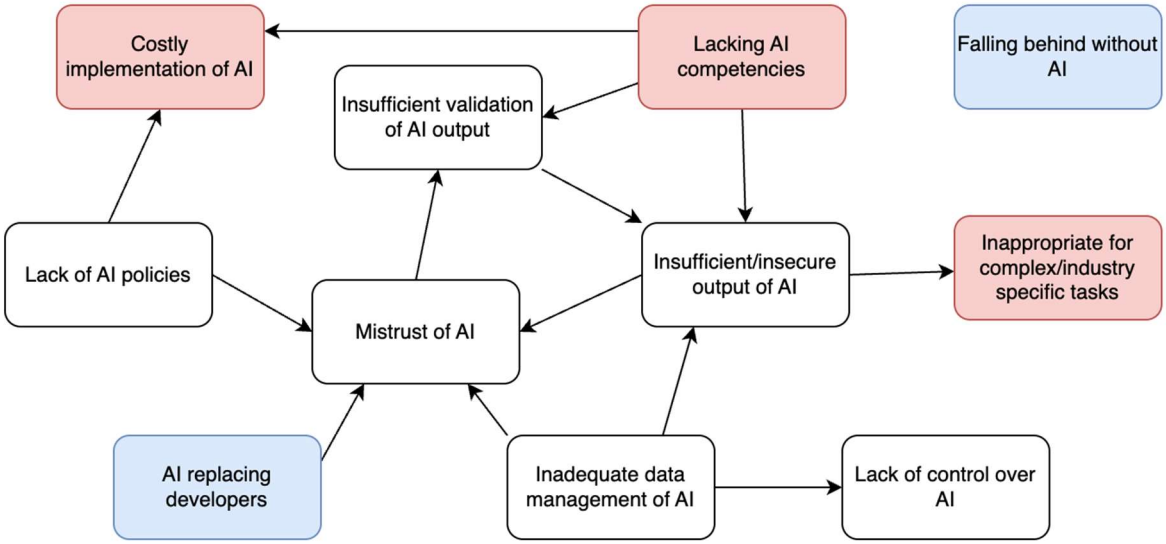


Figure 3: Causal map of the Automaters' risks and associated causes of using generative AI in software development

Throughout our investigation, the *Automaters* adopted ChatGPT and GitHub Copilot for their developers, maintaining a trial-and-error approach to utilizing these tools.

Several risks were identified, causing the chaining risk *Insufficient/insecure AI output*. For example, the root risk, *Lacking AI competencies,* was cited: "If you don't understand it, then you might get incorrect results that lead you to take actions that are not ideal" (P16). During the validation interview, the *Automaters* noted that this risk had gained emphasis as generative AI tools rapidly evolve, making it challenging to keep up with their development. They stated, "There are currently no reliable guidelines for using generative AI" (P16). The fast-paced evolution and introduction of new tools not only make it hard to stay current but also to find trustworthy guidelines for using generative AI in software development. As shown in Figure 3, we colored the risk red. Additionally, during the validation interview, they identified a causal relationship between *Lacking AI competencies* and the chaining risk, *Insufficient validation of AI output*, noting that a lack of competencies can similarly cause an undesirable AI output and inadequate validation of the output. Similarly, the risk of *Insufficient validation of AI output* could cause *Insufficient/insecure AI output*: "I believe the challenge in using AI is to avoid engaging blindly; we must validate both the input and the output. After all, garbage in means garbage out" (P15).

Considering the root risk *Inadequate data management of AI*, the *Automaters* expressed concerns about the ambiguity regarding how the generative AI tools store and manage data, as well as how this can cause untrustworthy responses from the AI. On the other end of the spectrum, the risk of *Insufficient/insecure output of AI* can cause the tail-end risk *Inappropriate for complex/industry-specific tasks*: "It makes sense to use AI for the simple tasks, in contrast to the complex tasks… where one must determine how things are supposed to fit together, and other aspects that are even more complex" (P17). Their generative AI tools are particularly unsuitable for operating systems development for Linux: "Currently, there are no AI tools appropriate for operating systems development… especially regarding

the hardware platform and managing the Wi-Fi component" (P17). As part of the validation interview, they noted that they have become increasingly aware of the inefficiencies of their generative tools when applied to complex tasks. They are now more hesitant to use them for complex and industry-specific tasks, such as operating systems development, as they frequently find themselves wasting time on generative AI for these responsibilities. Therefore, we found it fitting to color-code the risk to reflect this increased emphasis.

Like the other two cases, the *Automaters* displayed mistrust stemming from the risks of *Inadequate data management of AI* and *Insufficient/insecure AI output.* They highlight the data management aspect: "The most critical aspect is data security and avoiding leaks of our source code, because if you have that, you can also break into it" (P16). Another participant elaborates: "We do not know what happens as soon as you install Copilot. What do they have access to? What data are they stealing?" (P15). They also highlight concerns about AI's output: "You might have a simple while loop that you aren't allowed to use because someone has a license on it. We then need to ensure that we don't just take it" (P15). This mistrust of AI leads them to validate the output before integrating the code. In line with the risk of *AI replacing developers,* they express: "The fact that I can provide it with functional requirements, and it can generate something useful means that if it becomes more advanced, we risk it replacing developers" (P16). Amid the validation interview, they noted that the risk had significantly decreased since they began using the tools: "It is no longer much of a concern for us" (P18). The risk became less significant as they grew more familiar with the tool's limitations, particularly for complex and industry-specific tasks.

Moreover, their concern about the root risk, *Lack of AI policies,* indicates an underlying Mistrust of AI, as explained: "We currently lack policies internally that dictate how we can use it. It's quite a barrier for us that we do not have them" (P16). In continuation of the former statement, the participant explains how their *Lack of policies* and *Lacking AI competencies* can cause the risk tail-end risk, *Costly implementation of AI*: "I am skeptical of whether we as an organization are equipped to use it efficiently, so that it creates value, and we do not waste money needlessly" (P16). They highlighted the risk of *Costly implementation of AI* during the validation interview and reiterated at the workshop that its emphasis has substantially increased. In the validation interview, one participant remarked, "It's important for us that it meets our expectations regarding the economic aspects of using it" (P15). This was also restated at the workshop, where a critical aspect for them is ensuring their investment in the technology aligns with their economic expectations for its implementation. There was considerable hype surrounding generative AI when it first launched, and it appears to have fallen short of those early expectations. During our initial interviews, they expressed a concern for the isolated risk *Falling behind without AI*: "I'm afraid we'll be left behind if we don't start implementing these tools. Denmark has high salary costs. We must compete on efficiency and job satisfaction as well" (P15). However, a year later, in the validation interview, they indicated that the risk had diminished in emphasis: "The hype around generative AI has significantly lessened. We still need to monitor developments in the AI field and acknowledge that our customers are urging us to do so" (P15). The risk of *Falling behind without AI* was a concern for them early in our examination, as they worried about their ability to compete with others in terms of efficiency. However, as the hype surrounding generative AI has stagnated and they have become more aware of its limitations, they are now more focused on aligning their economic expectations with the implementation of generative AI tools to justify their investment.

The *Automaters'* causal map disclosed that the root risk *Lacking AI competencies* has gained emphasis, as they struggle to keep up with the rapidly changing generative AI landscape. After a year of experimentation, they lack reliable guidelines for using these tools. Additionally, the tail-end risk, *Costly implementation of AI*, is now more significant as they assess the balance between efficiency and cost to justify their investment. Further, the risk of *Inappropriate for complex/industry-specific tasks* is increasingly recognized due to their limitations, especially in areas like operating systems development. The *Automaters'* causal map reflects the same concerns as the Insurers and *Bankers* regarding risks such as *Mistrust of AI*, *Insufficient validation*, and *Insufficient/insecure AI output*.

5 Combined causal map of the three cases

This subsection synthesizes the three cases by comparing the main cross-case patterns in how generative AI risks are causally related and interpreted across the organizations.

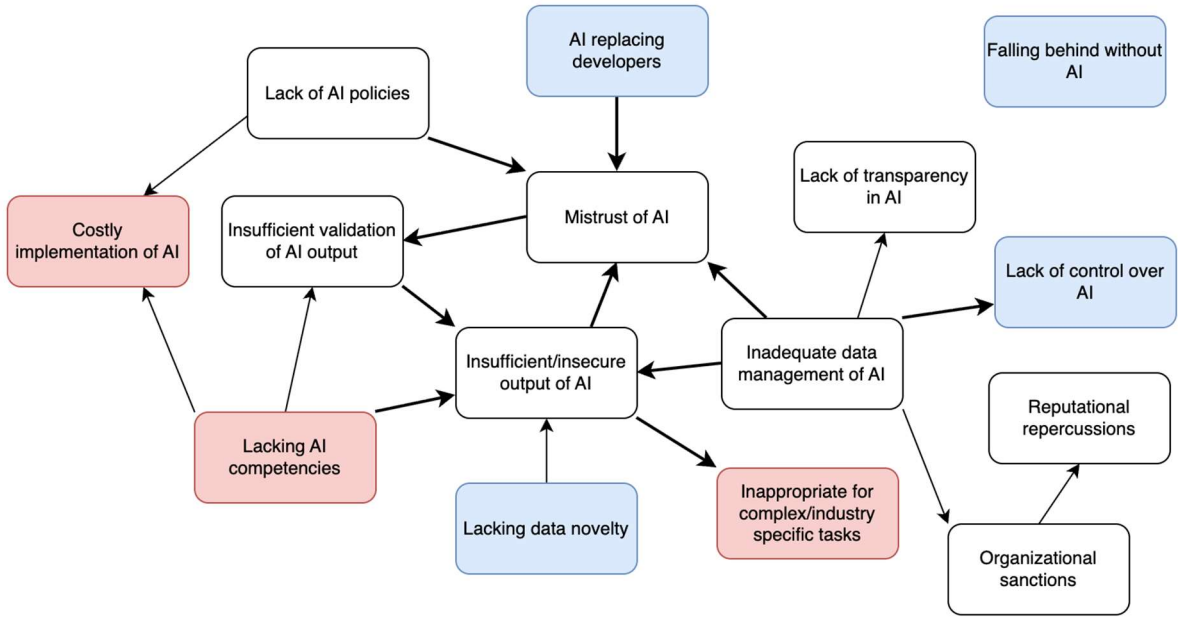


Figure 4: Combined causal map of the three cases' risks and associated risk causes of using generative AI in software development.

In Figure 4, we have aggregated all identified risks and causes into a comprehensive causal map that encompasses all cases. This aggregated map presents comparisons among the three company cases and should not be seen as an attempt at generalization. The thick arrows indicate causal explanations common to all cases, while the thin arrows represent causal explanations derived from only one or two cases. The map illustrates the root-cause explanations used across all three software organizations. These explanations include a consistent emphasis on *inadequate data management* of AI, *Lack of AI policies,* and a decreasing emphasis (marked in blue) on *AI replacing developers*. Meanwhile, the *Bankers* and the *Automaters* recognized an emerging emphasis (marked in red) on *Lacking AI competencies,* while the *Insurers* maintained their emphasis. The root risk, *Lacking data novelty,* with a decreasing emphasis, was uniquely identified by the *Insurers*.

The map illustrates a circular reasoning loop regarding the risks associated with AI, which all three software organizations repeat. This reasoning relates to the *Mistrust of AI*. The three organizations infer this *Mistrust of AI* from the risk of *Inadequate data management of AI*, *Lack of AI policies*, *Insufficient/insecure AI output*, and *AI replacing developers*. This mistrust compels software organizations to validate the quality and credibility of AI outputs, preventing the integration of low-quality or insecure code into their codebase. If validation is not conducted or is inadequate, it causes *Insufficient/insecure AI output* due to the inherent risk that the AI's output may be flawed or insecure. The *Insufficient/insecure AI output*, in turn, causes *Mistrust of AI*, thereby perpetuating the circular causal chain of reasoning until the AI output has been adequately validated. However, they cannot be certain when AI-generated code is sufficiently validated, creating this circular reasoning loop.

Tail-end risks that do not serve as the reason for other risks (*costly implementation of AI, lack of control over AI, lack of transparency of AI, reputational repercussions, and inappropriate for complex/industry-specific tasks)* were revealed by the causal maps. All three organizations identified two key risks: *Inappropriate for complex/industry-specific tasks* and *Lack of control over AI.* While all organizations had increased their emphasis on *Inappropriate for complex/industry-specific tasks,* only the *Bankers* decreased their emphasis on *Lack of control over AI*. Only the *Automaters* recognized the risk, *Costly*

implementation of AI, and did so with an emerging emphasis on the risk. The *Insurers* were the only case to acknowledge the risk *Reputational repercussions,* and the risk had a consistent emphasis. The tail-end risks reveal distinct differences between the cases as the *Bankers* are uniquely preoccupied with controlling AI, the *Insurers* are uniquely concerned with their image, and the *Automaters* are succinctly worried about the cost-efficiency of adopting generative AI in software development.

Finally, Figure 4 highlights one isolated risk: *Falling behind without AI*, which is not involved with any other risks in the map. All three software organizations recognized the risk, but the *Bankers* and the *Automaters* share a declining emphasis, while the *Insurers* maintained a consistent emphasis of the risk.

6 Discussion

This section discusses the findings in relation to prior research and outlines implications for both research and practice, as well as threats to validity and future research directions.

6.1 Implications for research

We contribute to the research on generative AI in software development (Asare et al., 2023; Russo, 2024; Sergeyuk et al., 2025) by showing how causal mapping can reveal explanations of risks that three software organizations associate with generative AI. Causal mapping has been shown to be useful for causally explaining project risks (Ackermann et al., 2014; Ackermann and Alexander, 2016; Hein et al., 2025), barriers to knowledge sharing (Ghobadi and Mathiassen, 2014), and perceptions of technology adoption (Kjærgaard and Jensen, 2014). Our study extends this body of research by demonstrating that causal mapping is also suitable for illustrating the causality of risks associated with generative AI and the perceived evolution of these risks. Moreover, we contribute to the literature on the concept of causality of risks (Bjarnason et al., 2012; Boholm and Corvellec, 2011; Carlo et al., 2008; Yu et al., 2015) by introducing the terminology of root risks, chaining risks, and tail-end risks to categorize the causality of risks. Specifically, we expand on Boholm and Corvellec's (2011) relational theory of risk by stating that risk is interpreted within a specific context. Additionally, we extend their concept of a risk, consisting of an object at risk and a risk object, to also apply to relationships between risks. Our causal mapping of the three software organizations reveals substantial similarities in the risks they identified and the causal explanations they provided. These similarities are common among software organizations, with variations in context based on industry, size, customer segment, and software development process. This similarity was particularly evident in the circular reasoning involving *Mistrust of AI, Insufficient validation of AI output,* and *Insufficient AI output.* All three case organizations' causal maps (c.f. Figures 1, 2, 3) show this circular explanation with these three chaining risks.

As generative AI continues to evolve rapidly (Acharya et al., 2025; Carleton et al., 2024), we also investigated the risk explanations of three software organizations a year after adopting generative AI tools. The three software organizations then emphasized the identified tail-end risks differently. The *Bankers* reduced their emphasis on the *Lack of control over AI* but increased their focus on *Lacking AI competencies* to accommodate their Scaled Agile Framework. The *Automaters* uniquely demonstrated an emerging concern about the risk of *Costly AI implementation*, reflecting their focus on cost efficiency as a smaller software organization with limited opportunities for economy-of-scale AI tool use. Finally, the *Insurers* distinctly recognized and consistently emphasized the risk of *Reputational repercussions*, reflecting the organization's need for a trustworthy public image as a retail insurance provider. As such, these tail-end risks highlight how the different organizational contexts of the three cases shape their explanations of the risks of using generative AI in software development.

6.2 Implications for practice

Our findings present some implications for practice. First, causal mapping may be valuable not only as an analytical tool for illustrating the causes of the risks associated with generative AI in software development. Similar research highlights that causal mapping can serve as a valuable aid in decision-making. By visualizing the causes and relationships of AI-related risks, causal maps can promote

internal alignment, guide prioritization, and assist managers in communicating risks to upper management while deciding on appropriate mitigation strategies. The case organizations, all at the early stages of adoption, expressed uncertainty about managing the emerging risks, underscoring the need for practical frameworks to help them proactively oversee their AI usage.

For software organizations looking to create their own causal maps, we offer these practical step-by-step guidelines:

1. **Define the decision focus.**
   Frame a concrete question the map should inform (e.g., "Which gen-AI risks most threaten our delivery pipeline in the next two quarters, and where should we intervene first?"). Set scope (product(s), teams, lifecycle phases) and timebox the exercise.
2. **Assemble a cross-functional group.**
   Include developers, architects, operations/SRE, product, UX, data/ML engineers, and IT managers. Assign a facilitator (neutral), a scribe (captures statements verbatim), and a timekeeper.
3. **Identify risks.**
   Pull from interviews, incident/postmortem logs, threat models, pen-test findings, and developer feedback. Use a uniform sentence pattern to keep inputs crisp:
   "Because [cause], the risk of [risk object] may harm [object at risk] by [impact]."
4. **Align on mapping syntax.**
   Nodes = risks.
   Arrows = perceived causation ("A leads to B").
   Tags/Colors = Leavitt's components *(Actors, Tasks, Structure, Technology)* to maintain a sociotechnical map. Optionally mark the risk object (source of risk) vs the object at risk (what holds value and could be harmed).
5. **Run a fast-mapping workshop.**
   Start with silent sticky-note brainstorming of risks, clustering by theme and Leavitt component, and then draw arrows to capture "what causes what." Press for mechanisms ("how, exactly?") rather than correlations.
6. **Classify risk types to guide attention.**
   Identify root risks (only cause others), tail-end risks (only outcomes), chaining risks (both cause and are caused), and isolated risks (neither cause nor are caused—still important if high impact).
7. **Detect loops and propagation paths.**
   Circle feedback loops (e.g., "model misuse → incident load → rushed patches → weaker reviews → more misuse"). These often signal compounding dynamics that merit priority.
8. **Revisit the causal map.**
   We recommend that software organizations revisit the first causal map after 3 months in a workshop to incorporate any changes since its creation; if the map remains stable, it should be revisited annually instead.

Second, while the nature of the risks remained consistent, differences arose in how organizations perceived and prioritized them. Given the rapid pace of evolution in generative AI technologies, software organizations need to adopt a mindset of continuous reassessment. Risks will likely evolve swiftly, and what is considered low-impact today may become high-priority tomorrow. Regarding our three cases, the risks of *AI replacing developers*, *Falling behind without AI, Lacking data novelty*, and *Lack of control over AI* have decreased in emphasis. On the other hand, the three risks, *Lacking AI competencies, Inappropriate for complex/industry-specific tasks,* and *Costly implementation of AI,* have increased significantly since adoption. Ongoing monitoring and reflection will be necessary to maintain a current and practical risk management approach for managing the risks associated with generative AI in software development. Software organizations planning on adopting generative AI tools for software development should be attentive to the root risks identified by all our cases, such as *Inadequate data management of AI data,* as this risk can cause a variety of other risks. The *Bankers* attempted to address

this risk by hosting generative AI internally in their organization, gaining more control over the data when interacting with AI.

Third, the circular reasoning loop between *Inadequate data management, Mistrust of AI,* and *Inadequate validation of AI output* signifies a mutual exacerbation. The introduction of an external code-generation tool could worsen code validation practices in software organizations, making it even more challenging to ensure code is adequately validated. To break this loop, the *Bankers* have automated static analysis tools, such as SonarQube, at key CI/CD integration points to detect insecure AI-generated code before it is merged. The *Automaters* developed AI-specific code reviews, such as verifying prompt handling, ensuring no sensitive information is exposed, and flagging insecure library usage.

6.3 Threats to validity

Drawing on Runeson et al.'s (2012) recommendations on case study methodology, we evaluate the study's validity across four dimensions: construct validity, internal validity, external validity, and reliability.

Construct validity

Construct validity concerns whether we accurately captured the concepts we studied. In this case, the risks associated with generative AI in software development. The concept of risk may have been interpreted differently by participants, as it was not explicitly defined in strict terms. This could lead to inconsistencies in what respondents understood as a risk, affecting the comparability of their responses. Since data collection relied heavily on semi-structured interviews, there is a risk that researchers might have misinterpreted participants' intentions or inadvertently assigned unintended meanings to their statements.

Additionally, we validated the causal maps through follow-up interviews with company representatives and a cross-case workshop, which enhanced the alignment between the collected data and our interpretations. Nonetheless, construct validity could still be threatened by participants' own evolving understanding of generative AI over time, as well as our interpretation of what constitutes a risk. Lastly, the causal maps derived from the initial interviews were presented to participants approximately one year after the original data collection. While this enabled us to examine longitudinal developments, it may have compromised the accuracy of feedback on the original representations due to memory decay or evolving organizational contexts. Presenting the maps earlier, closer to the initial interviews, along with a year after the first examination, could have strengthened triangulation and ensured a more immediate confirmation of the concepts and causal relationships as understood by participants at that time.

Internal validity

Internal validity pertains to the accuracy of identified causal relationships and is especially pertinent given our reliance on causal mapping. Studying multiple companies allowed for cross-case comparison, but it may have limited the depth of insight into the contextual nuances of each case. A single, in-depth case could have offered more details on the causal chains of specific risks. Although causal mapping helps to visualize risk causality, it is subject to interpretation by the researchers, who may have imposed structure on relationships that were less explicit in the raw data. While the causal maps were constructed from participants' statements and refined through validation interviews, these maps are ultimately interpretations rather than objective truths.

Lastly, causal mapping posed interpretive challenges, particularly regarding the directionality of relationships among concepts. In several cases, determining whether a specific risk caused another or was instead a result of it proved ambiguous, resembling a "chicken or the egg" dilemma. Although these decisions were based on participant narratives and researcher consensus, the risk of misinterpreting causal direction remains a limitation, particularly given the controversies surrounding perceived cause–and–effect in organizational contexts (Markus and Rowe, 2018).

External validity

External validity relates to the generalizability of the findings. Although the case companies were selected for their size, industry, and variation in customer base, all three are Danish software organizations in the early stages of adopting generative AI. Therefore, the findings may not apply to companies in different national, regulatory, or technological contexts or those with more advanced AI integration. Nevertheless, the relative consistency in the types of risks identified across the three organizations indicates that some findings might have relevance beyond the immediate context of the cases. Our exclusive use of qualitative methods means that replication may be challenging, as another researcher might not elicit the same responses or interpret the data in the same manner. The absence of quantitative measures limits the ability to statistically validate the findings. We prioritized in-depth detail and richness in our three cases over generalizability.

Reliability

Reliability refers to the extent to which the study can be replicated with similar results. The coding, causal mapping, and cross-case analysis depended on the research team's interpretations, potentially threatening reliability. Despite discussions to reach a consensus, researcher bias could still influence the results. We adopted specific strategies to enhance reliability: all interviews were recorded and transcribed, coding was performed and collaboratively reviewed by two authors, and the analysis was thoroughly documented. Furthermore, member checking was utilized through map validation and a follow-up workshop. However, the qualitative nature of the research and the interpretive aspects of causal mapping still pose challenges to replicability, particularly considering the evolving landscape of generative AI. Therefore, we chose to revisit the cases one year after adopting generative AI tools to determine whether anything had changed since their early adoption.

6.4 Future research

This study examined how software organizations explain the risks associated with generative AI using a multi-case qualitative approach. Our results revealed that the rapid evolution of generative AI has implications for transforming organizations' perceptions of these risks in software development. Therefore, future research could build on our insights and explore how these risks evolve over time by conducting longitudinal studies that trace organizational responses as generative AI becomes more integrated into development workflows. This should be viewed in consideration of the future evolution of AI that software companies are exploring. The expectation for the subsequent significant development in AI is that software companies will adopt Agentic AI. Future research must consider this new technology, as it may introduce additional implications for practice, particularly regarding the lack of coding control and the actions taken by agentic AI (Acharya et al., 2025). Although we interviewed participants with diverse roles, including developers, architects, managers, and security specialists, we did not observe clear systematic differences in how these roles explained the causes of generative AI risks. Participants across roles largely shared similar causal reasoning, particularly regarding mistrust of AI, validation practices, and concerns about AI-generated code quality. Future research could explore whether role-specific differences emerge in organizations with more mature generative AI adoption and have used the tools for some time.

Additionally, while this study identified risks qualitatively, follow-up research could develop and apply quantitative instruments, such as structured surveys or risk assessment tools, to validate and generalize the findings across a wider sample of organizations. By grouping the risks identified in our study into categories based on similar characteristics, we encourage further research to employ statistical methods, including Bayesian networks (Brændeland et al., 2010) or structural equation modeling (Russo, 2024; Russo and Stol, 2021), to explore predictions or correlations between the risks associated with using generative AI in software development. Finally, little is known about how to break out of the causal reasoning loop, and the literature on risk mitigation strategies for addressing risks associated with using generative AI is scarce. Thus, we call for more actionable research on risk mitigation strategies for managing generative AI risks in software development, for instance, through action research (Iversen et al., 2004) to test mitigation strategies in practice.

7 Conclusion

We employed a multi-case study to examine software organizations' explanations of the risks of using generative AI in software development and how these evolve. The case study was a collaborative effort involving three Danish software organizations, and we identified and assessed the risks and their underlying causes through interviews, validation sessions, and workshops. We employed the causal mapping technique to illustrate the causal reasoning behind these risks. Our study contributes to software engineering research by:

- Demonstrating that causal mapping effectively reveals the causes of risks linked to generative AI and their perceived evolution.
- Presenting a novel terminology of root risks, chaining risks, and tail-end risks to categorize the causality of risks.
- Showing substantial similarities in the risks they identified and their causal explanations. This similarity was particularly evident in the circular reasoning loop involving *Mistrust of AI, Insufficient validation of AI output,* and *Insufficient AI output.*
- Clarifying how generative risks can change in importance over time and thus require revisiting.
- We introduce guidelines for software teams and organizations to develop their own causal maps.
- Highlighting software organizations' need for actionable frameworks and methods to address generative AI risks. This is also evident in the lack of research in this area.

Acknowledgements

The authors of this paper have used ChatGPT-4 solely as a sparring partner to generate ideas and provide feedback, thereby enhancing the article's linguistic quality while ensuring overall readability. We reviewed and edited the content as needed and take full responsibility for the submitted article.

This article is the result of collaboration with the three software development organizations and funding from DigitalLead, Denmark's national cluster for digital technologies.

References

Abioye, T.E., Arogundade, O.T., Misra, S., Akinwale, A.T., Adeniran, O.J., 2020. Toward ontology-based risk management framework for software projects: An empirical study. J Software Evolu Process 32, e2269. https://doi.org/10.1002/smr.2269

Acharya, D.B., Kuppan, K., Divya, B., 2025. Agentic AI: Autonomous Intelligence for Complex Goals—A Comprehensive Survey. IEEE Access 13, 18912–18936. https://doi.org/10.1109/ACCESS.2025.3532853

Ackermann, F., Alexander, J., 2016. Researching complex projects: Using causal mapping to take a systems perspective. International Journal of Project Management 34, 891–901. https://doi.org/10.1016/j.ijproman.2016.04.001

Ackermann, F., Eden, C., 2020. Strategic Options Development and Analysis, in: Reynolds, M., Holwell (Retired), S. (Eds.), Systems Approaches to Making Change: A Practical Guide. Springer, London, pp. 139–199. https://doi.org/10.1007/978-1-4471-7472-1_4

Ackermann, F., Howick, S., Quigley, J., Walls, L., Houghton, T., 2014. Systemic risk elicitation: Using causal maps to engage stakeholders and build a comprehensive view of risks. European Journal of Operational Research 238, 290–299. https://doi.org/10.1016/j.ejor.2014.03.035

Adzgauskaite, M., Tam, C., Martins, R., 2025. What helps Agile remote teams to be successful in developing software? Empirical evidence. Information and Software Technology 177, 107593. https://doi.org/10.1016/j.infsof.2024.107593

Alpcan, T., Bambos, N., 2009. Modeling dependencies in security risk management, in: 2009 Fourth International Conference on Risks and Security of Internet and Systems (CRiSIS 2009). Presented at the 2009 Fourth International Conference on Risks and Security of Internet and Systems (CRiSIS 2009), IEEE, Toulouse, France, pp. 113–116. https://doi.org/10.1109/CRISIS.2009.5411969

Ardic, B., Brandt, C., Khatami, A., Swillus, M., Zaidman, A., 2025. The qualitative factor in software testing: A systematic mapping study of qualitative methods. Journal of Systems and Software 227, 112447. https://doi.org/10.1016/j.jss.2025.112447
Asare, O., Nagappan, M., Asokan, N., 2023. Is GitHub's Copilot as bad as humans at introducing vulnerabilities in code? Empir Software Eng 28, 129. https://doi.org/10.1007/s10664-023-10380-1
Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D., 2001. Manifesto for Agile Software Development.
Bird, C., Ford, D., Zimmermann, T., Forsgren, N., Kalliamvakou, E., Lowdermilk, T., Gazit, I., 2022. Taking Flight with Copilot: Early insights and opportunities of AI-powered pair-programming tools. Queue 20, 35–57. https://doi.org/10.1145/3582083
Bjarnason, E., Wnuk, K., Regnell, B., 2012. Are you biting off more than you can chew? A case study on causes and effects of overscoping in large-scale software engineering. Information and Software Technology 54, 1107–1124. https://doi.org/10.1016/j.infsof.2012.04.006
Boehm, B., 1989. Software risk management, in: Ghezzi, C., McDermid, J.A. (Eds.), ESEC '89. Springer, Berlin, Heidelberg, pp. 1–19. https://doi.org/10.1007/3-540-51635-2_29
Boehm, B., Turner, R., 2005. Management challenges to implementing agile processes in traditional development organizations. IEEE Software 22, 30–39. https://doi.org/10.1109/MS.2005.129
Boehm, B.W., 1991. Software risk management: principles and practices. IEEE Software 8, 32–41. https://doi.org/10.1109/52.62930
Boholm, Å., Corvellec, H., 2011. A relational theory of risk. Journal of Risk Research 14, 175–190. https://doi.org/10.1080/13669877.2010.515313
Borg, M., Jonsson, L., Engström, E., Bartalos, B., Szabó, A., 2022. Adopting Automated Bug Assignment in Practice: A Longitudinal Case Study at Ericsson. https://doi.org/10.48550/arXiv.2209.08955
Brændeland, G., Refsdal, A., Stølen, K., 2010. Modular analysis and modelling of risk scenarios with dependencies. Journal of Systems and Software 83, 1995–2013. https://doi.org/10.1016/j.jss.2010.05.069
Carleton, A., Falessi, D., Zhang, H., Xia, X., 2024. Generative AI: Redefining the Future of Software Engineering. IEEE Softw. 41, 34–37. https://doi.org/10.1109/MS.2024.3441889
Carlo, J.L., Lyytinen, K., Boland, R.J., 2008. Systemic Risk, IT Artifacts, and High Reliability Organizations: A Case of Constructing a Radical Architecture. All Sprouts Content 4.
Chatzipoulidis, A., Michalopoulos, D., Mavridis, I., 2015. Information infrastructure risk prediction through platform vulnerability analysis. Journal of Systems and Software 106, 28–41. https://doi.org/10.1016/j.jss.2015.04.062
Cheng, R., Wang, R., Zimmermann, T., Ford, D., 2023. "It would work for me too": How Online Communities Shape Software Developers' Trust in AI-Powered Code Generation Tools.
Derner, E., Batistič, K., Zahálka, J., Babuška, R., 2024. A Security Risk Taxonomy for Prompt-Based Interaction With Large Language Models. IEEE Access 12, 126176–126187. https://doi.org/10.1109/ACCESS.2024.3450388
Ebert, C., Arockiasamy, J.P., Hettich, L., Weyrich, M., 2024. Hints for Generative AI Software Development. IEEE Software 41, 24–33. https://doi.org/10.1109/MS.2024.3410641
Ebert, C., Louridas, P., 2023. Generative AI for Software Practitioners. IEEE Software 40, 30–38. https://doi.org/10.1109/MS.2023.3265877
Eisenhardt, K.M., Graebner, M.E., 2007. Theory Building from Cases: Opportunities and Challenges. The Academy of Management Journal 50, 25–32.
Eisenhardt, K.M., Ott, T.E., 2017. Rigor in Theory Building from Multiple Cases, in: The Routledge Companion to Qualitative Research in Organization Studies. Routledge.
Elbanna, A., Sarker, S., 2016. The Risks of Agile Software Development: Learning from Adopters. IEEE Software 33, 72–79. https://doi.org/10.1109/MS.2015.150
Ernst, N.A., Bavota, G., 2022. AI-Driven Development Is Here: Should You Worry? IEEE Software 39, 106–110. https://doi.org/10.1109/MS.2021.3133805
Fairley, R., 1994. Risk management for software projects. IEEE Software 11, 57–67. https://doi.org/10.1109/52.281716

Fawzy, A., Tahir, A., Galster, M., Liang, P., 2025. Exploring data management challenges and solutions in agile software development: a literature review and practitioner survey. Empir Software Eng 30, 77. https://doi.org/10.1007/s10664-025-10630-4
Feldt, R., de Oliveira Neto, F.G., Torkar, R., 2018. Ways of applying artificial intelligence in software engineering, in: Proceedings of the 6th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering, RAISE '18. Association for Computing Machinery, New York, NY, USA, pp. 35–41. https://doi.org/10.1145/3194104.3194109
Flyvbjerg, B., 2006. Five Misunderstandings About Case-Study Research. Qualitative Inquiry 12, 219–245. https://doi.org/10.1177/1077800405284363
Flyvbjerg, B., Budzier, A., Lee, J.S., Keil, M., Lunn, D., Bester, D.W., 2022. The Empirical Reality of IT Project Cost Overruns: Discovering A Power-Law Distribution. Journal of Management Information Systems 39, 607–639. https://doi.org/10.1080/07421222.2022.2096544
Ghobadi, S., Mathiassen, L., 2014. Perceived barriers to effective knowledge sharing in agile software teams. Information Systems Journal. https://doi.org/10.1111/isj.12053
Harman, M., 2012. The role of Artificial Intelligence in Software Engineering, in: 2012 First International Workshop on Realizing AI Synergies in Software Engineering (RAISE). Presented at the 2012 First International Workshop on Realizing AI Synergies in Software Engineering (RAISE), pp. 1–6. https://doi.org/10.1109/RAISE.2012.6227961
Hein, D., Persson, J., Nielsen, P., 2025. Explaining Project Risks. A case study of causal mapping in an agile software team. Scandinavian Journal of Information Systems 37. https://doi.org/10.17705/3SJIS/037.07
Imai, S., 2022. Is GitHub copilot a substitute for human pair-programming? an empirical study, in: Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings, ICSE '22. Association for Computing Machinery, New York, NY, USA, pp. 319–321. https://doi.org/10.1145/3510454.3522684
Iversen, J.H., Mathiassen, L., Nielsen, P.A., 2004. Managing Risk in Software Process Improvement: An Action Research Approach. MIS Quarterly 28, 395–433. https://doi.org/10.2307/25148645
Jaworski, M., Piotrkowski, D., 2023. Study of software developers' experience using the Github Copilot Tool in the software development process. https://doi.org/10.48550/arXiv.2301.04991
Jensen, V.V., Alami, A., Bruun, A.R., Persson, J.S., 2025. Managing expectations towards AI tools for software development: a multiple-case study. Inf Syst E-Bus Manage 23, 869–901. https://doi.org/10.1007/s10257-025-00704-7
Knearem, T., Khwaja, M., Gao, Y., Bentley, F., Kliman-Silver, C.E., 2023. Exploring the future of design tooling: The role of artificial intelligence in tools for user experience professionals, in: Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems. Presented at the CHI '23: CHI Conference on Human Factors in Computing Systems, ACM, Hamburg Germany, pp. 1–6. https://doi.org/10.1145/3544549.3573874
Kotti, Z., Galanopoulou, R., Spinellis, D., 2023. Machine Learning for Software Engineering: A Tertiary Study. ACM Comput. Surv. 55, 256:1-256:39. https://doi.org/10.1145/3572905
Kou, B., Chen, S., Wang, Z., Ma, L., Zhang, T., 2024. Do Large Language Models Pay Similar Attention Like Human Programmers When Generating Code? Proc. ACM Softw. Eng. 1, 100:2261-100:2284. https://doi.org/10.1145/3660807
Kula, E., Greuter, E., van Deursen, A., Gousios, G., 2022. Factors Affecting On-Time Delivery in Large-Scale Agile Software Development. IEEE Transactions on Software Engineering 48, 3573–3592. https://doi.org/10.1109/TSE.2021.3101192
Laukkanen, M., 1998. Conducting Causal Mapping Research: Opportunities and Challenges. Managerial and organizational cognition.
Laukkanen, M., 1994. Comparative Cause Mapping of Organizational Cognitions. Organization Science 5, 322–343.
Laukkanen, M., Eriksson, P., 2013. New designs and software for cognitive causal mapping. Qualitative Research in Organizations and Management 8, 122–147. http://dx.doi.org/10.1108/QROM-08-2011-1003
Leonhardt Kjærgaard, A., Blegind Jensen, T., 2014. Using Cognitive Mapping to Represent and Share Users' Interpretations of Technology. CAIS 34. https://doi.org/10.17705/1CAIS.03457

Li, J., Zhang, Y., Karas, Z., McMillan, C., Leach, K., Huang, Y., 2024. Do Machines and Humans Focus on Similar Code? Exploring Explainability of Large Language Models in Code Summarization, in: Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension. Presented at the ICPC '24: 32nd IEEE/ACM International Conference on Program Comprehension, ACM, Lisbon Portugal, pp. 47–51. https://doi.org/10.1145/3643916.3644434

Liu, Y., Le-Cong, T., Widyasari, R., Tantithamthavorn, C., Li, L., Le, X.-B.D., Lo, D., 2024. Refining ChatGPT-Generated Code: Characterizing and Mitigating Code Quality Issues. ACM Trans. Softw. Eng. Methodol. 33, 116:1-116:26. https://doi.org/10.1145/3643674

Lockey, S., Gillespie, N., Holm, D., Someh, I.A., 2021. A Review of Trust in Artificial Intelligence: Challenges, Vulnerabilities and Future Directions.

Lopes, S., Souza, R., Contessoto, A., Oliveira, A., 2021. A Risk Management Framework for Scrum Projects. International Conference on Enterprise Information Systems (ICEIS 2021), pp. 30–40. https://doi.org/10.5220/0010448300300040

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

Markus, M., Rowe, F., 2018. Is IT Changing the World? Conceptions of Causality for Information Systems Theorizing.

Masso, J., García, F., Pardo, C., Pino, F.J., Piattini, M., 2022. A Common Terminology for Software Risk Management. ACM Trans. Softw. Eng. Methodol. 31, 1–47. https://doi.org/10.1145/3498539

Moradi Dakhel, A., Majdinasab, V., Nikanjam, A., Khomh, F., Desmarais, M.C., Jiang, Z.M. (Jack), 2023. GitHub Copilot AI pair programmer: Asset or Liability? Journal of Systems and Software 203, 111734. https://doi.org/10.1016/j.jss.2023.111734

Muntés-Mulero, V., Ripolles, O., Gupta, S., Dominiak, J., Willeke, E., Matthews, P., Somosköi, B., 2019. Agile risk management for multi-cloud software development. IET Software 13, 172–181. https://doi.org/10.1049/iet-sen.2018.5295

Nguyen, S., Babe, H.M., Zi, Y., Guha, A., Anderson, C.J., Feldman, M.Q., 2024. How Beginning Programmers and Code LLMs (Mis)read Each Other, in: Proceedings of the CHI Conference on Human Factors in Computing Systems. Presented at the CHI '24: CHI Conference on Human Factors in Computing Systems, ACM, Honolulu HI USA, pp. 1–26. https://doi.org/10.1145/3613904.3642706

Nguyen-Duc, A., Cabrero-Daniel, B., Przybylek, A., Arora, C., Khanna, D., Herda, T., Rafiq, U., Melegati, J., Guerra, E., Kemell, K.-K., Saari, M., Zhang, Z., Le, H., Quan, T., Abrahamsson, P., 2023. Generative Artificial Intelligence for Software Engineering -- A Research Agenda. https://doi.org/10.48550/arXiv.2310.18648

Odzaly, E.E., Greer, D., Stewart, D., 2018. Agile risk management using software agents. J Ambient Intell Human Comput 9, 823–841. https://doi.org/10.1007/s12652-017-0488-2

Peng, S., Kalliamvakou, E., Cihon, P., Demirer, M., 2023. The Impact of AI on Developer Productivity: Evidence from GitHub Copilot. https://doi.org/10.48550/arXiv.2302.06590

Persson, J., Schlichter, B., 2015. Managing Risk Areas in Software Development Offshoring: A CMMI Level 5 Case. Journal of Information Technology Theory and Application (JITTA) 16.

Pfleeger, S.L., 2000. Risky business: what we have yet to learn about risk management. Journal of Systems and Software 53, 265–273. https://doi.org/10.1016/S0164-1212(00)00017-0

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

Rasnayaka, S., Wang, G., Shariffdeen, R., Iyer, G.N., 2024. An Empirical Study on Usage and Perceptions of LLMs in a Software Engineering Project. https://doi.org/10.48550/arXiv.2401.16186

Rique, T., Perkusich, M., Gorgônio, K., Almeida, H., Perkusich, A., 2025. Constructing the graphical structure of expert-based Bayesian networks in the context of software engineering: A systematic mapping study. Information and Software Technology 177, 107586. https://doi.org/10.1016/j.infsof.2024.107586

Runeson, P., Höst, M., 2009. H??st, M.: Guidelines for Conducting and Reporting Case Study Research in Software Engineering. Empirical Software Engineering 14, 131-164. Empirical Software Engineering 14, 131–164. https://doi.org/10.1007/s10664-008-9102-8
Runeson, P., Host, M., Rainer, A., Regnell, B., 2012. Case Study Research in Software Engineering: Guidelines and Examples. John Wiley & Sons.
Russo, D., 2024. Navigating the Complexity of Generative AI Adoption in Software Engineering. ACM Trans. Softw. Eng. Methodol. 33, 135:1-135:50. https://doi.org/10.1145/3652154
Russo, D., Stol, K.-J., 2021. PLS-SEM for Software Engineering Research: An Introduction and Survey. ACM Comput. Surv. 54, 78:1-78:38. https://doi.org/10.1145/3447580
Schmidt, C., Dart, P., Johnston, L., Sterling, L., Thorne, P., 1999. Disincentives for communicating risk: a risk paradox. Information and Software Technology 41, 403–411. https://doi.org/10.1016/S0950-5849(99)00011-7
Sergeyuk, A., Golubev, Y., Bryksin, T., Ahmed, I., 2025. Using AI-based coding assistants in practice: State of affairs, perceptions, and ways forward. Information and Software Technology 178, 107610. https://doi.org/10.1016/j.infsof.2024.107610
Shrivastava, S.V., Rathod, U., 2017. A risk management framework for distributed agile projects. Information and Software Technology 85, 1–15. https://doi.org/10.1016/j.infsof.2016.12.005
Sommerville, I., 2016. Software engineering, 10. ed., global ed. ed, Always learning. Pearson, Boston Munich.
Stige, Å., Zamani, E.D., Mikalef, P., Zhu, Y., 2023. Artificial intelligence (AI) for user experience (UX) design: a systematic literature review and future research agenda. Information Technology & People 37, 2324–2352. https://doi.org/10.1108/ITP-07-2022-0519
Tambon, F., Moradi-Dakhel, A., Nikanjam, A., Khomh, F., Desmarais, M.C., Antoniol, G., 2025. Bugs in large language models generated code: an empirical study. Empir Software Eng 30, 65. https://doi.org/10.1007/s10664-025-10614-4
Tavares, B.G., da Silva, C.E.S., de Souza, A.D., 2019. Risk management analysis in Scrum software projects. International Transactions in Operational Research 26, 1884–1905. https://doi.org/10.1111/itor.12401
Tian, H., Lu, W., Li, T.O., Tang, X., Cheung, S.-C., Klein, J., Bissyandé, T.F., 2023. Is ChatGPT the Ultimate Programming Assistant -- How far is it? https://doi.org/10.48550/arXiv.2304.11938
Tsintzira, A.-A., Arvanitou, E.-M., Ampatzoglou, A., Chatzigeorgiou, A., 2020. Applying Machine Learning in Technical Debt Management: Future Opportunities and Challenges, in: Shepperd, M., Brito e Abreu, F., Rodrigues da Silva, A., Pérez-Castillo, R. (Eds.), Quality of Information and Communications Technology. Springer International Publishing, Cham, pp. 53–67. https://doi.org/10.1007/978-3-030-58793-2_5
Wang, S., Huang, L., Gao, A., Ge, J., Zhang, T., Feng, H., Satyarth, I., Li, M., Zhang, H., Ng, V., 2023. Machine/Deep Learning for Software Engineering: A Systematic Literature Review. IEEE Transactions on Software Engineering 49, 1188–1231. https://doi.org/10.1109/TSE.2022.3173346
Weber, T., Brandmaier, M., Schmidt, A., Mayer, S., 2024. Significant Productivity Gains through Programming with Large Language Models. Proc. ACM Hum.-Comput. Interact. 8, 256:1-256:29. https://doi.org/10.1145/3661145
Williams, R.C., Walker, J.A., Dorofee, A.J., 1997. Putting risk management into practice. IEEE Software 14, 75–82. https://doi.org/10.1109/52.589240
Yetistiren, B., Ozsoy, I., Tuzun, E., 2022. Assessing the quality of GitHub copilot's code generation, in: Proceedings of the 18th International Conference on Predictive Models and Data Analytics in Software Engineering. Presented at the PROMISE '22: 18th International Conference on Predictive Models and Data Analytics in Software Engineering, ACM, Singapore Singapore, pp. 62–71. https://doi.org/10.1145/3558489.3559072
Yin, R.K., 2018. Case study research and applications: design and methods, Sixth edition. ed. SAGE, Los Angeles.
Yu, C.-P., Chen, H.-G., Klein, G., Jiang, R., 2015. The roots of executive information system development risks. Information and Software Technology 68, 34–44. https://doi.org/10.1016/j.infsof.2015.08.001

Zhang, B., Liang, P., Zhou, X., Ahmad, A., Waseem, M., 2023. Practices and Challenges of Using GitHub Copilot: An Empirical Study. Presented at the The 35th International Conference on Software Engineering and Knowledge Engineering, pp. 124–129. https://doi.org/10.18293/SEKE2023-077

Zhou, X., Liang, P., Zhang, B., Li, Z., Ahmad, A., Shahin, M., Waseem, M., 2025. Exploring the problems, their causes and solutions of AI pair programming: A study on GitHub and Stack Overflow. Journal of Systems and Software 219, 112204. https://doi.org/10.1016/j.jss.2024.112204

Appendix A: Sample Interview Guide

- Could you describe your role in the organization?
- How long have you been working with software development in this organization?
- Are you currently using any generative AI tools (e.g., ChatGPT, GitHub Copilot)? If so, which ones, and in what capacity?
- From your perspective, what are the main benefits of using generative AI in software development?
- What concerns or risks do you associate with using generative AI tools?
    - Why do you see that as a risk?
- Are there particular situations where using generative AI might cause problems?
    - Why?
- Have you encountered any unintended outcomes when using generative AI (e.g., incorrect output, security concerns, legal issues)?
    - Do you know why it happened?
- Are there risks that impact certain parts of your workflow or organization more than others (e.g., security, productivity, collaboration)?
    - Why?

Appendix B: Validation Interview Guide

- Do you have any questions about the causal map?
- Do the risks in the map explain your concerns?
- Are there any risks missing in the causal map?
- Are there any risks that should not be in the map?
- What are the most important risks for you?
- Do you think the arrows between the risks explain the cohesion among the
- risks in the map?
- Are there any arrows missing among any of the risks?
- Are there any arrows that you think should be added?

Appendix C: Example of Data Collection and Coding Structure

Example data collection and coding structure

To support the reproducibility of the causal mapping approach, researchers may structure the data collection and coding process using a systematic template that links interview statements to risk concepts and causal relationships. We have used the software tool Draw.io to construct the causal maps; similar tools, such as Kumu, Figma, or Vensim, can also be used, while NVivo can be used to create the coding schemes that constitute the data for the causal maps. For other specific examples of

using and presenting causal maps, see these works (Ackermann et al., 2014; Ackermann and Eden, 2020; Ghobadi and Mathiassen, 2014; Leonhardt Kjærgaard and Blegind Jensen, 2014) and see these for descriptions of using the method (Laukkanen, 1998, 1994; Laukkanen and Eriksson, 2013). Table C1 illustrates an example structure for organizing interview data, codes, concept groupings, and inferred causal links during analysis.

The procedure may be conducted in four primary steps:

Step 1: Identify risk statements
Interview transcripts are reviewed to identify statements describing potential risks associated with the use of generative AI in software development. Each statement is assigned a preliminary descriptive code capturing the essence of the perceived risk.

Step 2: Develop a codebook and group concepts
Codes describing similar phenomena are grouped into broader concepts that represent risks within the organization. During this step, researchers may iteratively refine the codebook by merging similar codes, clarifying definitions, and aligning concepts across cases.

Step 3: Identify causal relationships between risks
Causal relationships are identified when participants explicitly or implicitly describe one risk leading to another. Linguistic cues such as *because*, *due to*, *therefore*, or *as a result* often indicate perceived causality. When such relationships are identified, the relevant concepts are connected through directional causal links.

Step 4: Document supporting participants and cases
To improve transparency and traceability, researchers may document which participants or case organizations referenced the risk or causal relationship. This enables cross-case comparison and helps assess whether relationships are shared across organizations or specific to individual cases.

The following table illustrates how interview data may be organized during analysis,

Interview statement	Initial code	Risk concept	Causal link	Participant(s)	Case organization(s)
"I have no idea what it bases its knowledge on or how it stores the data you sent it to. It's somewhat hidden in terms of ChatGPT." / "It's a black box how it works in the machine room."	Uncertainty about how AI handles data	Inadequate data management of AI	→ Mistrust of AI	P4, P9	Bankers, Insurers
"The use of large language models introduces a security risk because we are not entirely in control of what it sends elsewhere." / "The most critical aspect is data security and avoiding leaks of our source code."	Concern about external data exposure	Inadequate data management of AI	→ Lack of control over AI	P2, P16	Bankers, Automaters

Interview statement	Initial code	Risk concept	Causal link	Participant(s)	Case organization(s)
"It may be beneficial to use, but it also presents a threat… we must validate the code very thoroughly before we use it." / "Some of this AI-generated code can be really good, but it can also be of very poor quality."	Need to review AI-generated code	Insufficient validation of AI output	→ Insufficient/insecure AI output	P2, P9	Bankers, Insurers
"That would make it more trustworthy if you were guaranteed to get the right answer. It's something that undermines my trust in it." / Developers explaining the need to validate AI outputs before use.	Lack of trust in AI outputs	Mistrust of AI	→ Insufficient validation of AI output	P12, P5	Insurers, Bankers
"I am not very experienced in using ChatGPT… It's unclear to me how I use it properly." / "If you don't understand it, then you might get incorrect results."	Lack of experience using AI tools	Lacking AI competencies	→ Insufficient/insecure AI output	P6, P16	Bankers, Automaters
"It makes sense to use AI for simple tasks… where things become complex it is not much help." / "I would be hesitant to use it for something related to insurance as it does not understand our business."	AI limitations in complex tasks	Inappropriate for complex/industry-specific tasks	← Insufficient/insecure AI output	P17, P10	Automaters, Insurers

Table C1: Overview of coding scheme

Table C1 illustrates how interview statements were translated into initial codes and aggregated into risk concepts during the analysis. The examples are drawn directly from the findings section and our coding schemes. They demonstrate how similar concerns expressed by participants across the three case organizations were coded and linked to causal relationships in the maps. Including participant identifiers and case organizations enhances traceability and supports cross-case comparison during causal map construction.

Appendix D: Extended Guidelines for Constructing Causal Maps

Best Practices for Participant Validation of Causal Maps

Participant validation is an important step when applying causal mapping in organizational studies, as it helps ensure that the causal relationships identified by researchers accurately reflect the explanations participants express. Based on our experience in this study, we outline several practices for conducting validation sessions with participants. See Appendix B for the semi-structured interview guide for an example of structuring the validation session.

First, validation sessions should be structured as interactive discussions rather than confirmation exercises with 2-4 participants from each organization. Participants should be presented with the causal maps derived from the initial interviews, and the interviewer should begin by explaining the rationale for the identified risks and relationships. The participants should be asked to reflect on whether the identified risks and causal relationships correspond with their experiences. Rather than simply asking participants to approve the maps, researchers should encourage participants to explain whether the relationships appear plausible, incomplete, or misleading.

Second, researchers should actively probe for clarification when reviewing causal relationships. For instance, participants can be asked whether a specific risk truly causes another risk, whether the direction of the relationship is correct, or whether an alternative explanation may exist. This probing helps identify ambiguous or weakly supported relationships and may reveal additional causal links that were not explicitly captured in the original interviews.

Third, participant feedback should be systematically integrated into the analysis. When participants suggest modifications, researchers should revisit the original interview transcripts to determine whether the proposed changes align with previously expressed explanations. This step helps maintain traceability between the causal maps and the underlying data.

Finally, when participant feedback diverges, for example, when some participants disagree with a particular causal relationship, researchers should treat such differences as valuable insights rather than inconsistencies that must be eliminated. Diverging views may indicate that risks are interpreted differently across roles or organizational contexts. In such cases, researchers should evaluate whether the causal relationship reflects a plausible explanation grounded in the interview data. If so, the relationship can be retained while acknowledging that participants' perceptions of risk may vary. In our case, we did not find any meaningful differences between participants or roles inside the case organizations, only between the organizations.

Addressing Ambiguous or Conflicting Causal Relationships

When constructing the causal maps, some causal relationships between risks were not immediately clear or could be interpreted in multiple ways. To address such cases and ensure consistent interpretation, we followed a consensus-building procedure.

First, the first author initially inferred causal relationships from interview statements indicating cause–and–effect reasoning (e.g., through phrases such as “because”, “due to”, or “as a result of”). These preliminary causal links were then reviewed by the second and third authors, who examined whether the interpretation was consistent with the participants’ statements and the surrounding interview context.

Second, when disagreements or ambiguity arose, the researchers revisited the original interview transcripts and quotations to reassess the intended meaning of the participant's explanation. Causal links were only retained when the researchers agreed that the relationship reflected the participant's reasoning. Disagreement and ambiguity were noted in the codebook to ensure they were raised during validation of the causal maps with representatives from the case organizations.

Third, the resulting causal maps were validated with representatives from the case organizations during follow-up interviews and a cross-case workshop. This process typically led to only small adjustments to the causal maps (e.g., adding a missing causal relationship or removing a risk that participants considered irrelevant).

Saturation and Selection of Risks for the Causal Maps

The purpose of the causal mapping analysis was to capture how the case organizations explained the causes of risks associated with generative AI rather than to measure the frequency of specific risks. Consequently, risks were not selected based on quantitative thresholds or frequency counts.

In the interview analysis, risks were identified whenever participants described a potential negative outcome related to the use of generative AI that aligned with the definition of risk as the "possibility of loss." The coding process continued across the interviews until no substantially new risk concepts emerged from additional transcripts, indicating thematic saturation within the cases.

All risks identified during the coding process were initially included in the causal maps for each organization. Risks were then linked when participants described a causal relationship between them (e.g., through statements indicating that one risk occurred "because of" another).

Rather than prioritizing risks based on frequency of mention, we focused on capturing the causal reasoning expressed by the participants. Therefore, risks were retained in the maps even if they were mentioned by a small number of participants, provided they formed part of a coherent causal explanation.

During the validation interviews and cross-case workshop, representatives from the case organizations reviewed the causal maps and assessed whether the identified risks and causal relationships accurately reflected their experiences.