

Information Systems Development in the Age of Artificial Intelligence



Inauguraldissertation
zur Erlangung des Doktorgrades
der Wirtschafts- und Sozialwissenschaftlichen Fakultät der
Universität zu Köln

2025

vorgelegt von
Mei Wu-Gehbauer
aus Jinan

Referent: Prof. Dr. Christoph Rosenkranz

Korreferent: Prof. Dr. Stefan Seidel

Tag der Promotion: 04.03.2026

*With love,
for my parents, my husband, and my children.*

Contents

List of Figures	VII
List of Tables	VIII
Summary	X
Introductory Paper	1
1 Introduction	2
1.1 Motivation	2
1.2 Research Questions	4
1.3 Dissertation Structure	10
2 Theoretical Background	11
2.1 Information Systems Development	11
2.1.1 ISD Stakeholders	11
2.1.2 ISD Processes	12
2.1.3 ISD Outputs	13
2.2 Artificial Intelligence-based Systems as Outputs of Information Systems Development	14
2.2.1 Shifting Nature in ISD for AI-based Systems	14
2.2.2 Discriminative and Generative AI-based Systems	15
2.3 Artificial Intelligence-augmented Information Systems Development	17
2.4 Theoretical Lenses	18
2.4.1 Knowledge Boundary	19
2.4.2 Cognitive Fit	20
2.4.3 Delegation	21
2.4.4 Job Crafting	23
3 Research Methodology	25
3.1 Qualitative Research	26
3.1.1 Case Study Research Design	26
3.1.2 Data Sources and Collection	29
3.1.3 Data Analysis and Rigor	31
3.2 Quantitative Research	33
3.2.1 Survey Research Design	33
3.2.2 Measurement, Validity, and Analysis	34
3.2.3 Hypothesis Testing	36
4 Main Research Results	37

4.1	Statement of Individual Contributions and Co-Author Collaboration	37
4.2	Perspective I: Development of Artificial Intelligence-based Systems	39
4.2.1	Paper I: Developing Artificial Intelligence Systems – Bridging Knowledge Boundaries in the Triad	39
4.2.2	Paper II: Towards a Framework for Developing Artificial Intelligence Systems for Human-Artificial Intelligence Augmentation	44
4.2.3	Paper III: Between Enablement and Control – Generative Artificial Intelligence-based Systems Development	46
4.2.4	Paper IV: Delegation in Developing Artificial Intelligence-based Systems – How Developers Construct Conditions for User-Artificial Intelligence Delegation	47
4.3	Perspective II: AI-augmented Information Systems Development	52
4.3.1	Paper V: Unlocking the Potential of Generative Artificial Intelligence – A Case Study in Software Development	52
4.3.2	Paper VI: Developers’ Job Crafting and Individual Work Outcomes in Artificial Intelligence-augmented Information Systems Development	54
5	Discussion	57
5.1	Theoretical Implications	57
5.1.1	Development of AI-based Systems from a Triadic Perspective	57
5.1.2	Development of AI-based Systems for Joint Human-AI Problem-Solving	59
5.1.3	Development of AI-based Systems from a Delegation Perspective	61
5.1.4	Implication of AI-augmented ISD for ISD Processes	63
5.1.5	Implication of AI-augmented ISD for Developers’ Individual Work Outcomes and Job Crafting	64
5.2	Practical Implications	66
5.2.1	Orchestrating AI-based Systems Development in the Triad	67
5.2.2	Designing Delegation in AI-based Systems Development	68
5.2.3	Shaping AI-augmented ISD	69
5.2.4	Assessing the Impact of AI-augmented ISD	70
6	Limitations and Future Research	70
6.1	Limitations	70
6.2	Future Research	72
7	Conclusion	74
	References	76
	Paper I: Developing AI-based Systems – Bridging Knowledge Boundaries in the Triad	92

Paper II: Towards a Framework for Developing Artificial Intelligence Systems for Human-Artificial Intelligence Augmentation	116
Paper III: Between Enablement and Control – Generative Artificial Intelligence-based Systems Development	142
Paper IV: Delegation in Developing Artificial Intelligence-based Systems – How Developers Construct Conditions for User-Artificial Intelligence Delegation	162
Paper V: Unlocking the Potential of Generative Artificial Intelligence – A Case Study in Software Development	209
Paper VI: Developers’ Job Crafting and Individual Work Outcomes in Artificial Intelligence-augmented Information Systems Development	237
Complete References	273

List of Figures

Figure 1:	Research focus for the two perspectives of the dissertation . . .	4
Figure 2:	Overall structure of the cumulative dissertation	10
Figure 3:	Overview of the research approaches, methods, and designs in this dissertation	26
Figure 4:	Framework for the development of AI-based systems for human- AI problem-solving	45
Figure 5:	Model of two delegation cycles in the development of AI- based systems: Delegation-in-development & delegation-in-use	49
Paper II – Figure 1:	Excerpt of coding scheme	127
Paper II – Figure 2:	Framework for developing AI systems for human-AI augmen- tation (integrated and extended based on Baird and Maruping (2021) and Vessey and Galletta (1991)).	135
Paper III – Figure 1:	Framework for developing GenAI-based systems	157
Paper IV – Figure 1:	Data structure (analysis based on Strauss and Corbin (1998); data structure based on Gioia et al. (2013))	177
Paper IV – Figure 2:	Model of two delegation cycles in the development of AI- based systems: Delegation-in-development & delegation-in-use	195
Paper V – Figure 1:	Representational perspective on GenAI-assisted software de- velopment	214
Paper V – Figure 2:	Action possibilities for GenAI-assisted software development	217
Paper V – Figure 3:	Realizing action possibilities of GenAI-assisted software de- velopment	227
Paper VI – Figure 1:	Research model	247

List of Tables

Table 1:	Discriminative and LLM-based generative AI-based systems characterized by their agentic attributes (adapted from Baird and Maruping (2021))	16
Table 2:	Overview of qualitative case studies (Paper I-V)	28
Table 3:	Overview of the three-stage coding approach across Paper I to V	32
Table 4:	Overview of papers included in this dissertation	38
Table 5:	Knowledge-related boundaries along key phases of AI-based systems development	42
Table 6:	Development practices and boundary objects in AI-based systems development	43
Table 7:	Delegation-in-development per developers' temporal orientation: summary of concepts and definitions	50
Table 8:	Construction of user-AI delegation per developers' temporal orientation: summary of concepts and definitions	51
Table 9:	Coefficients for PLS-SEM	56
Paper I – Table 1:	Interview data	99
Paper I – Table 2:	Overview of AI systems: Decision making prior to versus with AI	100
Paper I – Table 3:	Knowledge-related boundaries along key phases of AI systems development	104
Paper I – Table 4:	Development practices and boundary objects	108
Paper II – Table 1:	Interview data	124
Paper II – Table 2:	Overview of AI system development projects	125
Paper III – Table 1:	Data sources	148
Paper III – Table 2:	GenAI systems development projects	149
Paper III – Table 3:	Summary of mechanisms	155
Paper IV – Table 1:	Overview of discriminative and LLM-based generative AI systems along the agentic attributes (as specified by Baird and Maruping (2021))	172
Paper IV – Table 2:	Overview of data	175
Paper IV – Table 3:	Delegation-in-development per developers' temporal orientation: summary of concepts and definitions	197
Paper IV – Table 4:	Construction of user-AI delegation per developers' temporal orientation: summary of concepts and definitions	198
Paper V – Table 1:	Interview data	219
Paper V – Table 2:	Overview of alignment mechanisms	224

Paper VI – Table 1:	Construct reliability and validity of reflective measures . . .	251
Paper VI – Table 2:	Means, standard deviation, and correlations	253
Paper VI – Table 3:	Coefficients for PLS-SEM	254
Paper VI – Table 4:	Bootstrapped results and confidence intervals of estimated paths of direct effect and mediation	255
Paper VI – Table 5:	Bootstrapped results and confidence intervals of additional es- timated paths	256
Paper VI – Table 6:	Measurement items	268

Summary

Information systems (IS) in organizations increasingly embed artificial intelligence, which exhibits agentic characteristics as it perceives its environment, responds, and adapts its behavior. While a substantial IS literature examines how work is accomplished with artificial intelligence (AI), the implications of AI for information systems development (ISD) remain underexplored. Yet, AI raises more than technological challenges in ISD: It has broader implications for the humans involved and the development activities undertaken to create a working IS comprising people, processes, and technology. Thus, the overarching inquiry of this dissertation is: *How does artificial intelligence reshape information systems development in organizations?*

To address this inquiry, this cumulative dissertation provides a comprehensive account of how AI reshapes ISD by examining two perspectives in which AI enters ISD. First, it examines AI as an ISD output in the form of agentic AI-based systems, foregrounding the implications of agentic system characteristics for development practices and for the humans involved. Second, it examines AI as a technology that augments developers' work within ISD processes, focusing on its implications for ISD outcomes and for the developers' jobs.

The dissertation offers novel theoretical and empirical insights into ISD in the age of AI. For the development of AI-based systems, it shows that ISD must move beyond the traditional user-developer dyad towards a triadic view that treats the AI-based system as an agent in the development process. The dissertation theorizes how development work in the triad bridges knowledge-related boundaries, iteratively aligns AI-based systems for collaborative human-AI problem-solving, and involves developers entering a delegation relationship with the AI-based system during development. These insights help organizations understand how development work must be orchestrated across developers, users, and the AI-based system. For AI-augmented ISD, the dissertation shows how generative AI reshapes ISD processes through additional development practices that enable developers to realize and shape generative AI-augmented ways of working. While generative AI use is positively associated with developers' work engagement, no evidence was found for an association with broader, proactive changes to developers' jobs. Overall, this dissertation provides a comprehensive understanding of how AI reshapes ISD across both the development of AI-based systems and AI-augmented ISD.

Keywords: Information systems development, artificial intelligence, generative artificial intelligence, artificial intelligence-based systems, development practices, triadic perspective, AI-augmented information systems development

Introductory Paper

Information Systems Development in the Age of Artificial Intelligence

Mei Wu-Gehbauer
University of Cologne

Introductory Paper

1 Introduction

1.1 Motivation

Information systems (IS) in organizations increasingly embed artificial intelligence (AI) to augment and automate work well beyond the remit of traditional rule-based information technology (IT) (e.g., Benbya et al., 2021; Kemp, 2024; Murray et al., 2021; Raisch & Fomina, 2025). The breadth of AI deployments in organizations across decision-making, customer interaction, product development, and operations positions AI as a pervasive technology that is reshaping work and organizing (Brynjolfsson et al., 2023; Deng et al., 2023; Faraj et al., 2018; Lebovitz et al., 2022; Seidel et al., 2019)¹.

AI technologies are goal-directed agents that perceive their environment, act in response, and adapt their behavior to change (Russell & Norvig, 2016). Compared with deterministic technologies, contemporary AI exhibits several agentic characteristics that set it apart: autonomy (the ability to initiate actions without step-by-step human specification), learning (the ability to learn and update its decision model from data and feedback), and opacity (lack of visibility of how inputs are transformed into decisions and outputs) (Berente et al., 2021; Burrell, 2016).

A substantial IS literature contributes to our understanding of how work and decision-making are accomplished using AI (e.g., Abdel-Karim et al., 2023; Jussupow et al., 2021; Lebovitz et al., 2021; Lou & Wu, 2021; Zhang et al., 2021). By contrast, the implications of AI for ISD are underexplored. ISD encompasses all development activities (e.g., planning, analysis, design, building, testing, and maintenance) carried out by agents to create a working IS comprising people, processes, and technology (Bostrom & Heinen, 1977a, 1977b; Matook et al., 2021; Sarker et al., 2019). Much of the work on developing AI-based systems has been driven by the computer science and software engineering literature, with an emphasis on the technology (e.g., Amershi et al., 2019a; Baylor et al., 2017; Breck et al., 2017; Paleyes et al., 2022; Polyzotis et al., 2018; Sculley et al., 2015). Yet, AI raises more than technological challenges in ISD. It has broader implications for ISD, including all agents involved and the development activities undertaken.

First, AI has implications for how this new generation of IS is developed across its lifecycle (Berente et al., 2021). One shift is the increased salience of domain expertise and human-in-the-loop configurations that combine machine learning with expert judgment during development (Fügener et al., 2022; Gronsund & Aanestad, 2020; van den Broek et al., 2021). Another shift is the importance of managing opacity and enabling oversight of AI (Asatiani et al., 2021). Under-

¹In accordance with the rules of the Faculty of Management, Economics, and Social Sciences of the University of Cologne regarding self-quotation in the introduction of a cumulative dissertation, parts of this introductory paper have been adopted from the research articles

standing how ISD evolves in the development of AI-based systems is critical because it shapes whether these systems align with human goals and values and support human accountability (e.g., Asatiani et al., 2021; Teodorescu et al., 2021).

Second, generative AI (GenAI) is increasingly used to augment software development work within ISD. Because generative AI can generate content previously produced by developers themselves, such as code, tests, and documentation, it has sparked debate about whether it may disrupt developers' work practices and role boundaries (Benbya et al., 2024; Nguyen-Duc et al., 2023; Sætra, 2023). The empirical evidence on the implications of AI-augmented ISD remains inconclusive. On the one hand, studies report productivity gains for developers using generative AI (Cui et al., 2024; Ziegler et al., 2024). On the other hand, evidence also points to adverse individual-level effects, such as reduced critical thinking, decreased motivation, and technostress (Chuang et al., 2025; Gerlich, 2025; Wu et al., 2025).

Despite its relevance, our understanding of how ISD is reshaped by AI remains limited. Recent work explicitly calls for research at the intersection of ISD and AI, underscoring this gap (Baird & Maruping, 2021; Benbya et al., 2020; Rosenkranz et al., 2025). Accordingly, this dissertation poses the following overarching research question:

How does artificial intelligence reshape information systems development in organizations?

This question is examined from two perspectives:

- I. how ISD is reshaped when developing AI-based systems, and
- II. how ISD is reshaped when augmenting development work in ISD processes with AI.

In addressing the core research question from these two perspectives, the dissertation comprises this introductory paper and six research papers. Across both perspectives, this dissertation predominantly adopts a practice-oriented, sociotechnical view of ISD, examining how development work is accomplished through situated development practices in organizational settings (Nicolini, 2012; Sarker et al., 2019). The analysis focuses on individual-level agents (humans and AI) involved in ISD within organizations and investigates how they influence, and are in turn influenced by the ISD practices through which development unfolds (referred to as development practices hereafter).

In structuring the research focus of Perspective I and Perspective II, this dissertation draws on the core research themes in ISD: ISD stakeholders, ISD processes, and ISD outputs (Matook et al., 2021). ISD stakeholders are human agents involved in the ISD processes, while ISD processes are planned and ad hoc activities conducted to create ISD outputs. ISD outputs are outcomes of the ISD processes that include systems characteristics as well as ISD performance.

In Perspective I, AI is considered as the ISD output in the form of an AI-based system. An AI-based system is an agentic IS artifact that applies AI to enable or support tasks within orga-

nizational task structures and contexts (Baird & Maruping, 2021; Benbasat & Zmud, 2003). The focus of the papers in Perspective I is on the implications of AI's agentic systems characteristics for the development practices and for the developers and users involved in the development of the AI-based system.

In Perspective II, AI is considered as a technology that augments developers' work within ISD processes. AI-augmented ISD refers to ISD processes in which developers collaborate with generative AI to perform development activities (Raisch & Krakowski, 2021). Papers in Perspective II examine how developers carry out the development practices with AI, and how individual developers and ISD outcomes are affected by AI-augmented ISD. Figure 1 provides an overview of the focus for each perspective.

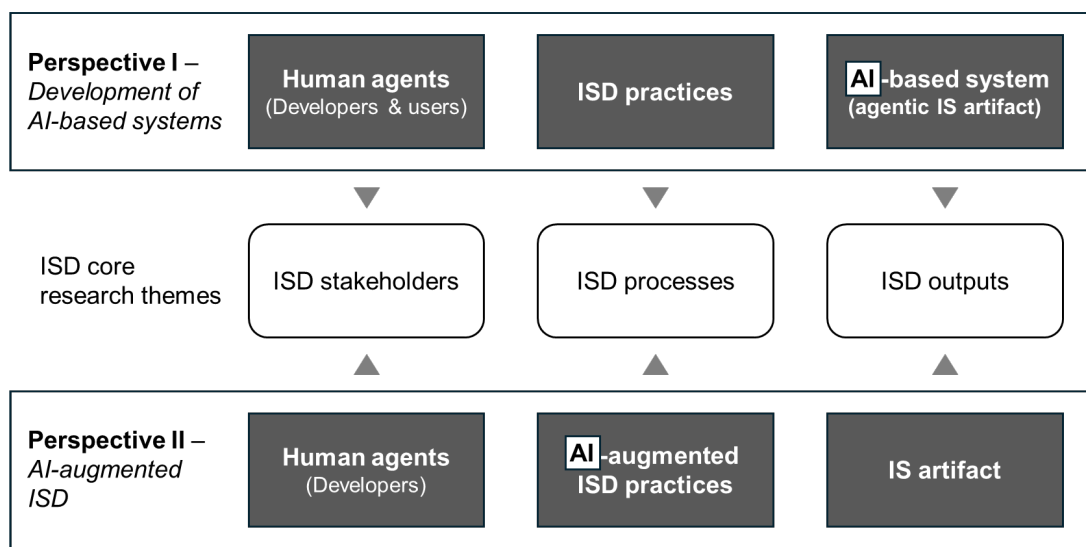


Figure 1: Research focus for the two perspectives of the dissertation

1.2 Research Questions

Building on existing scholarship, this section identifies key research gaps at the intersection of ISD and AI and formulates five detailed research questions. Together, these questions enable an investigation of the **overarching inquiry** “*How does artificial intelligence reshape information systems development in organizations?*” along the two perspectives: development of AI-based systems (Perspective I) and AI-augmented ISD (Perspective II).

Starting with **Perspective I**, developing AI-based systems departs markedly from ISD in the past (Rosenkranz et al., 2025). For decades, ISD has been structured primarily around the dyadic relationship between users and developers (e.g., Chakraborty et al., 2010; Gallivan & Keil, 2003; Newman & Robey, 1992). This relationship was conceptualized as one in which developers elicit requirements and knowledge from users and then encode it into deterministic systems that reliably execute predefined logic (Berente et al., 2021; Bostrom & Heinen, 1977a; Byrd et al., 1992). Consequently, ISD literature has traditionally focused on the user-developer coordination

and the tensions arising from goal incongruencies and information and knowledge asymmetries (e.g., Hirschheim et al., 1991; Rosenkranz et al., 2014). The system itself was largely treated as a rule-based deterministic artifact whose behavior is specified by ISD stakeholders (Berente et al., 2021; Bostrom & Heinen, 1977a; Byrd et al., 1992; Herbst, 1996).

These assumptions no longer hold for AI-based systems. AI-based systems, particularly those relying on machine learning, are no longer developed by specifying explicit rules *ex ante*, nor can their input-output relationships be fully understood *ex post* (Berente et al., 2021; Zhang et al., 2021). Instead, they learn causal relationships from data on their own, and behave in a probabilistic and opaque way (Berente et al., 2021; Burrell, 2016). Unlike rule-based systems, an AI-based system is agentic in its nature and capable of pursuing goal-oriented behavior and acting autonomously without continuous human intervention, making an AI-based system an agent in its own right (Baird & Maruping, 2021; Russell & Norvig, 2016). As a result, ISD shifts from a traditionally dyadic focus to a triadic configuration in which developers, users, and the AI-based system are interdependent agents who together shape development practices and are shaped by them (Ågerfalk et al., 2022). With respect to developers in Perspective I, the focus is on developers with data and machine learning (ML) expertise, such as data scientists, ML engineers, data engineers, who have the core expertise of developing an AI-based system (Amershi et al., 2019a; Kim et al., 2016; Kreuzberger et al., 2023). Users refer to individuals with domain expertise who are either users of AI-based systems or user representatives knowledgeable about the user domain (Laato et al., 2022; Lebovitz et al., 2021; van den Broek et al., 2021).

Accordingly, in Perspective I, this dissertation investigates the development of AI-based systems from a triadic view by recognizing the AI-based system as an own agent. The developer, the user, and the AI-based system are treated as three interdependent agents, each endowed with its own agentic attributes (Baird & Maruping, 2021). A growing IS research stream has begun to examine the implications of introducing such agentic systems into ISD. This work offers important but partial insights into the triad and can be clustered into two sub-streams: (1) studies foregrounding how developers and users align their values, goals, and practices during development, for example, through human-centered design practices such as participatory and inclusive design (e.g., Auernhammer, 2020) or through mutual learning between developers and users who iteratively include and exclude domain expertise (e.g., van den Broek et al., 2021); and (2) studies emphasizing how users are enabled to retain oversight over AI-based systems by shaping the AI-based system (e.g., through sociotechnical envelopment of inscrutable AI-based systems) to support human accountability, control, and user-centered outcomes (e.g., Asatiani et al., 2021; Kane et al., 2021), or how users are enabled in understanding through explainability mechanisms that make model functioning more intelligible in use (e.g., Arrieta et al., 2020). What remains underexplored is how the triad as a whole is jointly shaped during development. Thus, the first research question is:

Research Question 1 (RQ1): How are development practices enacted within the triad of developers, users, and the AI-based system during systems development?

As AI-based systems shift ISD outputs from deterministic, rule-based systems towards agentic systems, this dissertation further examines the implications of developing such agentic systems for users and for developers. Prior work on the implications of AI-based systems on users during development separates two lines. The first line focuses on shaping the AI-based system so that its capabilities remain consistent with user values, goals, tasks, and accountability requirements, for example, by enveloping inscrutable models or through system-level design principles for user-centered and accountable outcomes (e.g., Asatiani et al., 2021; Kane et al., 2021). The second line of work focuses on shaping the human side by building users' understanding of the AI-based system, for example, through explainability approaches that make the system's model functioning transparent and intelligible to users (e.g., Arrieta et al., 2020).

Both lines of work are essential in understanding the development of AI-based systems, yet in practice, they are interdependent: Changes to the AI-based system capabilities, including its goals, guardrails, or outputs, redefine what users must understand and how they can act with the system. Conversely, users' practices and feedback inform what the AI-based system should be capable of (Gronsund & Aanestad, 2020; Waardenburg & Huysman, 2022). Despite this reciprocity, we lack an understanding of an integrated account of how development practices shape both sides together in aligning the AI-based systems with joint human-AI problem-solving tasks in organizations. Consequently, the second research question asks:

Research Question 2 (RQ2): How do developers enact development practices to align AI-based systems with joint human-AI problem-solving tasks, and how do they thereby shape users' understanding of the system?

From a developer's viewpoint, existing work mainly conceptualizes the AI-based system as an artifact within a development workflow, emphasizing development stages, quality gates, and operations. The AI-based system remains an artifact to be built and maintained, while developers are primarily framed as technical implementers of machine learning workflows (e.g., Amershi et al., 2019a; Studer et al., 2021; Wan et al., 2019).

By contrast, contemporary IS scholarship characterizes AI-based systems as agentic artifacts whose behavior cannot be fully specified ex ante and whose reasoning is often only partly intelligible ex post (Berente et al., 2021; Faraj et al., 2018; Zhang et al., 2021). For developers, this shifts the focus of development from programming fixed rules to engaging with a learning and partly opaque system. Their development practices determine how they cope with opacity and with the inability to fully specify or inspect the system's behavior. Understanding how these development practices unfold is relevant because they set the conditions for how users can later engage with these systems during use (Baird & Maruping, 2021). Accordingly, the third research question is:

Research Question 3 (RQ3): How do developers enact development practices when developing AI-based systems whose behavior they cannot fully specify or inspect, and how do these practices condition how users engage with these systems?

In **Perspective II**, this dissertation addresses the overarching research question by examining how ISD is reshaped when AI augments developers in their ISD processes. While Perspective I considers AI as an ISD output in the form of an AI-based system, Perspective II focuses on AI-augmented ISD, that is, ISD processes in which developers work with generative AI while building conventional, rule-based information systems whose behavior is specified in program code and deterministic business logic (Bürkland & Zachariassen, 2014; Chalmers, 2006; Vara & Marcos, 2012). The developers considered in Perspective II are therefore primarily traditional software developers who design, implement, test, and maintain software applications and systems across the software lifecycle (Meyer et al., 2017; Minelli et al., 2015). Software developers have a long history of applying tools and automation technologies to cope with the complexity of software development (Banker & Kauffman, 1991). These technologies typically address specific, well-defined use cases along the software development lifecycle, following formal rules and pre-defined logics, such as automated code generation from explicit models or schemas, finding bugs using static analysis, or identifying run-time errors using static program analyzers (Bessey et al., 2010; Cousot et al., 2005; Sunitha & Samuel, 2019).

The advent of generative AI changes how software development can be supported (Rosenkranz et al., 2025). Generative AI is part of a new generation of agentic IS artifacts that can learn, adapt, and act with a degree of autonomy (Baird & Maruping, 2021). By learning patterns in data, it can generate entirely new content (Feuerriegel et al., 2024; Foster, 2022). In Perspective II, the focus is on large language models (LLMs), a subset of generative AI for text understanding and text generation, due to their relevance for software development tasks (Chui et al., 2023a). LLMs are general-purpose foundation models that are pre-trained on vast text-based data, including public code repositories (Touvron et al., 2023). They are not optimized for a specific organizational context or task, but can be adapted to many tasks without task-specific training (Bommasani, 2021; Brown et al., 2020; Radford et al., 2018). For ISD, this implies that, on the one hand, LLMs can be applied across a wide range of software development tasks, such as code generation from natural language descriptions (Bird et al., 2023), requirements-related text processing (Zhang et al., 2023), unit test generation (Yuan et al., 2023) or code summarization (Ahmed & Devanbu, 2022). On the other hand, because these models are trained predominantly on public data and not on the specific code bases, architecture and system contexts, including legacy systems, of a given organization, their outputs can be contextually inappropriate or incorrect in such idiosyncratic organizational settings (Huang et al., 2025).

To address this mismatch, research on LLMs has developed techniques that show how models can be technically guided once task goals and relevant data are known, for example, through prompting techniques and retrieval-augmented generation, which provide task instructions and

contextual information to shape the model's output generation at inference time (Brown et al., 2020; Lewis et al., 2020; Liu et al., 2023). These techniques specify how to better align LLM outputs with a given task and context and thus show how LLM behavior can be shaped once a problem framing and relevant data sources are in place. They offer little insight into how developers in organizations use such techniques in their everyday work to realize and further shape generative AI-augmented ways of accomplishing development tasks.

Research on generative AI in software development has begun to examine the use of LLMs by developers and what outcomes this yields. One stream of work maps prevalent task scenarios for generative AI in software development, how developers interact with LLM-based tools in these scenarios, and the benefits and challenges they perceive (Banh et al., 2025; Barke et al., 2023; Santos et al., 2025). A second stream, largely based on experimental designs, analyzes the impact of LLMs on ISD outcomes, such as productivity (Weber et al., 2024; Ziegler et al., 2024). These research streams primarily clarify what kinds of development tasks LLMs can support and how well they perform in those tasks.

However, we know comparatively little about how software developers in organizational ISD settings realize and shape generative AI augmentation in their ISD processes to accomplish their development tasks. This understanding is relevant because developers in organizational settings are confronted with powerful but general-purpose LLMs on the one hand and with a complex, historically grown systems landscape and project-specific constraints on the other hand. To realize augmentation from generative AI, developers must shape its use within their ISD processes so that it fits specific organizational and task contexts. Therefore, the fourth research question asks:

Research Question 4 (RQ4): How do software developers realize and shape generative AI-augmented ways of accomplishing their development tasks in organizations?

As generative AI becomes embedded in ISD processes as a general-purpose technology that can support a wide range of software development tasks, it is expected to reshape how software development work is carried out and, in turn, how developers experience their work (Ahmed & Devanbu, 2022; Banh et al., 2025; Barke et al., 2023; Nguyen-Duc et al., 2023; Ozkaya, 2023; Yuan et al., 2023; Zhang et al., 2023). Existing research on the impact of generative AI in ISD has primarily focused on examining technical productivity factors, that are, factors related to the ISD output or process (Wagner Ruhe, 2018). This research stream demonstrates that generative AI can increase output and speed in programming tasks (Cui et al., 2024; Peng et al., 2023; Ziegler et al., 2024). In contrast, the first empirical studies on generative AI in workplace settings point to a more ambivalent picture at the level of individual employees. Generative AI can introduce new demands and technostress that undermine individual well-being (Chuang et al., 2025). Studies further show that while generative AI can improve performance, it has also been associated with decreased motivation and reduced critical thinking (Gerlich, 2025; Wu et al., 2025). These mixed findings indicate that focusing on technical factors alone is insufficient

to understand the implications of generative AI-augmented ISD. Moreover, research on developers' productivity in organizational settings shows that productivity in ISD is not captured by technical factors alone (e.g., lines of code or number of completed features). Instead, developers' productivity and willingness to contribute depend on individual-level work outcomes, such as being satisfied with one's job and feeling enthusiastic and energized at work (Murphy-Hill et al., 2019; Storey et al., 2019). In other words, developers' individual-level outcomes are key predictors of productivity in ISD. Against this backdrop, focusing only on the technical productivity effects of generative AI in ISD is problematic because it provides an incomplete understanding and risks overlooking situations in which technical indicators improve while individual-level outcomes could deteriorate, thereby potentially undermining the productivity gains.

Thus, to gain a more comprehensive understanding of the implications of AI-augmented ISD, it is crucial to examine how generative AI affects developers' individual-level work outcomes. Consequently, this dissertation investigates the impact of generative AI on the individual-level work outcomes of software developers in organizational ISD settings. Specifically, this dissertation focuses on three individual-level work outcomes that are central in the organizational behavior literature: work engagement, job satisfaction, and experience of meaningful work (Bowling & Hammond, 2008; Schaufeli et al., 2006; Spreitzer, 1995).

Generative AI offers developers a broad range of possibilities for supporting their development work, but at the same time, the outputs of generative AI need to be evaluated and adapted to specific development contexts to avoid inappropriate or incorrect results (Bird et al., 2023; Huang et al., 2025; Santos et al., 2025). Existing research indicates that, as employees integrate generative AI into their work, they proactively shape their tasks, responsibilities, and collaboration patterns (Freise et al., 2025; Mayer et al., 2025a). These bottom-up adjustments to the job can be captured by the concept of job crafting. Job crafting refers to the self-initiated changes employees make to their jobs to better align work with their skills and needs (Wrzesniewski & Dutton, 2001). Building on the approach-avoidance, role-resource model of job crafting (Bruning & Campion, 2018), this dissertation distinguishes between approach crafting, in which developers expand their roles and resources to pursue valued aspects of work, and avoidance crafting, in which they reduce their roles or interactions to distance themselves from less desirable aspects of work.

Prior research shows that approach job crafting is associated with higher work engagement, job satisfaction, and work meaning, whereas avoidance crafting tends to be weakly or even negatively related to such positive outcomes (Bruning & Campion, 2018; Lichtenthaler & Fischbach, 2019; Petrou et al., 2012; Zhang & Parker, 2019). Emerging studies on generative AI and job crafting suggest that generative AI can lead to both approach and avoidance forms of job crafting (Freise et al., 2025; Mayer et al., 2025a). This suggests that generative AI's impact on individual developers can depend on how the developers craft their jobs in AI-augmented ISD. Accordingly, this dissertation conceptualizes job crafting as a key mediator through which the

impact of generative AI use on developers' individual-level work outcomes can be explained, with approach versus avoidance crafting expected to have different implications for job satisfaction, work engagement, and work meaning. As job crafting in the context of generative AI has been examined primarily in qualitative studies so far, this dissertation aims to quantitatively investigate how job crafting mediates the relationship between generative AI use and developers' individual-level work outcomes in organizations. This leads to the fifth research question:

Research Question 5 (RQ5): How does developers' use of generative AI influence their individual-level work outcomes (work engagement, job satisfaction, and work meaning), and what is the mediating role of developers' job crafting in this relationship?

1.3 Dissertation Structure

This cumulative dissertation comprises this introductory paper and six full research papers (Paper I to VI). The introductory paper motivates the overarching inquiry of how AI reshapes ISD in organizations, derives the detailed research questions, and provides the dissertation's theoretical foundation and methodological approach (Chapter 1 to 3). It then summarizes the main results of the six papers (Chapter 4) and discusses the dissertation's theoretical and practical implications (Chapter 5), before outlining limitations, opportunities for future research, and concluding (Chapter 6 and Chapter 7).

The six full papers included in the dissertation are organized into two perspectives that jointly address the overarching research question. Perspective I (Paper I to IV) examines the AI-based systems as ISD outputs, analyzing their development from a triadic view of developers, users, and the AI-based system. Perspective II (Paper V and Paper VI) examines AI within ISD processes, investigating how generative AI augments ISD in organizations and its implications for developers' jobs and individual work outcomes. Figure 2 provides an overview of the dissertation's structure.

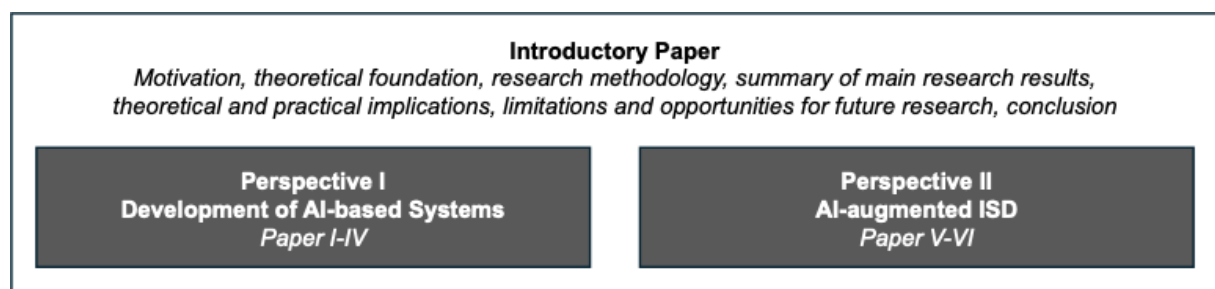


Figure 2: Overall structure of the cumulative dissertation

2 Theoretical Background

This chapter presents the theoretical foundation for examining how AI reshapes ISD. Section 2.1 reviews core ISD research and introduces ISD stakeholders, processes, and outputs as central themes that structure how ISD is commonly conceptualized in the IS field. Building on this, Section 2.2 turns to AI-based systems as ISD outputs and explains how the shift from deterministic to agentic artifacts alters what it means to develop agentic systems whose behavior emerges from learning from data and cannot be fully specified or inspected. Section 2.3 then discusses AI-augmented ISD, highlighting how generative AI is entering development work as a general-purpose technology that changes how developers accomplish and experience ISD work. Finally, Section 2.4 introduces key theoretical lenses that guide the dissertation's analyses, which are knowledge boundaries, cognitive fit, delegation, and job crafting. It also clarifies how these lenses are applied across the papers of this dissertation to theorize both the triadic development of AI-based systems (Perspective I) and the implications of generative AI use for developers' work in ISD (Perspective II).

2.1 Information Systems Development

Information systems development (ISD) has been a core topic of concern in information systems (IS) research since the inception of the IS field (Bostrom & Heinen, 1977b; Matook et al., 2021). ISD is understood as the full set of activities (planning, analysis, design, building, testing, maintenance) undertaken by human and software agents to create a working information system, embedded in a social, organizational, and technical context in which multiple stakeholders influence and are influenced by these activities (Matook et al., 2021). Software development is a core component of ISD and is carried out within ISD processes. It focuses on specifying, implementing, testing, and maintaining software-based systems (Meyer et al., 2017; Sommerville, 2015).

Three major ISD themes are identified by Matook et al. (2021): ISD stakeholders, ISD processes, and ISD outputs. This view is important, as it clarifies who is involved in ISD, how development work is organized and conducted, and what kinds of outputs are in focus.

2.1.1 ISD Stakeholders

The ISD research theme of ISD stakeholders focuses on the actors who shape and are shaped by development work, in particular, users and developers or development teams, alongside managers. Users and their roles in ISD have been extensively studied because they hold domain knowledge required in systems development and bear the consequences of ISD outputs. User-related issues examined include user participation, user involvement, and user attitude (Barki & Hartwick, 1989, 1994; He & King, 2008; Hunton & Beeler, 1997).

Developers and the development team have been examined as central ISD stakeholders, whose characteristics shape how development unfolds (Matook et al., 2021). This work shows that development teams' domain knowledge across business and technical domains affects ISD performance (Hahn & Lee, 2021), that employment arrangements and job design, such as contract developers versus permanent developers, shape developers' work attitudes and behaviors in software development teams (Ang & Slaughter, 2001), and that team autonomy and team characteristics condition how effectively development teams respond to requirement changes (Lee & Xia, 2010).

2.1.2 ISD Processes

The research theme of ISD processes addresses how development work is organized and carried out, encompassing methods, development practices, and organizational and coordination processes that contribute to the development of an information system (Matook et al., 2021). A first stream focuses on methods and techniques that guide ISD processes. Studies have examined differences in ISD methods (Mahmood, 1987), analyzed specific techniques, for example, how prototyping can be controlled through risk analysis (Baskerville & Stage, 1996), and how pair programming affects performance (Balijepally et al., 2009). With the rise of agile methods, ISD research has increasingly conceptualized ISD processes as flexible and adaptive development practices rather than standardized processes (Conboy, 2009).

A second stream examines how ISD methods and techniques are applied and adapted in organizational practice. This work shows that ISD processes are realized through situated development practices. Fitzgerald (1997) shows how organizations selectively use and modify formal ISD methods in practice. Ågerfalk and Fitzgerald (2006) similarly show how globally distributed software processes emerge as flexible, locally adapted practices. Iivari and Maansaari (1998) further demonstrate that the usage of ISD methods varies across organizations and is shaped by contextual conditions. This stream conceptualizes ISD processes as situated enactment of development practices by developers and development teams within organizational settings.

A third stream focuses on the organizational and coordination aspects of ISD processes, examining how contributions from different stakeholder groups are coordinated and integrated. Within this stream, ISD processes are predominantly analyzed through relationships among human agents. A substantial body of work studies the user-developer relationship as a central locus of coordination, particularly in requirements determination and knowledge integration. In requirements analysis, interactions between users and developers are needed to elicit, structure, and validate requirements for information systems (Byrd et al., 1992; Watson & Frolick, 1993). Research on user-developer communication and social processes shows how expectations, power, and communication patterns between users and developers shape system outcomes (Gallivan & Keil, 2003; Hirschheim et al., 1991; Newman & Robey, 1992). Building on a knowledge per-

spective, this stream further examines how users and developers jointly contribute and integrate knowledge in ISD. Development is conceptualized as social action in which heterogeneous understandings are interpreted, shared, and reconciled. Shared or common knowledge between users and developers, users as knowledge co-producers in ISD projects, and boundary interactions in requirements elicitation are central to achieving shared understanding and successful outcomes (Hirschheim et al., 1991; Hsu et al., 2012; Rosenkranz et al., 2014; Tesch et al., 2009).

Overall, this organizational stream on ISD processes primarily studies how development activities are organized and coordinated between human agents, especially in the dyadic relationship between users and developers, but also between development teams and managers, and within development teams (Jenkin et al., 2019; Kudaravalli et al., 2017; Newman & Robey, 1992). The information system itself is largely treated as a deterministic IS artifact whose behavior is rule-based and specified by ISD stakeholders (Berente et al., 2021; Bostrom & Heinen, 1977b; Byrd et al., 1992; Herbst, 1996; Matook et al., 2021).

2.1.3 ISD Outputs

The research theme of ISD outputs concerns both the characteristics of the resulting information system and ISD performance (Matook et al., 2021). On the system side, ISD outputs have been examined in terms of system characteristics such as reliability, maintainability, and complexity (Banker et al., 1993; Halloran et al., 1978; Swanson & Beath, 1989).

Regarding ISD performance, research distinguishes between process performance (e.g., development productivity, adherence to budget and schedule) and product performance (e.g., system quality, user satisfaction). Quality management studies conceptualize ISD performance as the joint outcome of these two dimensions and show that total quality management practices, such as top management leadership, process management, and continuous improvement, enhance both process efficiency and product quality (Ravichandran & Rai, 1999, 2000). Research further demonstrates that higher process maturity is associated with shorter cycle times, lower effort, and fewer defects, highlighting the role of the maturity of processes and ongoing process improvements for process and product performance (Harter et al., 2000; Krishnan et al., 1999). Software development research often operationalizes process and product performance using technical indicators such as change frequency and effort, code size, code complexity, fault density, or build failure rates (Banker et al., 1993; Graves et al., 2002; Saidani et al., 2020; Subramanyam & Krishnan, 2003; Wagner & Ruhe, 2018).

A further stream of research highlights that ISD process and product performance are related to individual-level work outcomes. In software development, empirical studies show that developers' productivity cannot be captured by technical factors alone. Instead, ISD performance is related to developers being enthusiastic, satisfied, and engaged in their work (Murphy-Hill et al., 2019; Storey et al., 2019). This aligns with organizational behavior research that shows that job

satisfaction and work engagement are associated with job and task performance (Christian et al., 2011; Judge et al., 2001). A comprehensive view of ISD outputs, therefore, needs to consider both technical performance indicators and individual-level work outcomes of ISD stakeholders.

2.2 Artificial Intelligence-based Systems as Outputs of Information Systems Development

2.2.1 Shifting Nature in ISD for AI-based Systems

When developing artificial intelligence (AI)-based systems, the ISD output shifts from a traditional deterministic IS artifact to an agentic IS artifact (Baird & Maruping, 2021). This dissertation considers agentic IS artifacts that apply AI to support tasks within organizations. With agentic AI-based systems, the development relationship between developers and the system changes. AI-based systems can self-learn decision models and act with a degree of autonomy (Berente et al., 2021). They are agentic in that they can operate autonomously, learn from data, and adapt their behavior without explicit programming by humans (Baird & Maruping, 2021; Berente et al., 2021; Chollet, 2021).

The role of developers changes in that they embed goals into the agentic AI-based system, while the decision model is learned by the system from data rather than being fully specified *ex ante* by developers (Baird & Maruping, 2021; Murray et al., 2021). As a result, developers cannot fully specify the behavior of the AI-based systems they develop. For example, Strich et al. (2021) report on a company that introduced an AI-based credit decision system that autonomously learns its decision model to make loan decisions, determine terms and conditions, and adjust lending criteria based on customer behavior and market changes. Moreover, the learned decision model typically remains unavailable for user scrutiny and is often not fully transparent even to developers, so they cannot fully inspect *ex post* how the AI-based system arrives at its outcomes (Burrell, 2016; Faraj et al., 2018; Zhang et al., 2021). Traditional ISD approaches, which assume deterministic systems whose behavior is specified and knowable, are therefore inadequate for addressing these challenges that these agentic AI-based systems pose for both developers and users.

Against this backdrop, a nascent stream of IS research has begun to examine how AI-based systems enter ISD as agentic IS outputs, but it largely does so from a dyadic perspective. One stream foregrounds the developer-user relationship and extends human-centered and participatory design principles to AI-based systems, emphasizing how developers and users align their values, goals, and practices during development (Auernhammer, 2020; van den Broek et al., 2021). Another stream emphasizes the user-AI relationship and investigates how users can understand, oversee, and remain accountable for AI-based systems, for example, by enveloping inscrutable AI into sociotechnical arrangements or designing explainable outputs that support user interpretation (Arrieta et al., 2020; Asatiani et al., 2021; Gronsund & Aanestad, 2020; Kane

et al., 2021).

These streams provide important insights into the development of AI-based systems, but leave underexplored how developers, users, and the AI-based system together form a triad of interdependent agents whose knowledge, goals, and decision models must be aligned throughout development and use (Ågerfalk et al., 2022; Baird & Maruping, 2021). Perspective I of this dissertation addresses this gap by adopting a triadic perspective on the development of AI-based systems, treating developers, users, and the AI-based system as distinct but interdependent agents.

2.2.2 Discriminative and Generative AI-based Systems

This dissertation focuses on two types of AI-based systems that are widely adopted in organizations and fall within its scope: discriminative and generative AI-based systems (Berg et al., 2023; Chui et al., 2023a; Foster, 2022). By learning the relationships between inputs and outputs or by learning the structures within the input data, discriminative AI classifies inputs, predicts labels given one or more inputs, or clusters inputs (James et al., 2021). Generative AI, in contrast, learns the underlying structure and patterns in the data to generate entirely new data that is similar to a given dataset (Foster, 2022). Rather than predicting a label or category, generative AI generates new content, such as text or images (Foster, 2022). For generative AI, this dissertation focuses specifically on large language models (LLMs), a subset of generative AI for text understanding and text generation, due to their growing relevance for organizations (Yang et al., 2024). LLMs are general-purpose foundation models that are pre-trained on vast and diverse text-based data (Touvron et al., 2023), making them applicable across a variety of domains (Chui et al., 2023a), such as customer support (Brynjolfsson et al., 2023), internal knowledge management (Alavi et al., 2024), or software development (Peng et al., 2023).

Discriminative AI-based systems and generative AI-based systems differ in how developers can shape them during development regarding the systems' agentic attributes, including goals, decision models, endowments, and roles (Baird & Maruping, 2021). The agentic attributes of discriminative AI-based systems are typically developed in the triadic relationship of developers, users, and the AI-based system (Baird & Maruping, 2021; Glaser et al., 2021; Waardenburg & Huysman, 2022). Developers, in collaboration with users, embed the goals into the system and provide endowments, such as selecting learning algorithms, providing domain- and task-specific data, as well as infrastructure for data processing. The system's decision model then emerges through learning on its own via optimization procedures, which fit the model to the specified goals using the provided endowments (Goodfellow et al., 2016). The role of the discriminative AI-based system is established to support specific tasks where the AI-based system is perceived as being helpful to achieve these goals (Baird & Maruping, 2021).

In contrast, the agentic attributes of generative AI-based systems, specifically LLMs, emerge

differently. LLMs come with predefined goals and pre-trained decision models, developed outside of the organization and without the involvement of developers and users who engage with the LLMs in the organizational setting (Radford et al., 2018; Touvron et al., 2023). The decision model of LLMs is shaped by learning the general patterns and structures of text and language through training on vast, heterogeneous text-based datasets that are not specific to an organization, domain, or task (Radford et al., 2018; Touvron et al., 2023). LLMs model the probability distribution of tokens, where a token refers to a word, subword, or character (Foster, 2022). Given a preceding input context, LLMs predict the likelihood of the next tokens and generate the next token by sampling from this distribution (Vaswani et al., 2017). As such, LLMs are initially general-purpose models, neither optimized for a specific organizational context nor for task-specific goals (Brown et al., 2020; Radford et al., 2018).

Table 1: Discriminative and LLM-based generative AI-based systems characterized by their agentic attributes (adapted from Baird and Maruping (2021))

Agentic Attributes	Discriminative AI-based System	Generative AI-based System (focused on LLMs)
Goal	Goals are embedded into the discriminative AI-based system by developers in collaboration with users (Baird & Maruping, 2021)	Goals are predefined outside of the organization as next-token prediction rather than an organization-specific task goal (Radford et al., 2018; Touvron et al., 2023). Task goals can be shaped in practice at inference time via prompting, through predefined system prompts or user prompts (Brown et al., 2020; Liu et al., 2023).
Decision model	The system's decision model emerges through learning, as optimization procedures fit the model to the developer-specified goals using endowments provided by developers and users (Baird & Maruping, 2021)	The decision model is pretrained outside of the organization as a foundation model (Radford et al., 2018; Touvron et al., 2023). Organizations typically apply these models unchanged (Saffarizadeh et al., 2024), though they can further adapt them through fine-tuning to embed factual knowledge in the model parameters (Radford et al., 2018; Roziere et al., 2023).
Endowments	Developers, in collaboration with users, curate and provide domain- and task-specific data and the infrastructure for data processing, thereby enabling the learning process (Amershi et al., 2019a)	LLM-based systems are initially trained on vast, heterogeneous text-based datasets that are not specific to a task or domain (Radford et al., 2018; Touvron et al., 2023). Their endowments can be dynamically expanded at inference time through retrieval-augmented generation (RAG), which provides access to domain- and task-specific documentation without retraining the model (Lewis et al., 2020).
Role	The role of the discriminative AI-based system is established to support specific tasks (Baird & Maruping, 2021)	The role of the LLM-based system is fluid and depends on how human agents provide goals and endowments (Krakowski, 2025)

Rather than directly embedding goals and developing the decision model, LLMs can be adapted for specific tasks in the collaboration of developers and users in two ways: (1) by shaping the goals through providing inputs such as task instructions and examples in the interaction, often referred to as prompts (Brown et al., 2020; Liu et al., 2023) that can be either provided as

predefined system prompts or dynamically provided as user prompts; and (2) by providing additional endowments in the form of additional, task-specific knowledge base from a domain or an organizational context that can be accessed by the LLM during output generation without additional training, referred to as retrieval-augmented generation (RAG) (Lewis et al., 2020). This does not change the underlying decision model. Instead, it influences how the model accesses external task-specific and context-specific knowledge and generates responses based on it. As a result, the role of the generative AI-based system is fluid and depends on how dynamically goals and endowments are specified and provided by developers and users. LLMs can be further trained through fine-tuning, modifying their decision models (Radford et al., 2018; Roziere et al., 2023). However, the dissertation focuses on the adaptations as described above without the need to fine-tune the model, as they are predominant ways through which organizations adapt LLMs to support organizational tasks. The differences in how developers can configure the discriminative and LLM-based generative AI-based systems are summarized along the agentic attributes in Table 1.

2.3 Artificial Intelligence-augmented Information Systems Development

Software developers are increasingly augmenting their development work with generative AI (Chui et al., 2023a). In this perspective, AI enters ISD within the development process, where it supports software developers in designing, implementing, testing, and maintaining information systems (Banh et al., 2025; Barke et al., 2023; Ozkaya, 2023; Santos et al., 2025).

Software development has a long tradition of using tools and automation to cope with the complexity of building and evolving software-based systems (Banker & Kauffman, 1991). Research on computer-aided software engineering (CASE) examined how tool support for activities such as design, code generation, and error detection can improve development productivity (Banker & Kauffman, 1991). Traditional forms of automation in software development typically target narrow, well-defined tasks along the development lifecycle and operate on the basis of explicit rules or specified models. Examples include automatic code generation from models and schemas, static analysis for identifying potential bugs, or static program analyzers that detect run-time errors (Bessey et al., 2010; Cousot et al., 2005; Sunitha & Samuel, 2019). These tools share the characteristic that they implement predefined logic for specific use cases.

The advent of generative AI, particularly large language models (LLMs), changes the nature and scope of support for work and activities in ISD. Generative AI-augmented ISD differs from traditional automation along several dimensions. First, LLMs are general-purpose rather than task-specific models. The same model can be used across a variety of different software development tasks, including requirements-related text processing, code generation, test creation, documentation, and code explanation, among others (Ahmed & Devanbu, 2022; Bird et al., 2023; Yuan et al., 2023; Zhang et al., 2023). Second, instead of executing fixed rules, LLMs

generate outputs probabilistically based on learned patterns in large-scale training data. This enables assistance in more open-ended and ambiguous problem situations (Baird & Maruping, 2021), for example when developers explore solution approaches (Barke et al., 2023). Third, because LLMs are trained primarily on public data and not on an organization's specific code bases, architectures, and legacy systems, their suggestions need to be evaluated and adapted to fit the idiosyncratic organizational contexts and constraints (Huang et al., 2025). As a consequence, when developers use generative AI, they do not simply apply a deterministic tool to a predefined input. Instead, they engage in an interactive process with a general-purpose, agentic system whose outputs must be obtained through prompts and then evaluated and integrated into their development work (Barke et al., 2023). Existing research largely examines generative AI usage scenarios and their benefits and challenges in ISD (Banh et al., 2025; Barke et al., 2023; Ozkaya, 2023; Santos et al., 2025). Yet, it remains comparatively underexplored how software developers in organizational settings realize and shape generative AI-augmentation in their ISD processes to accomplish their development work. This gap is addressed by Perspective II of this dissertation.

Furthermore, because generative AI can, in principle, augment a broad range of development activities, it is widely discussed as transforming developers' jobs and work experience (Nguyen-Duc et al., 2023; Sætra, 2023; Sauvola et al., 2024; Ulfsnes et al., 2024). Existing research on the implications of generative AI use in ISD has so far primarily focused on examining the technical productivity indicators, such as volume and speed of code production, and shows that generative AI use can increase output and reduce completion time in programming tasks (Cui et al., 2024; Peng et al., 2023; Ziegler et al., 2024). Studies on software development productivity and ISD performance, on the other hand, demonstrate that technical indicators alone are not sufficient. Developers' individual-level work outcomes, such as work engagement and job satisfaction, are important predictors of work performance (Christian et al., 2011; Murphy-Hill et al., 2019; Storey et al., 2019). As we lack a more comprehensive understanding of how generative AI use in organizational ISD settings affects developers' individual-level work outcomes and how it changes their jobs, Perspective II of this dissertation addresses these open questions.

2.4 Theoretical Lenses

This dissertation draws on multiple theoretical lenses to theorize how AI reshapes ISD from two perspectives: the development of AI-based systems and the augmentation of developers' work through generative AI. The following sections introduce four core theoretical lenses: knowledge boundaries, cognitive fit, delegation, and job crafting. The sections outline how each lens is used as a sensitizing lens to illuminate a distinct aspect of the phenomenon, ranging from challenges in developer-user-AI triads to shifts in developers' work practices in AI-augmented ISD.

2.4.1 Knowledge Boundary

Building on the relational view from Carlile (2004), knowledge boundaries arise when actors (1) hold different knowledge, (2) face novelty, and (3) are mutually dependent on each other to accomplish their tasks. Under these conditions, differences in expertise become consequential for coordinating work. Prior IS and organization research has shown that integrating specialized and distributed knowledge across such boundaries is critical for effective systems development (Faraj & Sproull, 2000; Patnayakuni et al., 2007; Rus et al., 2002; Walz et al., 1993).

In the context of AI-based systems, the involvement of an agentic IS artifact means that these conditions also arise at the interface between humans and the AI-based system, creating an additional knowledge-related boundary in ISD. First, AI-based systems differ from human actors in the amount and type of knowledge they can process. They handle vast data volumes at high speed and are trained on heterogeneous data sources spanning organizational units and external environments, thereby exceeding human capabilities (Baird & Maruping, 2021). They can uncover patterns that remain hidden to humans (Dhar, 2013), making the resulting insights different in scope and character.

Second, AI-based systems introduce novel ways of representing and solving problems. Through machine learning, they acquire a probabilistic decision logic that is learned from data rather than explicitly programmed (Chollet, 2021). The resulting algorithmic rules are often opaque (Burrell, 2016) and can produce recommendations that diverge from established human decision logics (Murray et al., 2021). For developers and users, this means that AI-based systems can frame problems in unfamiliar ways, surface unexpected relations, and suggest actions that are difficult to reconcile with domain knowledge. Third, these differences and novelties become consequential because humans and AI-based systems are increasingly mutually dependent on each other's contributions to accomplish organizational tasks (Raisch & Krakowski, 2021). AI-based systems, therefore, introduce knowledge-related boundaries at the interfaces among developers, users, and the AI-based system that must be actively managed during development.

To understand how these boundaries are bridged in practice, this dissertation draws on the concept of boundary objects (Star, 2010; Star & Griesemer, 1989). Boundary objects are artifacts that are sufficiently plastic to be used differently in multiple domains, yet robust enough to maintain a common identity across them (Star & Griesemer, 1989). Star (2010) emphasizes that boundary objects are part of concrete arrangements of work and information that emerge from work requirements. In IS and organization studies, boundary objects have been used to explain how diverse communities of practice coordinate their work around design documentation, prototypes, data, and information systems (Levina & Vaast, 2005; Majchrzak et al., 2012; Pawlowski & Robey, 2004; Tim et al., 2017).

In this dissertation, the lens of knowledge boundaries and boundary objects is applied in Paper I, which examines the development of AI-based systems as a triadic relationship among

developers, users, and the AI-based system. The focal phenomenon is how knowledge-related boundaries emerge at these interfaces and how they are bridged through development practices and boundary objects. In this context, data objects, the AI-based system itself and its model artifacts, and business knowledge objects are identified and analyzed as boundary objects that enable the triad to coordinate development work.

2.4.2 Cognitive Fit

Cognitive fit posits that problem-solving performance improves when the external problem representation of the technology matches the demands of the problem-solving task (Vessey & Galletta, 1991). When both the external problem representation and the task emphasize the same type of information and knowledge requirements, a cognitive fit occurs. Under such a fit, costly transformations are avoided, and users' cognitive requirements are reduced in forming a mental representation for the task solution. As a result, problem-solving performance improves. When cognitive fit is absent, cognitive load increases, and the problem-solving performance degrades. Consequently, the implication of the cognitive fit theory is that developers should focus on determining the characteristics of the tasks that users must address and on supporting the task with an appropriate external problem representation.

The mental representation for task solution is formed by users based on the internalization of information from the external problem representation, and any internal representations of users relevant to the task solution (Shaft & Vessey, 2006). External representations are in the world that are, for example, external rules or constraints embedded in artifacts and environments, whereas internal representations are in the mind, such as schemas or mental images (Zhang & Norman, 1994). It is useful to distinguish a user's background internal representation of the task domain drawn on existing knowledge from the task-specific mental representation for the task solution (Shaft & Vessey, 2006). The latter is constructed to address the problem-solving task by integrating the system's external problem representation with the users' background internal representation. It is this constructed mental representation for problem solution that guides actions.

In this dissertation, cognitive fit serves as the theoretical lens in Paper II. The focal phenomenon is the development of an AI-based system to support human-AI problem-solving tasks within an organization. In this context, the analysis centers on the AI-based system's external problem representations, which are the system's learned representations of the task and task domain as made available to users through the system interface and through development practices. Such external problem representations include, for example, system outputs, explanations, and embedded rules and constraints that are exposed to users. Users' internal representations are their mental representations of the problem domain, grounded in their existing knowledge and experience. Effective performance of these problem-solving tasks depends on aligning the task demands,

the AI-based system's problem representations, and the users' internal representations.

2.4.3 Delegation

Baird and Maruping (2021) introduce delegation as a central concept for understanding relationships between humans and agentic IS artifacts such as AI-based systems. They define delegation as the transfer of rights and responsibilities for task execution between agents. With this delegation framework, they shift attention away from traditional use-centric IS theories, which focus on how humans adopt and use technologies. Instead, they advance a delegation-centric perspective that asks what humans hand over to an IS artifact, under which conditions, and how this handover is governed over time. This shift is motivated by the observation that contemporary AI-based systems no longer operate as passive tools. AI-based systems perform tasks under uncertainty, initiate actions, and pursue goals in a partly autonomous and goal-directed manner. This repositions AI-based systems as agentic actors and makes delegation a key concept for theorizing human-AI relationships.

In their framework, Baird and Maruping (2021) characterize an agentic IS artifact through three agentic attributes: its preferences (goals and decision models), endowments, and role. Preferences comprise the goals the system is meant to pursue and the decision model through which it selects among options. Goals specify desired outcomes and thereby delimit what counts as successful action. The decision model represents causal links among inputs, actions, and outcomes, allowing the system to rank alternatives and guide decisions. Endowments refer to the resources that enable the AI-based system to act, such as access to data and digital infrastructure for data processing. Role captures the rights and responsibilities that the AI-based system is allowed to assume in the work system.

In this delegation framework, delegation is discussed from the perspective of a user-AI dyad. Baird and Maruping (2021), however, already note that this dyadic view is incomplete and call for extending the framework to include developers, who play a central role in making user-AI delegation possible. Developers, together with users, decide on goals, design and train decision models, provide endowments, and specify the role of the AI-based system. These choices take place prior to use and thereby set the conditions under which user-AI delegation can later occur. This dissertation takes up this call and adopts a triadic perspective on delegation, involving developers, users, and the AI-based system. In this view, developers determine the agentic attributes of the AI-based system. This triadic perspective enables an understanding of how developers' design choices, together with users' input and the AI-based system's learned behavior, jointly form the conditions for delegation that users encounter in use.

This triadic perspective is particularly relevant because a key step in this dissertation is to extend the delegation lens from the user-AI dyad to the developer-AI relationship itself. Traditional ISD perspectives often treat developers and the AI-based system as part of the same technical domain

(Amershi et al., 2019a; Sculley et al., 2015; Studer et al., 2021). However, AI-based systems learn a probabilistic decision logic that developers cannot fully specify *ex ante* or inspect *ex post* (Berente et al., 2021; Faraj et al., 2018; Zhang et al., 2021). This means that the path from inputs to outputs and the resulting decisions of the AI-based systems are not entirely under developers' control. Against this backdrop, Paper IV examines the developer-AI relationship through the delegation lens. Developers hand over a part of their traditional responsibility for determining the decision model to the AI-based system. Instead of explicitly programming all decision rules, they let the AI-based system learn its own decision model from data. Developers thus become principals who retain responsibility for outcomes but delegate parts of the procedure for reaching those outcomes to a learning system (Baird & Maruping, 2021; Eisenhardt, 1989a), whose inner workings may be partly opaque (Burrell, 2016; Faraj et al., 2018; Russell & Norvig, 2016). Treating the developer-AI relationship as delegation foregrounds that the development work for learning and partly opaque systems is a problem of handing over responsibility and managing that handover.

To analyze how this delegation between developers and the AI-based systems unfolds, this dissertation draws on the three core mechanisms of delegation identified by Baird and Maruping (2021): appraisal, distribution, and coordination. Appraisal refers to how human agents assess what is at stake for them in relation to the system and what can be done in response (Beaudry & Pinsonneault, 2005; Fadel & Brown, 2010). In a delegation context, appraisal focuses on identifying beneficial complementarities between human and AI capabilities (Baird & Maruping, 2021). This entails assessing the capabilities of the agents with respect to the task fit (Pinski et al., 2023), as well as the compatibility of their decision-making processes for a given task (Baird & Maruping, 2021). Appraisal may change as humans learn more about the capabilities of the AI-based system during interaction or when the system adapts and is updated (Baird & Maruping, 2021). In this dissertation, appraisal is used to examine how developers assess user capabilities, AI capabilities, and their compatibility to decide how their own responsibility can be handed to the AI-based system.

Distribution addresses the actual transfer of rights and responsibilities between agents. Because delegation is tied to goals, this transfer must be governed so that the agentic IS artifact pursues the intended objectives (Baird & Maruping, 2021; Eisenhardt, 1989a). Distribution is realized through contracting and regulating. Contracting concerns how agents agree, explicitly or implicitly, on goals and acceptable behavior, identify misalignments, and define how conflicts should be resolved (Castelfranchi & Falcone, 1998; Eisenhardt, 1989a; Hadfield-Menell & Hadfield, 2019). Regulating refers to *ex-ante* control activities that set boundaries on the agent's behavior, for example, by limiting data access or constraining the actions the AI-based system may autonomously take (Baird & Maruping, 2021; Kirsch, 1996). In the context of AI-based systems, developers face the challenge that they cannot fully align the system's behavior with their goals because they can only set objectives and constraints that guide the system's

learning (Hadfield-Menell & Hadfield, 2019). As various cases show, developers may design an AI-based system for specific goals, yet the deployed system can still produce harmful or unexpected results (Amodei et al., 2016). In this dissertation, distribution is used to analyze how developers design goal constraints, learning constraints, and ex-ante control mechanisms to govern the responsibility of the decision model construction they hand to the AI-based system during development.

Coordination refers to aligning actions and tasks under conditions of interdependence and uncertainty so that work advances and goals are achieved (Gulati et al., 2012; Malone & Crowston, 1994). For delegation, coordination is crucial because the delegator must monitor how delegated tasks are carried out and how this affects interdependent work. Baird and Maruping (2021) emphasize mechanisms such as monitoring and updating between humans and the agentic IS artifact. The delegator checks how the artifact has performed its work, the artifact reports outcomes through various representations, and both sides adjust their behavior accordingly (Okhuysen & Bechky, 2009). Delegation, therefore, entails not only transferring responsibilities but also coordinating this transfer to maintain accountability, predictability, and common understanding. In this dissertation, coordination is applied to examine how developers, in their role as delegators, monitor the AI-based system during development and how they respond to what they observe.

The delegation lens is used in this dissertation in Paper IV to move beyond a dyadic user-AI view and to theorize how developers delegate to the AI-based system during development and thereby set the conditions for later user-AI delegation.

2.4.4 Job Crafting

Job crafting refers to employees' self-initiated, proactive changes to the tasks, relationships, and cognitive boundaries of their jobs so as to better align work with their skills, motives, and aspirations (Tims & Bakker, 2010; Wrzesniewski & Dutton, 2001). It complements classic, top-down approaches to job design (Campion & McClelland, 1993; Hackman & Oldham, 1976), which assume that managers specify jobs in advance. In increasingly flexible and dynamic work arrangements made possible by advances in information technology, such top-down designs can be inadequate, so that employees themselves actively shape their jobs in practice (Demerouti, 2014). Through job crafting, employees can broaden their work roles by taking on additional tasks or expanding their social relations, or they can reduce their work roles by reducing certain tasks or social interactions (Bruning & Campion, 2018). Empirical studies show that such crafting can foster positive individual-level work outcomes, including higher work engagement (Bakker et al., 2012; Petrou et al., 2012), job satisfaction (Bruning & Campion, 2018), and work meaning (Berg et al., 2013).

Two influential theoretical perspectives on job crafting have emerged over time. The seminal

work from Wrzesniewski and Dutton (2001) conceptualizes job crafting as changes in task, relational, and cognitive boundaries. The job demands-resources perspective conceptualizes crafting as changes in job characteristics, such as increasing structural or social resources and adjusting job demands to better fit personal needs (Tims & Bakker, 2010; Tims et al., 2012). Both perspectives emphasize that employees can expand their jobs (e.g., by taking on additional tasks, building new relationships) or reduce them (e.g., by limiting workload, reducing draining interactions).

Bruning and Campion (2018) integrate these streams by distinguishing between and conceptualizing approach and avoidance crafting. Approach crafting activities are active, effortful, motivated activities, aiming at problem-focused and improvement-based goals to increase one's fit with the job by expanding roles and resources (Bruning & Campion, 2018). This can include enlarging one's work role, expanding social interactions, or creating structures and strategies to organize work more effectively. Avoidance crafting comprises proactive activities that reduce the work role, effort expenditures, or task accountability. This can include avoiding specific tasks through delegation, avoiding bothersome activities, or withdrawing from certain social interactions. This approach-avoidance distinction is central for understanding why some forms of job crafting are typically associated with beneficial outcomes, whereas others may have more ambivalent or negative effects (Bruning & Campion, 2018; Lichtenthaler & Fischbach, 2019).

The introduction of new technologies can serve as a catalyst for job crafting. Digital technologies and new work arrangements create both opportunities and pressures for employees to adapt how they do their work, (Demerouti, 2014; Li et al., 2022; Tarafdar & Saunders, 2022). Generative AI is a particularly salient example. It can automate routine elements of knowledge work, provide new informational resources, and alter collaboration patterns, thereby shaping how employees distribute their attention between mundane and complex tasks and how they interact with colleagues (Law & Varanasi, 2025; Stray et al., 2025). In software development, studies suggest that generative AI can help developers offload repetitive coding tasks and focus more on challenging or interesting problem-solving work, while changing patterns of collaboration and communication within teams (Freise et al., 2025; Stray et al., 2025; Ulfesnes et al., 2024). These dynamics can support approach crafting (e.g., expanding problem-solving roles or building new ways of work with generative AI) but may also foster avoidance crafting (e.g., reducing certain social interactions).

These bottom-up adjustments have implications for individual-level work outcomes. Prior research shows that approach job crafting is generally associated with higher work engagement, job satisfaction, and work meaning, whereas avoidance crafting tends to be weakly or negatively related to such positive outcomes (Bruning & Campion, 2018; Lichtenthaler & Fischbach, 2019; Petrou et al., 2012; Zhang & Parker, 2019). In the context of generative AI, this suggests that how developers craft their jobs in response to AI, whether predominantly through approach or avoidance forms of job crafting, can shape the impact of generative AI use on their individual-

level work outcomes.

In this dissertation, job crafting provides the theoretical lens in Paper VI to examine how developers' self-initiated changes to their jobs mediate the relationship between generative AI use and individual-level work outcomes.

3 Research Methodology

This cumulative dissertation applies both qualitative and quantitative research methods to address the overarching research question of how AI reshapes ISD in organizations and the detailed research questions across the six papers. This combination of qualitative and quantitative approaches is appropriate because scholarly understanding of AI-based systems development and AI-augmented ISD is still emerging and varies in maturity across the sub-questions of the overarching inquiry. This calls for both the exploration of novel phenomena and the testing of theoretically grounded relationships where prior work supports testable hypotheses (Edmondson & McManus, 2007). Applying different research methods further enables a richer understanding of the research topic (Mingers, 2001).

The qualitative research in this dissertation is motivated by the novelty and complexity of the focal phenomena. When phenomena are still emergent and not yet well captured by established constructs and measures, qualitative research is particularly suitable because it enables rich, in-depth, and contextualized understanding of how phenomena unfold in real-life settings. It supports the identification and refinement of salient concepts and the development of empirically grounded theory (Benbasat et al., 1987; Edmondson & McManus, 2007; Eisenhardt, 1989b). In this dissertation, case study research is the primary qualitative method to examine AI-based development practices in real organizational settings, where multiple stakeholders interact with AI-based systems as agentic IS artifacts. Case studies are well established in IS for investigating contemporary sociotechnical phenomena in their natural context, especially when the boundaries between phenomenon and context are intertwined (Lee, 1989; Recker, 2021). Depending on each paper's analytic objective, the qualitative papers draw on different case study designs, including single case study, embedded single case study, and multiple case study.

The quantitative research in this dissertation complements the case study-based context-sensitive theorizing by examining relationships in AI-augmented ISD across a broader sample and using standardized measures through survey research (Kaplan & Duchon, 1988). Survey research is a core quantitative method in IS for analyzing relationships among theoretically defined constructs and assessing the magnitude and statistical support of the hypothesized associations in a nomological net (Pinsonneault & Kraemer, 1993). In this dissertation, the survey research provides empirical evidence for hypothesized links among key constructs in AI-augmented ISD, and supports the generalizability to the target population (Lee & Baskerville, 2003).

Together, the dissertation's qualitative and quantitative papers provide a complementary empirical evidence base. The case studies develop context-sensitive explanations and empirically grounded theoretical accounts of emergent AI-based development phenomena, while the survey research provides statistically supported tests of key relationships in AI-augmented ISD across a broader sample. This combination supports a more comprehensive account of how AI reshapes ISD. Figure 3 summarizes the research approaches, methods, and designs used in this dissertation.

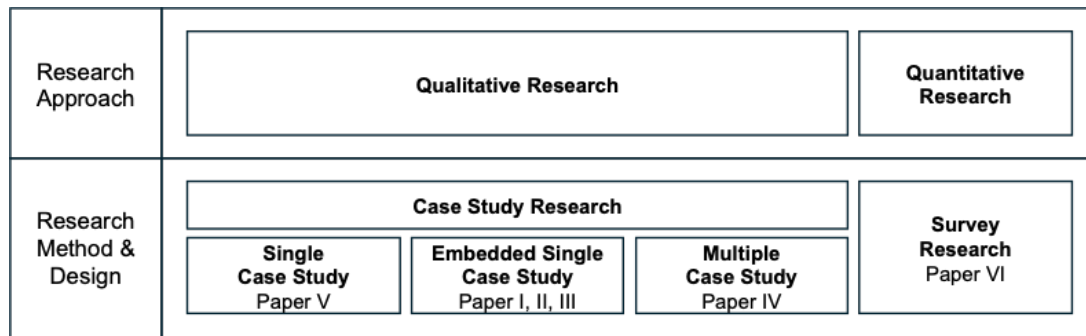


Figure 3: Overview of the research approaches, methods, and designs in this dissertation

3.1 Qualitative Research

3.1.1 Case Study Research Design

The qualitative studies in this dissertation (Paper I to V) are designed as exploratory, theory-building case studies to examine emergent and complex sociotechnical phenomena in AI-based systems development (Perspective I) and AI-augmented ISD (Perspective II). Exploratory case research is used to develop a better understanding of an ill-understood phenomenon and to derive empirically grounded concepts and provisional theoretical accounts (Benbasat et al., 1987; Edmondson & McManus, 2007; Eisenhardt, 1989b; Sarker et al., 2018).

Case studies are particularly appropriate here because AI-based systems development and AI-augmented ISD are sociotechnical phenomena that unfold in real-life settings and are shaped by organizational contexts. Studying them in their natural settings supports theorizing that remains sensitive to contextual conditions, which is essential when phenomenon and context are intertwined (Benbasat et al., 1987; Recker, 2021).

Key design choices of case study research concern (1) specifying the unit(s) of analysis (holistic versus embedded), (2) deciding on the number of cases to be studied (single versus multiple case design), and (3) determining how prior theory will be used (Benbasat et al., 1987; Dubé & Paré, 2003; Recker, 2021). A holistic single case design retains one primary unit of analysis for one case (often the organization). This design is appropriate when no meaningful subunits can be specified or when the theoretical interest is inherently global at the case level (Recker,

2021). In an embedded single case design, the study still investigates one overarching case, but it incorporates multiple embedded units of analysis within that case (e.g., different projects). Embedded designs are appropriate when the case boundary is an organization, but the phenomenon is most observable at a finer-grained level (e.g., development projects). Here, the organizational context can be preserved while also analyzing subunit dynamics (Benbasat et al., 1987). A multiple case design treats each case (e.g., project or organization) as a distinct analytical unit and builds theory through replication logic and cross-case comparison (Eisenhardt & Graebner, 2007; Recker, 2021). The replication-oriented comparison strengthens confidence in emerging findings because it helps to distinguish case-specific idiosyncrasies from recurrent patterns, thereby improving the robustness and generalizability of the resulting theoretical account (Eisenhardt & Graebner, 2007; Recker, 2021).

Paper I to III are designed as embedded single case studies. They treat the organization as the case boundary and analyze AI-based systems development in an organizational setting through multiple projects as embedded units of analysis. This design fits because the dissertation's theoretical interest in these papers is to develop empirically grounded accounts of AI-based systems development practices, and the project level provides the most direct access to the phenomenon.

Paper IV is designed as a multiple case study. It treats each AI-based systems development project as a distinct case and follows a replication-oriented sampling and comparison logic to build insights across cases. This is an appropriate choice because Paper IV's core theorizing targets patterns and variations in patterns of delegation practices within AI-based systems development projects. This requires treating projects as independent cases and strengthening the emerging findings through replication and extension across multiple cases.

Paper V uses a holistic single case design because the empirical focus is not on comparing different development projects but on understanding how developers across an organization realize and shape generative AI's action possibilities within their ISD processes as an organization-wide phenomenon. The holistic single case design is appropriate here because the theoretical interest is primarily at the case level.

In exploratory, theory-building case research, prior theory may be used in multiple ways. First, it may orient the initial research focus and data collection. Second, it may inform iterative analysis through a flexible set of *a priori* sensitizing concepts. Third, it may be further developed as part of the study's theoretical contribution (Eisenhardt, 1989b; Sarker et al., 2018). Crucially, any *a priori* concepts are treated as provisional and remain open to refinement, revision, or removal based on the empirical evidence, and as analysis progresses (Dubé & Paré, 2003; Eisenhardt, 1989b).

Across Paper I to V, the qualitative analyses are primarily inductive. They build explanations from close engagement with empirical material, while drawing on prior theory in a non-prescriptive way. The papers differ in how they use prior theory in the case study research

Table 2: Overview of qualitative case studies (Paper I-V)

Paper	Phenomenon Focus	Case Study Design	Cases and Unit(s) of Analysis	Primary Data	Theoretical Lens
Paper I	Developing AI-based systems in the triad of developers, users, and the AI-based system	Embedded single case study	Five AI-based systems development projects as embedded units within the case of AI-based systems development at one organization	24 interviews (developers and users)	Knowledge boundaries, boundary objects
Paper II	Developing AI-based systems to align them with joint human-AI problem-solving	Embedded single case study	Five AI-based systems development projects as embedded units within the case of AI-based systems development at one organization	24 interviews (developers and users)	Cognitive fit
Paper III	Developing generative AI-based systems	Embedded single case study	Three generative AI-based systems development projects as embedded units within the case of generative AI-based systems development at one organization	12 interviews (developers and users)	No lens
Paper IV	Developing AI-based systems that cannot be fully specified or inspected	Multiple case study	Eight AI-based systems development projects within one organization treated as distinct cases	34 interviews (developers and users)	IS delegation, temporal agency
Paper V	Realizing and shaping generative AI-augmented ISD	Single case study	One case of generative AI-augmented software development at one organization	24 interviews (software developers)	Affordance realization / action possibilities

process. Paper I uses the theoretical lenses of knowledge boundaries and boundary objects to focus attention on where and how knowledge-related boundaries arise among developers, users, and the AI-based system during development, and how boundary objects are used to bridge these boundaries. Paper II draws on the cognitive fit lens to sensitize to the challenges of aligning AI-based systems with joint human-AI problem-solving during their development. Paper III does not apply a theoretical lens at the outset. This choice reflects the early-stage character of generative AI-based systems development. Thus, Paper III prioritizes inductive category development. Paper IV starts from IS delegation as an orienting lens to focus attention on how developers delegate a part of their responsibility for constructing the system's decision logic to the AI-based system during systems development, and how this again shapes the user-AI delegation. Later, Paper IV draws on temporal agency as a lens to better explain the observed development trajectories and their differences across projects. Paper V uses affordance realization and action possibilities mainly as a framing to theorize how developers realize and shape generative AI-augmented ISD, without treating these concepts as *ex ante* coding categories. Table 2 provides an overview of the qualitative case studies (Paper I to V), summarizing each paper's study focus, core design choices, empirical setting, primary data sources, and theoretical lens.

3.1.2 Data Sources and Collection

In exploratory, theory-building case study research, case sites are selected for their theoretical relevance and the opportunity to examine the focal phenomenon in depth (Benbasat et al., 1987; Eisenhardt, 1989b). Researchers, therefore, specify and bound the empirical setting, that is, define the population from which cases and embedded units of analysis are drawn. Such bounding, for example, by studying cases within a single organizational context, helps constrain contextual heterogeneity and clarifies the empirical scope of the emerging theory (Eisenhardt, 1989b). All qualitative papers in this dissertation draw on the same large telecommunications organization as the overarching case site. Using a common organizational setting provides a shared contextual background while allowing the empirical and analytical focus to vary across the case studies (Benbasat et al., 1987).

The case organization is a leading European telecommunications organization. The selected organization provides an ideal case site for studying phenomena in AI-based systems development and AI-augmented software development because it has developed extensive AI capabilities, that is, the ability to select, orchestrate, and leverage AI-based resources (Mikalef & Gupta, 2021). This includes data and technological foundations, AI-relevant technical and business skills, and organizational resources that enable AI-based projects to be coordinated. In addition, the organization maintains a broad portfolio of AI-based projects across business areas, including discriminative and generative AI-based systems development projects (Paper I to IV), as well as provides wide access to generative AI for augmenting software developers' work within the organization (Paper V).

Case study research commonly combines interviews as the primary data source with additional sources of evidence (e.g., documents and other materials) to strengthen the findings. Using multiple sources helps triangulate the findings and improve their validity (Benbasat et al., 1987; Dubé & Paré, 2003; Eisenhardt, 1989b). Across Paper I to V, the primary data source consists of semi-structured interviews conducted either on-site at the case organization or via video conference. All interviews were recorded and transcribed to enable fine-grained analysis. To substantiate and triangulate insights from the interviews, all qualitative papers additionally draw on project- and system-related materials, including project documentation and internal materials (e.g., presentations and intranet documentation) and, where available, system-related materials, that is, outputs that represent and document the AI-based systems and their development (e.g., demo materials and videos describing system features and aspects of the development process). Paper I to V use these materials to validate and deepen the understanding of the systems and development processes and to triangulate emerging findings.

Sampling in theory-building case study research should be done in a purposeful way. Cases, units of analysis, and informants are selected because they are expected to illuminate the phenomenon and help develop, elaborate, or challenge emerging theoretical accounts (Benbasat et al., 1987; Eisenhardt, 1989b). In multiple case research, case selection can follow replication logic. Cases are selected and analyzed to support either literal replication (where similar patterns are expected to recur across cases) or theoretical replication (where contrasting patterns are expected for theoretically specified reasons) (Benbasat et al., 1987). Theory building from cases is iterative in nature. Accordingly, data collection and analysis proceed in parallel, and additional cases or interviews are added when they help in further developing the insights. Sampling reaches closure when theoretical saturation is achieved, that is, when additional cases or interviews yield only marginal incremental insight (Eisenhardt, 1989b).

Paper I and II investigate five AI-based systems development projects as embedded units of analysis within the overarching case site and sample these based on the criteria of systems supporting joint user-AI problem-solving, and systems in productive use, or late-stage development. Based on similar criteria, Paper III samples three generative AI-based systems development projects. Paper IV treats projects as distinct cases and employs a replication-oriented expansion of the cases, beginning with discriminative AI-based projects and extending with additional projects (including generative AI-based projects), as earlier analysis indicated the need to refine emerging patterns. Paper V focuses on software developers as key informants and samples them based on differences in their needs for context when applying generative AI in their development work, that is, differing work situations in which developers require varying types and amounts of contextual information.

In Paper I to IV, informant selection follows a multi-perspective strategy to investigate the development of AI-based systems in the triad involving developers, users, and the agentic AI-based system. Thus, for each project, informants are sampled from both the development side and the

use side. This enabled an understanding of both developers' and users' accounts of how the AI-based system was developed and aligned with the work it would support. Additionally, this approach is consistent with established case research guidance to interview knowledgeable participants who hold different perspectives on the phenomenon to mitigate bias in interview-based research (Benbasat et al., 1987; Eisenhardt & Graebner, 2007).

In the qualitative studies, some projects and interviews are reused across the papers. For example, Paper I and Paper II draw on the same five projects, and Paper III overlaps with Paper IV on a part of the generative AI projects. However, each paper focuses on a distinct phenomenon and employs a different theoretical lens. Across all qualitative studies in this dissertation, 64 unique interviews were conducted, totaling 3,473 minutes (an average of 54 minutes per interview).

3.1.3 Data Analysis and Rigor

The data analysis for Paper I to V followed an iterative logic, in which data collection and analysis overlapped, and emerging insights informed subsequent sampling and refinement (Eisenhardt, 1989b). Interview transcripts were analyzed through iterative coding with constant comparison to organize qualitative evidence and to develop empirically grounded categories and relationships (Miles & Huberman, 1994). Coding was treated as an iterative analytic cycle, that is, linking data through iterative analysis to conceptual categories (Dubé & Paré, 2003; Saldaña, 2009).

All five qualitative papers applied a three-stage coding approach (open, axial, selective), adapted from Strauss and Corbin (1998), as an analytic technique to support the theory-building case study research in this dissertation. First, in open coding, interview transcripts were coded line-by-line to surface empirically grounded concepts, which were consolidated into first-order concepts. Second, in axial coding, first-order concepts were related to higher-level categories, and their relationships were specified. Third, in selective coding, the emerging categories were integrated around one or more core categories to form a coherent theoretical account and to refine category definitions and relationships. Across axial and selective coding, the theoretical lens served as a sensitizing device to guide abstraction and integration in Paper I, II, and IV, while remaining attentive to data-driven categories and without forcing the data into preconceived categories. Table 3 summarizes, paper by paper, how the three-stage-coding was conducted, including how sensitizing theoretical lenses guided the abstraction and integration.

To ensure rigor, the qualitative studies in this dissertation follow the four quality criteria for qualitative research: credibility, confirmability, dependability, and transferability (Dubé & Paré, 2003; Recker, 2021). Credibility concerns whether the proposed findings and conclusions are plausible and sufficiently substantiated by the empirical evidence. Confirmability concerns whether key findings are clearly grounded in the data and can be verified by others. Dependability concerns whether the study's procedures and analytic steps are transparent such that another

Table 3: Overview of the three-stage coding approach across Paper I to V

Paper	Open Coding	Axial Coding	Selective Coding
Paper I	Line-by-line coding to surface concepts related to knowledge boundaries, and development practices and artifacts to bridge the boundaries	Relating first-order concepts into higher-level categories The knowledge boundaries and boundary objects perspective served as sensitizing lenses to guide abstraction and specify relationships	Integrating the emerging categories around four interdependent core categories that jointly explain how knowledge boundaries are bridged in AI-based systems development
Paper II	Line-by-line coding focused on development practices in the triad of developers, users, and AI-based systems	Developing higher-level categories and relationships Cognitive fit served as a sensitizing lens for abstraction	Integrating categories around two core categories, that are, aligning the agentic IS artifact and aligning the user mental representation, to form the central explanatory account
Paper III	Line-by-line coding focused on development practices in the triad of developers, users, and generative AI-based systems	Relating first-order concepts into higher-level categories to capture how developers manage the user-AI interaction during development	Integrating categories into the core categories of enablement and control
Paper IV	Line-by-line coding to surface concepts related to how developers develop AI-based systems as agentic IS artifacts whose behavior they cannot fully specify or inspect	Relating first-order concepts into higher-level categories Delegation lens and, subsequently, an agency lens, which was introduced after temporal orientation became salient, sensitized how relationships were specified and compared across cases	Integrating categories into overarching categories around delegation-in-development, that are, appraisal-in-development, distribution-in-development, and coordination-in-development, as well as how delegation-in-development relates to user-AI delegation
Paper V	Line-by-line coding of developers' actions when using generative AI to augment them in their software development work, and developing first-order concepts around how developers realize and shape generative AI use in their work practices	Relating first-order concepts to higher-level categories to capture how possibilities of generative AI-augmented software development are realized and shaped under different conditions (different context needs)	Integrating emerging categories around the core categories of enacting and expanding

researcher could follow the same process and reach similar conclusions from the same materials. Transferability concerns whether the resulting findings can be carried to other settings.

Across Paper I to V, credibility was strengthened through the iterative, theory-building case logic in which data collection and analysis overlapped and emerging findings were progressively refined through within- and cross-case comparison. In the multiple case design (Paper IV), replication-oriented sampling further supported the credibility of emerging insights by seeking cases that replicate and extend patterns. Confirmability was supported by triangulating interview accounts with additional evidence (e.g., project documentation, system documentation, demo materials, presentations) to triangulate findings from the interviews and to use multiple sources of evidence to strengthen the evidential basis of findings. Dependability was supported by documenting data collection procedures in a case study protocol (including sampling logic, interview guide, interview procedures), by maintaining a case study database that organizes materials, such as transcripts, coding documentation, project documents, and by regularly discussing and reflecting on coding decisions and findings with other researchers. Finally, transferability was supported by drawing on theoretical lenses as orientation for the development of concepts and relationships and, where applicable, leveraging replication-oriented logic to strengthen the robustness of the emerging theoretical accounts.

3.2 Quantitative Research

3.2.1 Survey Research Design

In this dissertation, the quantitative study in Paper VI complements the qualitative case studies by examining relationships in AI-augmented ISD, specifically, how developers' use of generative AI relates to individual ISD-related work outcomes, and how developers' proactive job changes (job crafting) mediate this relationship. Quantitative research uses numerical data and statistical analyses to examine relationships among constructs and to evaluate theory-driven hypotheses, thereby refining theoretical explanations in light of empirical evidence (Chen & Hirschheim, 2004; Edmondson & McManus, 2007). In IS, this logic is often implemented through survey-based studies with hypothesis testing, in which theoretically motivated associations between constructs are operationalized through measures and evaluated using statistical analyses (Chen & Hirschheim, 2004; Recker, 2021). Building on prior work on IS use, job crafting, and work outcomes, the quantitative part of this dissertation therefore tests theoretically derived hypotheses about the relationships between developers' generative AI use, job crafting, and individual-level work outcomes in organizational ISD.

Survey research is an established quantitative IS method for examining relationships among theoretically defined constructs in natural settings, using standardized measurement (Pinsonneault & Kraemer, 1993; Recker, 2021). Paper VI employs a cross-sectional survey design, measur-

ing all constructs once for each participant. Cross-sectional surveys are appropriate when the research objective is to estimate how constructs relate at a particular point in time for the population of interest (Pinsonneault & Kraemer, 1993). This design aligns with the research objective of Paper VI to relate developers' generative AI use to individual work outcomes and to examine job crafting as a mediator. This requires collecting all focal constructs within the same observation window. The survey instrument captures the constructs defined in the research model and includes antecedents that theory suggests may account for variation in the relationships, enabling a statistically grounded test of the proposed research model (Pinsonneault & Kraemer, 1993).

As the case organization, we selected the internal IT service provider of a large European telecommunications organization. The internal IT service provider offers the full IT service portfolio and thus makes an important contribution to the organization's competitiveness and market success. It is responsible for the design, implementation, provision, and maintenance of all IT systems and IT infrastructure services required to support the business of the organization across all its business segments. The case organization introduced GitHub Copilot at the end of 2024.

Paper VI draws on a cross-sectional online survey conducted at the case organization. Data was collected using an online questionnaire administered over a period of four months (May - August 2025). The questionnaire was distributed to 508 employees in a defined organizational unit consisting of professional software developers. Consistent with the study's focus, participants were selected using purposive sampling: participation was open only to employees who (1) currently held a software development role and (2) actively used generative AI (GitHub Copilot) for development tasks during the period specified in the survey instructions. Before starting the questionnaire, participants were provided with an information sheet outlining the study purpose, voluntary participation, confidentiality, and intended data use. No personally identifiable data were collected, and responses were analyzed in an aggregated form. Data storage and processing followed data protection requirements. Of the 508 employees invited, 98 completed the survey, corresponding to a 19% response rate.

3.2.2 Measurement, Validity, and Analysis

Measurements link the theoretical model to empirical data by operationalizing constructs as observable indicators that can be quantified and analyzed (Recker, 2021). In quantitative IS research, this typically involves drawing on established multi-item scales when possible, transparently documenting item wording and response formats, and carefully justifying any adaptations to preserve construct meaning and measurement quality (Recker, 2021). In Paper VI, the survey operationalizes (1) developers' generative AI use, (2) job crafting in multiple dimensions, and (3) individual-level work outcomes (job satisfaction, work engagement, work meaning). In

addition, it includes theoretically motivated antecedent variables capturing job and individual characteristics relevant to technology use and job crafting (i.e., workload, autonomy, proactive personality, self-efficacy). All constructs were modeled as reflective latent variables using multi-item measures (Hair et al., 2017, 2021). Items were derived from validated scales from prior literature and assessed on a 7-point Likert scale unless stated otherwise.

Prior to model estimation, the data points were prepared through standard screening procedures. Responses were checked for completeness and consistency, indicator coding was harmonized (including the handling of reverse-coded items where applicable), and the data points were inspected for irregular response patterns that could undermine measurement quality (Recker, 2021). These steps ensure that data-handling problems were avoided.

To estimate the relationships among constructs, Paper VI applies partial least squares structural equation modeling (PLS-SEM). PLS-SEM is a variance-based SEM approach that estimates latent variable scores as weighted composites of their indicators and uses these scores to estimate the structural model (Chin, 1998; Hair et al., 2017). PLS-SEM is suitable here because it supports the simultaneous estimation of a measurement model and a structural model with multiple interrelated paths, including mediation (indirect paths), thereby enabling a statistically grounded test of the proposed framework.

Model evaluation followed established PLS-SEM guidance by assessing the measurement models first and interpreting the structural model only after satisfactory measurement quality had been established (Hair et al., 2021).

First, for all reflective measurement models in Paper VI, the established PLS-SEM guidelines to assess indicator reliability, internal consistency reliability, convergent validity, and discriminant validity are followed (Hair et al., 2021). Indicator reliability was examined via standardized outer loadings. Because the squared loading reflects the share of an indicator's variance explained by the construct, loadings of about 0.70 or higher indicate that the construct explains at least 50% of the indicator variance. Internal consistency reliability was assessed using Cronbach's alpha (α) and reliability coefficients recommended for PLS-SEM, particularly composite reliability (also called rho ρ). Values between 0.70 and 0.95 indicate satisfactory reliability, whereas values clearly above 0.95 may indicate item redundancy. In Paper VI, both Cronbach's alpha and composite reliability are reported. Convergent validity reflects the extent to which indicators of a construct share a high proportion of common variance. It was assessed via the average variance extracted (AVE), which is the mean of the squared loadings for a given construct (Hair, 2014). An AVE of 0.50 or higher indicates that the construct explains at least half of the variance of its indicators and is generally taken as evidence of convergent validity. Discriminant validity evaluates whether constructs are empirically distinct from one another. Two complementary criteria were used in Paper VI. First, the Fornell–Larcker criterion requires that the square root of each construct's AVE exceeds its correlations with all other constructs, indicating that a construct shares more variance with its own indicators than with other constructs

(Fornell & Larcker, 1981). Second, the heterotrait–monotrait ratio of correlations (HTMT) was examined (Henseler et al., 2015). HTMT values below 0.85 (strict criterion) or below 0.90 (more lenient criterion) provide evidence of discriminant validity.

Second, after measurement quality was established, the structural model was assessed following PLS-SEM guidelines by examining collinearity, the significance and relevance of path coefficients (including indirect effects), and the model's explanatory and predictive performance (Hair et al., 2021). Collinearity among the predictor constructs was checked using variance inflation factors (VIFs) for each endogenous construct. High collinearity inflates standard errors and makes it difficult to isolate each predictor's unique contribution. VIF values below 3 to 5 are generally considered acceptable, indicating no problematic multicollinearity (Hair et al., 2021). The central elements of the structural model are the path coefficients, which represent the strength and direction of the relationships between exogenous and endogenous constructs. Path coefficients were evaluated using bootstrapping and confidence intervals. In Paper VI, a coefficient is considered statistically significant when its 95% bootstrap confidence interval does not include zero. Explanatory power was reported using the coefficient of determination (R^2) for each endogenous construct. R^2 indicates the proportion of variance in an endogenous construct that is explained by its predictors. In PLS-SEM applications, R^2 values of around 0.75, 0.50, and 0.25 are often described as substantial, moderate, and weak, respectively, although acceptable levels depend on the research context and the complexity of the phenomenon studied (Hair et al., 2021). In Paper VI, R^2 values are reported for all job crafting dimensions and individual work outcomes to show how well generative AI use and control variables account for their variance.

3.2.3 Hypothesis Testing

Hypothesis testing in Paper VI relied on bootstrapping with 5,000 samples to obtain standard errors and confidence intervals for direct path coefficients and indirect effects (Hair et al., 2021). Bootstrapping repeatedly draws observations with replacement from the original dataset. The PLS-SEM model is re-estimated for each bootstrap sample, resulting in an empirical sampling distribution for each parameter estimate. A hypothesized effect was considered supported when the 95% bootstrap confidence interval for the corresponding estimate did not include zero (Hair, 2014; Henseler et al., 2015).

Mediation was tested by estimating direct and indirect paths within a single structural model and evaluating specific indirect effects using bootstrapped confidence intervals (Nitzl et al., 2016; Preacher & Hayes, 2008; Zhao et al., 2010). Moderation hypotheses were tested by incorporating latent-variable interaction terms following established PLS interaction modeling procedures and assessing the interaction effects via bootstrapping.

Taken together, Paper VI's cross-sectional survey, standardized measurement model, and PLS-

SEM analysis provide quantitative evidence on how developers' generative AI use relates to job crafting and, in turn, to individual-level work outcomes in organizational ISD.

4 Main Research Results

This cumulative dissertation comprises six research papers that examine information systems development (ISD) in the age of artificial intelligence (AI) from two perspectives: the development of AI-based systems as the output of ISD (Perspective I), and AI-augmented ISD in which AI augments developers' work within ISD processes (Perspective II). Each paper addresses one or more research questions and, together, the papers answer the overarching inquiry of how AI reshapes ISD. Table 4 provides an overview of the papers included in this dissertation. Section 4.1 documents my individual contributions to each paper in detail, followed by two sections (Section 4.2 and Section 4.3) that present the main results of the papers by perspective.

4.1 Statement of Individual Contributions and Co-Author Collaboration

This section documents my individual contributions for each paper included in this cumulative dissertation. My contributions are described in detail for each paper, including conceptualization, theory development, model development, data collection, data analysis, and writing.

Paper I is a single-author paper. For Paper II to VI, I served as first author. Across all six papers, I took primary responsibility for the end-to-end execution of each study, including study conceptualization, theory development, model development, data collection, data analysis, and writing. For Paper II to VI, my co-authors contributed through feedback, discussion, and review rounds at different stages of the study. Details of my contributions per paper are described in the following:

- **Paper I (single-author paper):** I was responsible for the conceptualization and theory development using knowledge boundary and boundary object lenses as sensitizing theoretical lenses and for developing the paper's conceptual account of the triadic development practices. I was responsible for conducting data collection, data analysis, and writing of the manuscript, including revisions.
- **Paper II:** I was responsible for the conceptualization and theory development using the cognitive fit lens as a sensitizing theoretical lens to theorize how to address the alignment challenges in the development of AI-based systems, and for developing the paper's alignment model. I was responsible for conducting data collection, data analysis, and writing of the manuscript, including revisions.
- **Paper III:** I was responsible for the conceptualization and theory development of generative AI-based systems development, and for developing the resulting conceptual ac-

Table 4: Overview of papers included in this dissertation

Perspective	Paper	RQ addressed	Publication / Manuscript Details & Status
Perspective I: Development of AI-based Systems	Paper I	RQ1, RQ2, RQ3	Wu-Gehbauer, M. (2025). “Developing Artificial Intelligence Systems – Bridging Knowledge Boundaries in the Triad.” <i>Academy of Management 2025 Proceedings</i> .
	Paper II	RQ2	Wu-Gehbauer, M., & Rosenkranz, C. (2024). “Towards a Framework for Developing Artificial Intelligence Systems for Human-Artificial Intelligence Augmentation.” <i>European Conference on Information Systems (ECIS) 2024 Proceedings</i> .
	Paper III	RQ2	Wu-Gehbauer, M., & Rosenkranz, C. (2025). “Between Enablement and Control – Generative Artificial Intelligence-based Systems Development.” <i>Hawaii International Conference on System Sciences (HICSS) 2025 Proceedings</i> .
	Paper IV	RQ3	Wu-Gehbauer, M., & Rosenkranz, C. “Delegation in Developing Artificial Intelligence-based Systems – How Developers Construct Conditions for User-Artificial Intelligence Delegation” <i>Status: Presented at the Paper Development Workshop AIS SIG DITE in 2025, journal submission planned in 2026.</i>
Perspective II: AI-augmented ISD	Paper V	RQ4	Wu-Gehbauer, M., & Rosenkranz, C. (2024). “Unlocking the Potential of Generative Artificial Intelligence – A Case Study in Software Development.” <i>International Conference on Information Systems (ICIS) 2024 Proceedings</i> .
	Paper VI	RQ5	Wu-Gehbauer, M., Wagner, F., Hennel, P., & Rosenkranz, C. “Developers’ Job Crafting and Individual Work Outcomes in Artificial Intelligence-augmented Information Systems Development” <i>Status: Conference submission planned in 2026.</i>

count. I was responsible for conducting data collection, data analysis, and writing of the manuscript, including revisions.

- **Paper IV:** I was responsible for the conceptualization and theory development using the IS delegation and temporal agency lenses as sensitizing theoretical lenses to develop the concept of delegation-in-development, and for developing the paper's conceptual model linking the developer-AI delegation during development to the user-AI delegation in use. I was responsible for conducting data collection and analysis, including cross-case synthesis, and for writing the manuscript, including revisions.
- **Paper V:** I was responsible for the conceptualization and theory development using the affordance realization lens as a sensitizing theoretical lens, and for developing the paper's conceptual account. I was responsible for conducting data collection, data analysis, and writing of the manuscript, including revisions.
- **Paper VI:** I was responsible for the conceptualization, theory development, developing the model (hypotheses, measurement model, and structural model), as well as for designing and conducting the cross-sectional survey. I was responsible for conducting the statistical analysis (PLS-SEM) and for the writing of the manuscript, including revisions.

4.2 Perspective I: Development of Artificial Intelligence-based Systems

Perspective I comprises Papers I to IV and examines AI-based systems development from a triadic view of developers, users, and the AI-based system. It analyzes how developing AI-based systems with agentic characteristics reshapes ISD by creating new boundaries, alignment challenges, and delegation challenges. It further examines how developers enact development practices to address these challenges.

4.2.1 Paper I: Developing Artificial Intelligence Systems – Bridging Knowledge Boundaries in the Triad

AI-based systems differ fundamentally from traditional rule-based information systems because they learn a probabilistic decision logic from data and thereby construct their own representations of a task domain (Baird & Maruping, 2021; Chollet, 2021; Murray et al., 2021). This learned decision logic may diverge from the decision logic of human experts and is only partially transparent to them (Faraj et al., 2018). As a result, the traditional ISD view, which treats users and developers as the main knowledge holders to be coordinated in a dyadic relationship (Chakraborty et al., 2010; Newman & Robey, 1992; Tesch et al., 2009), becomes insufficient for understanding how AI-based systems are developed for organizational use. Paper I addresses this gap by conceptualizing AI-based systems development as a triad of agents, involving developers, users, and the AI-based system. Specifically, Paper I examines knowledge-related

boundaries within this triad and how development practices are enacted to manage these boundaries.

Paper I draws on the relational notion of knowledge boundaries from Carlile (2004) and on the concept of boundary objects (Star & Griesemer, 1989) (see Section 2.4.1). Knowledge boundaries arise when actors hold different knowledge, face novelty, and are mutually dependent on each other's contributions. In the context of AI-based systems, these conditions occur not only between users and developers but also at their interfaces with the AI-based system.

In Paper I, AI-based systems are treated as if they were knowledge agents for three reasons: (1) they learn from a data basis that differs from that of human actors (Lawrence, 2017; Lyytinen et al., 2021; Raisch & Fomina, 2025), as a result, (2) they construct their own probabilistic decision logic from these data (Chollet, 2021; LeCun et al., 2015), and (3) their outputs are used as insights in knowledge work (Berg et al., 2023; Eloundou et al., 2023). This framing clarifies implications for developing AI-based systems. These characteristics mean that, at the interface with human actors, AI-based systems introduce knowledge-related boundaries in the development process. Boundary objects are artifacts that are sufficiently flexible to be adapted to multiple domains while retaining a common identity across them (Star & Griesemer, 1989). Paper I uses this lens to analyze how such knowledge-related boundaries are bridged in practice during AI-based systems development.

Empirically, Paper I uses a case study and examines multiple discriminative AI-based systems development projects. The analysis focuses on development practices that coordinate work within the triad during development. Paper I identifies (1) where knowledge-related boundaries arise across the development of the AI-based systems and (2) which development practices and boundary objects are used to bridge them.

The findings show that knowledge-related boundaries arise in four phases of developing AI-based systems, each defined by who holds critical knowledge in the triad and who needs access to it. In the data provisioning phase, users hold domain and task knowledge about which data and attributes matter, while developers and the AI-based system need access to this knowledge to train the model. Joint work on system-generated traces and user-shaped data structures turns these data objects into boundary objects through which user knowledge is made usable for developers and the AI-based system.

In the algorithm implementation phase, developers hold technical knowledge about model families, parameters, and trade-offs, for example, how broadly the model should make recommendations versus how accurate these recommendations must be. Users, in turn, need access to this knowledge so they can judge whether the model behavior fits their task, adjust thresholds and target groups, and provide informed feedback. Here, Paper I finds that the AI-based system itself functions as a boundary object: simulations, what-if analyses, and model runs on concrete cases make algorithmic behavior inspectable, enabling users and developers to jointly question,

negotiate, and refine model settings.

In the learning phase, where algorithmic opacity becomes salient, the AI-based system learns a probabilistic decision logic that is only partially intelligible to both users and developers. Both groups need ways to engage with this learned logic so that they can manage the knowledge-related boundary around it, assess its fit with task requirements, and govern its use. Across the projects studied, feature importance rankings and related explainability artifacts often functioned as boundary objects that render parts of this internal logic interpretable. Through joint interpretation, developers manage their own uncertainty about learning and partly opaque AI-based systems, while users connect the AI-based system's decision logic to their own knowledge.

In the output interpretation and actionability phase, users hold knowledge about business rules, task constraints, and business performance goals, while the AI-based system produces outputs that must be aligned with this knowledge. Developers depend on users' task and decision logic to embed the AI-based system meaningfully into business processes. Here, business rules and key performance indicators operate as boundary objects. They encode user task and decision logic, frame where and how the AI-based system's outputs can inform actions, and structure how the impact of human-AI collaboration is evaluated. Table 5 and Table 6 summarize the key findings. Table 5 outlines the knowledge-related boundaries that arise across the phases of developing AI-based systems, while Table 6 summarizes the corresponding development practices and boundary objects through which these boundaries are bridged.

In summary, Paper I answers RQ1 by specifying how the development of AI-based systems can be understood as a process of managing knowledge-related boundaries within a triad of developers, users, and the AI-based system. It shows, first, that knowledge-related boundaries arise not only between human actors but also between humans and the AI-based system, with boundaries at the interface to the AI-based system across development phases. Second, it identifies phase-specific development practices and associated boundary objects through which these boundaries are bridged. In doing so, Paper I extends ISD research beyond the traditional user-developer dyad and advances a triadic, boundary-bridging view of developing AI-based systems. Building on this triadic view, Paper I addresses RQ2 by showing how developers sustain joint human-AI problem-solving through complementary practices that align the AI-based system with task demands while also supporting users in interpreting and acting on the system's learned decisions and outputs. Paper I further addresses RQ3 by identifying a knowledge-related boundary between developers and the AI-based system and showing how developers must find ways to evaluate and control the system's learned decision logic and resulting behavior during development.

Table 5: Knowledge-related boundaries along key phases of AI-based systems development

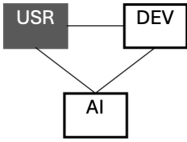
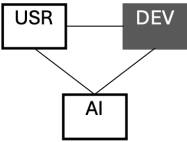
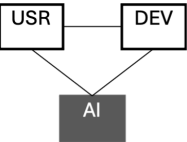
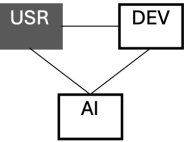
Knowledge Boundary	Phases of AI-based Systems Development			
	Input: Data Provisioning	Input: Algorithms	Learning: Algorithmic Opacity	Output Interpretation and Actionability
Knowledge boundary description	Developers and the AI-based system depend on domain-relevant input data held and interpreted by users	Algorithm implementation depends on the developers' expertise, inaccessible to users	AI's decision logic, learned from data, is opaque to users and partially to developers	AI outputs resulting from algorithmic optimization must be reconciled with domain-specific goals, business rules, and task constraints
Direction of effective boundary				
Impact of boundary	Impeding developers' ability to identify and prepare task-relevant data for the AI-based system to learn from	Impeding users' ability to comprehend key characteristics of AI-based system	Impeding ability of users and developers to comprehend AI-based system's decision logic	Impeding users' ability to act on the AI-based system's outputs in line with organizational goals and task requirements
Knowledge resides in x Knowledge to be accessed by y				

Table 6: Development practices and boundary objects in AI-based systems development

Knowledge Boundary	Phases of AI-based Systems Development			
	Input: Data Provisioning	Input: Algorithms	Learning: Algorithmic Opacity	Output Interpretation and Actionability
Development practice	Integrating user knowledge by selecting and structuring input data drawing on user domain knowledge	Integrating developer knowledge to configure algorithms and expose the algorithmic behavior of the AI-based system	Integrating AI's decision logic by tracing which attributes the AI-based system relies on most strongly for its outputs	Integrating user knowledge in the form of business rules and key performance indicators into the AI-based system
Impact of boundary	System-generated data, where users are extraneous to the data production User-shaped data, where users are involved in the data production	AI-based system and the simulations / what-if analyses of input-output relationships	Data objects generated by the AI-based system, such as feature importance scores and key driver attributes, which make the model's main decision drivers visible	Knowledge objects in the form of business rules that guide how the AI-based system generates outputs and how these may be used, and in the form of KPIs that evaluate and control the task performance in human-AI collaboration

4.2.2 Paper II: Towards a Framework for Developing Artificial Intelligence Systems for Human-Artificial Intelligence Augmentation

As AI-based systems increasingly augment rather than replace human work in organizations (Jain et al., 2021), it becomes critical to understand how they should be developed so that users can perform organizational tasks effectively with these systems. Prevailing ISD approaches have largely been framed for deterministic, rule-based systems and specification-driven methods (Berente et al., 2021; Byrd et al., 1992; Herbst, 1996; Matook et al., 2021). By contrast, AI-based systems learn patterns from task-related data and are opaque in how inputs map to outputs. They are agentic and act autonomously without human intervention (Baird & Maruping, 2021). This shift introduces distinctive challenges in aligning these systems to a concrete organizational task performed jointly with humans. Whereas rule-based systems allow developers to encode task logic *ex ante*, aligning agentic and learning systems to the task requires different development practices. Paper II, therefore, examines how developers enact such development practices to align the AI-based system with joint human-AI problem-solving tasks, and, reciprocally, how users' understanding and use of the system are shaped during development so that outputs become intelligible and actionable. The setting studied is human-AI augmentation, where users collaborate with AI-based systems to perform organizational tasks (Raisch & Krakowski, 2021).

Paper II conducts an exploratory case study, studying multiple discriminative AI-based systems development projects. It uses cognitive fit as the guiding lens (Vessey & Galletta, 1991) to identify development practices that enable users to work effectively with the system. Cognitive fit is an appropriate lens because it links performance gains to the alignment of task demands with the system's problem representations. Here, the problem representations of the AI-based system refer to its model's learned representations of the task and task domain as externally made available to the users via the system's interface and development practices, for example, articulations of the system's decision logic or system outputs such as scores or labels. Alignment of these system problem representations with task demands supports users in forming a mental representation for task solution (see Section 2.4.2).

The findings show that developers enact two interdependent streams of development practices. Practices directed at the AI-based system shape its behavior and outputs so that its problem representations align with the task demands. Practices directed at users enable their ability to understand how the AI-based system functions and how to act with it for problem-solving. Crucially, Paper II reveals that these two streams of development practices exhibit an iterative interplay. Alignment of the AI-based system depends on the internal representations of the users: Only when users gain an internal understanding of the AI-based system as well as of how to perform human-AI problem-solving, can they provide feedback that is essential for model refinement and for contextualizing the AI-based system (e.g., embedding business rules and constraints).

Conversely, understanding on the user side hinges on the alignment of the AI-based system with the task context: Only when model engineering and contextualization render the system's behavior and outputs aligned with the task can users form mental representations of acting with the system.

Developers play a central role in coupling these two development streams. On the user side, developers work in two complementary ways. First, they surface the AI-based system's learned problem representations and make them intelligible to users, for example, by exposing input-output relations through explanations of attributes influencing the outputs or by demonstrating system behavior via simulations. These practices build users' internal understanding of how the AI-based system functions. Second, developers support users in establishing when and how to apply the system in their work, for example, by linking joint human-AI problem-solving to business outcomes. Together, these practices support users in forming the mental representation needed for problem-solving with the AI-based system and to assess whether the system's external problem representations actually match task demands. When users detect misfits, such as missing attributes, inappropriate thresholds, or absent constraints, they provide feedback. Developers then use the feedback for model engineering and contextualization to improve the match between the system's external problem representations and the task.

Developing AI-based systems is thus conceptualized by Paper II as a mutual, iterative alignment of the AI-based system's problem representations and users' mental representations of how the AI-based system functions and how to act with it. Based on this, Paper II develops a theoretical framework for developing AI-based systems for joint human-AI problem-solving shown in Figure 4.

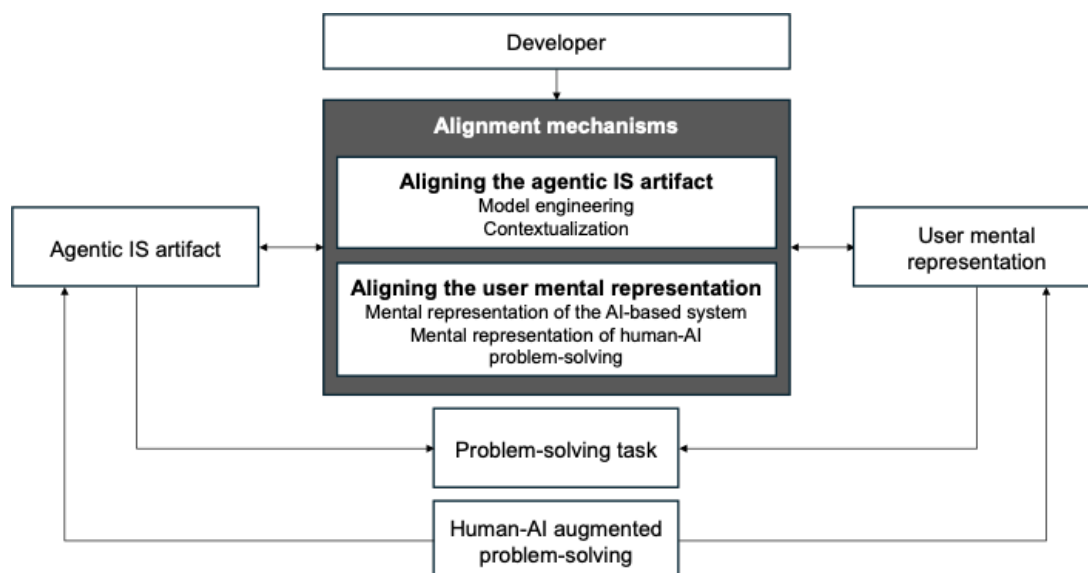


Figure 4: Framework for the development of AI-based systems for human-AI problem-solving

The proposed theoretical framework specifies that, through developers' iterative development practices, the problem representation of the AI-based system is shaped to converge towards an

alignment with the demands of the human-AI problem-solving task. In parallel, users' understanding is shaped so that the system's outputs can be interpreted and acted upon. Effective action with the AI-based system is enabled by this alignment of system behavior and outputs, on the one hand, and users' understanding, on the other hand. Put differently, Paper II finds that cognitive fit, which underpins effective human-AI task performance, emerges when iterative development practices jointly align the system to the task and make its behavior and outputs intelligible and actionable for users.

In summary, Paper II answers RQ2 by specifying how developers align AI-based systems with joint human-AI problem-solving tasks while, in parallel, shaping users' understanding of the system during development. Paper II demonstrates how these development practices are iteratively coupled to support effective human-AI problem-solving.

4.2.3 Paper III: Between Enablement and Control – Generative Artificial Intelligence-based Systems Development

Generative AI permeates organizations where it is integrated into information systems to augment knowledge work and enable human-AI problem-solving, that is, users accomplishing organizational tasks jointly with generative AI (Alavi et al., 2024; Benbya et al., 2024; Boussioux et al., 2024). Like discriminative AI-based systems, generative AI-based systems are self-learning, act autonomously, and are partly inscrutable (Berente et al., 2021). Yet, developers shape generative and discriminative AI-based systems through different means. Paper III examines how to develop generative AI-based systems that support users in accomplishing organizational tasks, focusing on large language models (LLMs) as the dominant form of generative AI applied in organizations (Singla et al., 2025). LLMs specialize in text understanding and text generation.

For development, the key difference is how developers can shape the system. LLMs are foundation models that come with predefined goals and pre-trained models (Radford et al., 2018; Touvron et al., 2023). As such, LLM-based systems are initially general-purpose systems, neither optimized for a specific organizational context nor for task-specific goals (Brown et al., 2020; Radford et al., 2018). Rather than adapting the model, an LLM-based system can be adapted by shaping the goals provided in the interaction with the system. Goal provisioning refers to how task goals and constraints are provided to the system via inputs. These inputs include prompts with task instructions and examples that convey user intent and task-specific knowledge. A user intent is the user's immediate aim in the current interaction with the generative AI-based system (Liu et al., 2023). Furthermore, additional task-specific knowledge can be made accessible to the generative AI-based system during output generation (Lewis et al., 2020).

Paper III employs a case study to examine how developers align generative AI-based systems with task context and requirements during development. Because generative AI produces outputs based on user or system inputs that convey intent, developing such a system, without adapt-

ing the underlying model, centers on designing and refining the user-AI interaction. The key finding of Paper III is that developers enact two core development practices to shape the user-AI interaction so that outputs align with the organizational task context and requirements: enablement and control. Both practices are enacted towards the users and towards the generative AI-based system.

On the user side, enablement practices, such as user training for interacting with the generative AI-based system, support users to express intent precisely and to provide well-structured inputs to the system. Control practices towards users complement this enablement. Developers enhance user prompts with system prompts and, when intent can be pre-specified, restrict user prompting. These user-side enablement and control practices aim at keeping inputs unambiguous and aligned with task requirements.

On the system side, enablement equips the system with task context at inference time. Developers curate context data, enrich sparse content, and supply examples so the generative AI-based system can access task-relevant knowledge when generating outputs. Control practices towards the system then steer the output generation. Developers encode rules to guide the output generation and to structure the outputs. Together, the purpose of these practices is to align the generative AI-based system with the task context and requirements. Additionally, upon output generation, user feedbacks are collected during the user-AI interaction to validate the system outputs against the task requirements. Insights from this validation feed back again both into the enablement and control practices towards users and towards the system.

In summary, Paper III answers RQ2 by specifying how developers align generative AI-based systems with joint human-AI problem-solving during system development. The study shows that developers enact enablement and control practices on both sides of the user-AI interaction, towards users and towards the system. Enacted in parallel, these development practices shape the human-AI interaction and, in turn, align system outputs with organizational task context and requirements.

4.2.4 Paper IV: Delegation in Developing Artificial Intelligence-based Systems – How Developers Construct Conditions for User-Artificial Intelligence Delegation

The behavior of AI-based systems cannot be fully specified in advance. Unlike traditional rule-based information systems, where developers encode decision rules *ex ante*, contemporary AI-based systems learn their own decision models from data (Baird & Maruping, 2021; Berente et al., 2021; Murray et al., 2021). Their behavior is partially opaque in that the learned input-output relationships cannot be fully determined *ex ante* or fully understood by developers *ex post* (Burrell, 2016; Faraj et al., 2018; Zhang et al., 2021). This changes the long-standing ISD assumptions that developers fully determine system behavior (Berente et al., 2021; Byrd et al.,

1992; Matook et al., 2021). For this dissertation, this raises the question of how developers can develop such learning and partially opaque systems, and how their ways of development condition how users engage with the systems in use.

Paper IV addresses this question by adopting a delegation lens (see Section 2.4.3). Building on the delegation framework from Baird and Maruping (2021), which conceptualizes human-AI use as a dyadic delegation configuration between users and an agentic IS artifact, Paper IV foregrounds developers as actors who configure the agentic attributes of the AI-based systems. In traditional ISD, developers inscribed decision logic directly into the system. With AI-based systems, developers delegate parts of this traditional responsibility to the system's learning processes, whose resulting decision model they can neither fully specify *ex ante* nor exhaustively inspect *ex post*. Developers thus find themselves in a dual delegation situation: they (1) delegate a part of their own responsibility for constructing the decision logic to the AI-based system, that is, letting the AI-based system learn its decision model, and (2) configure how users will later be able to delegate tasks and responsibilities to the AI-based system in use. A delegation lens is therefore appropriate because it makes visible how developers treat the AI-based system as an agent in development, one to whom they transfer rights and responsibilities under uncertainty, and how this shapes the conditions for later user-AI delegation.

Paper IV builds on an exploratory multiple case study of eight AI-based systems development projects, covering both discriminative AI-based systems and generative AI-based systems. The findings reveal two coupled delegation processes. Beyond the user-side delegation at use time theorized by Baird and Maruping (2021) and framed as a dyadic user-AI relationship, Paper IV shows that developers delegate during development as well. Paper IV theorizes this process as delegation-in-development: Developers transfer a part of their traditional responsibility for constructing the system's decision logic to the AI-based system, based on an appraisal-in-development that considers user capabilities, AI capabilities, and their compatibility. Developers then govern this responsibility transfer through distribution-in-development, which regulates the delegation *ex ante*, and through coordination-in-development, which controls and monitors the AI-based system *ex post*. These practices ensure that outputs and actions of the AI-based system align with desired goals.

Furthermore, Paper IV finds that, through delegation-in-development, developers configure how users will be able to appraise and coordinate the AI-based system in use. Developers adopt a temporal orientation (iterational, projective, or practical-evaluative) (Emirbayer & Mische, 1998) during delegation-in-development. An iterational past-oriented orientation involves the selective reactivation of prior patterns of thought and action, as incorporated in practical activity, thereby giving stability and continuity. A projective future-oriented orientation encompasses the imaginative exploration of possible future trajectories of action, in which received structures of thought and action may be creatively reconfigured in light of actors' desires for the future. And a practical-evaluative present-oriented orientation refers to the situated capacity of actors

to make practical judgements among alternative possible trajectories of action in response to the emerging demands, dilemmas, and ambiguities of present situations. Paper IV finds that the temporal orientation that developers adopt in delegation-in-development is mirrored at use time in user-AI delegation. Iterational delegation-in-development yields iterational conditions for user appraisal and coordination, projective yields projective, and practical-evaluative yields practical-evaluative. Figure 5 synthesizes the interplay between the processes of delegation-in-development and delegation-in-use.

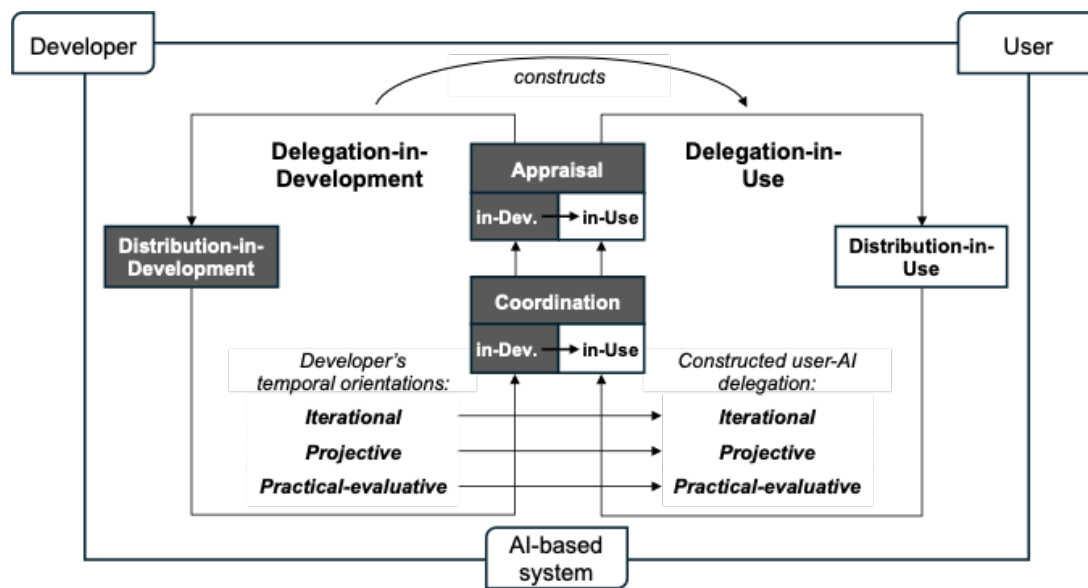


Figure 5: Model of two delegation cycles in the development of AI-based systems: Delegation-in-development & delegation-in-use

Thus, Paper IV demonstrates that the developers' temporal orientation (iterational, projective, practical-evaluative) is the key differentiator in delegation-in-development. Delegation-in-development in a specific orientation constructs the user-AI delegation through the corresponding orientation. This orientation determines what and how much is delegated, how delegation is governed, and where and to what extent user involvement is placed across development and use. Table 7 summarizes how each orientation determines appraisal-, distribution-, and coordination-in-development, while Table 8 shows how these choices materialize for users and shape the subsequent user-AI delegation in use.

In an iterational orientation, developers delegate narrow parts of the decision logic construction and maintain tight control loops with ongoing transparency and interpretability. This constructs a corresponding user appraisal and coordination that is iterationally oriented to align the AI's decision logic with established user work practices proven effective in the past. In a projective orientation, developers delegate more expansively and steer primarily by outcomes. They endow the AI-based system with broader learning autonomy and judge the AI-based system against business metrics. This constructs a projective appraisal and coordination at use-time oriented toward evaluating the performance of the projected new way of work enabled by the AI-based

Table 7: Delegation-in-development per developers' temporal orientation: summary of concepts and definitions

Concept	Iterational	Projective	Pactical-Evaluative
Appraisal-in-Development	<i>Developers' appraisal from a temporal stance from which they appraise user and AI capabilities for task completion and the compatibility between users' and AI's decision-making processes.</i>		
	Iterationally oriented Appraisal	Projectively oriented Appraisal	Practical-Evaluatively oriented Appraisal
	Iterationally oriented appraisal is oriented toward prioritizing users' work practices and the alignment of AI's compatibility with them.	Projectively oriented appraisal is oriented toward prioritizing AI's superior capabilities to envision new ways of work.	Practical-evaluatively oriented appraisal is oriented toward situated evaluation of the AI-based system and compatibility in the user-AI interaction.
Distribution-in-Development: Contracting	<i>For the transfer of developers' responsibilities of the decision model construction to AI, developers establish desired goals and how to achieve them ex ante to govern this transfer and be able to identify misalignments.</i>		
	Behavioral Contracting	Outcome Contracting	Input-Output Contracting
	Governance of how AI makes decisions to align AI's decision process with users' decision process.	Governance of what must be achieved by AI to align with desired business outcomes.	Governance of instance-level input-output fit.
Distribution-in-Development: Regulating	<i>For the transfer of developers' responsibilities of the decision model construction to AI, developers regulate the possible actions of the AI-based system by setting boundaries.</i>		
	Regulating Decision Logic	Regulating Frame Conditions for Learning	Regulating User-AI Interaction
	Regulating by encoding known patterns and by constraining model settings based on user practices.	Regulating by optimizing input and model performance for autonomous learning.	Regulating by pre-structuring user-AI interaction with room for situated user adjustments.
Coordination-in-Development	<i>Developers' coordination activities during development to monitor and update the AI-based systems for alignment with the desired goal defined by developers for the responsibility transfer.</i>		
	Coordination of Decision Logic Alignment	Coordination of Outcome Alignment	Coordination of Input-Output Fit
	Inspection and update of decision logic enabled by interpretable models.	Inspection of outcomes of task completion using business metrics.	Evaluation of input-output fit at the instance level of the user-AI interaction.

system. In a practical-evaluative orientation, developers delegate flexibly around instance-level fit. They emphasize input-output traceability, and design for situated adjustments in the moment of interaction. This constructs practical-evaluative appraisal and coordination for users at the instance-level of the user-AI interaction.

Table 8: Construction of user-AI delegation per developers' temporal orientation: summary of concepts and definitions

Concept	Iterational	Projective	Pactical-Evaluative
Development-Constructed User Appraisal	<i>Constructed user appraisal during user-AI delegation by delegation-in-development.</i>		
	Informed by Transparency of Decision Logic	Uninformed	Informed by Input-Output Transparency
	Transparency on system reasoning at use time with decision logic pre-aligned to user practice in development.	No transparency into the reasoning of the AI-based system, and no alignment of the AI's decision logic to user practice in development.	Transparency on input and output per user-AI interaction.
Development-Constructed User Coordination	<i>Constructed user coordination during user-AI delegation by delegation-in-development.</i>		
	User Coordination by Design	User Coordination Unsupported by Design	Interactional User Coordination by Design
	Built-in coordination for users at use time to review and initiate adjustments of the AI-based system.	No user-side coordination built into the system at use time.	Built-in, instance-level coordination for users at use time to review and adjust per output during the interaction.

Across the three orientations, what varies is how much responsibility is transferred to the AI-based system, where governance is anchored (check of decision logic versus outcome measurement versus interactional fit) and how this constructs user appraisal and coordination in the corresponding orientation for the user-AI delegation at use time. Developers' delegation-in-development thus sets the conditions under which use-time delegation unfolds, and the orientation of the constructed user-AI delegation corresponds to the temporal orientation during delegation-in-development.

In summary, Paper IV answers RQ3 by conceptualizing the development of AI-based systems, whose behavior can neither be fully specified in advance nor fully inspected ex post, as delegation-in-development. It shows that through delegation-in-development practices, developers both manage the AI-based system during development and construct the subsequent user-AI delegation. In doing so, Paper IV extends delegation theory from a dyadic, use-time perspective to a triadic view that conceptualizes developer-AI delegation during development and links it to user-AI delegation in use.

4.3 Perspective II: AI-augmented Information Systems Development

Perspective II comprises Paper V and Paper VI and examines how generative AI reshapes ISD by augmenting developers' work within ISD processes. It analyzes how developers realize and shape AI-augmented ISD in their development practices. It further examines how generative AI use relates to developers' job crafting and individual-level work outcomes in organizational ISD.

4.3.1 Paper V: Unlocking the Potential of Generative Artificial Intelligence – A Case Study in Software Development

Developers in organizations who use generative AI in their ISD processes face a tension between, on the one hand, powerful but general-purpose LLMs trained predominantly on public text-based data and code (Touvron et al., 2023) and, on the other hand, complex, historically grown enterprise application landscapes with idiosyncratic architectures, legacy systems, and project-specific constraints (Banker & Slaughter, 2000; MacCormack & Sturtevant, 2016; Monaghan & Bass, 2020). Paper V focuses on software development activities as a core component of ISD, understood as the set of development tasks through which software systems are specified, designed, implemented, tested, and maintained (Meyer et al., 2017; Sommerville, 2015), and examines how software developers respond to this tension in their everyday work when applying generative AI. Existing research shows which development tasks LLMs can support and how well they perform in these tasks (e.g., Banh et al., 2025; Weber et al., 2024). Yet, because general-purpose LLMs are not trained on the specific system context data of a given organization, such as proprietary code bases, their outputs can be inappropriate when applied in the development of specific enterprise systems in the organization (Huang et al., 2025; Ozkaya, 2023). As a result, developers must shape how generative AI is used so that it fits their particular development context and tasks. What remains underexplored is how software developers in organizational settings shape their work with such general-purpose generative AI-based tools within ISD processes to obtain support for their development tasks. Paper V addresses this gap by investigating how software developers realize and shape generative AI-augmented ways of accomplishing their development tasks within organizations.

To examine this, Paper V conducts a case study at the internal IT provider of an international telecommunications organization that introduced several LLM-based generative AI tools to support software developers in their work. Paper V adopts the lens of action possibilities, building on the affordance concept as the possibilities for goal-oriented action that a technical object affords to a specified user group, given their goals (Markus Silver, 2008). Because LLM-based tools are general-purpose technologies that can, in principle, be adapted to many different development tasks (Ahmed & Devanbu, 2022; Bird et al., 2023; Yuan et al., 2023; Zhang et al., 2023), this lens is suitable for conceptualizing generative AI as offering potential action pos-

sibilities for accomplishing software development tasks. Paper V focuses on how developers, in their ISD practices, realize and shape these action possibilities of generative AI augmented ways of working (Strong et al., 2014).

The findings show that software developers in an organizational setting realize and shape generative AI-augmented ways of working in their ISD processes in line with their goals and task contexts. Paper V identifies two mechanisms that operate at different goal levels and in different task contexts: enacting mechanisms for context-light tasks guided by individual-level goals and expanding mechanisms for context-heavy tasks guided by company-level goals.

In enacting mechanisms, individual-level goals are essential, such as increasing individual coding efficiency. Here, developers apply generative AI to context-light development tasks, that is, tasks that require little or no additional system-specific context, such as generating standard coding functions or unit tests. Paper V shows that developers realize generative AI support in these cases through a set of development practices. They align generative AI use with their own objectives, interact with generative AI by iteratively structuring and refining task instructions (e.g., decomposing tasks into subtasks, experimenting with different inputs), and validate generative AI outputs against their own expertise, coding standards, and using established practices such as code reviews and testing.

In expanding mechanisms, company-level goals are foregrounded, for example, accelerating complex and time-consuming development work to create organization-wide value, such as faster delivery or cost reductions. Developers here work with generative AI on context-heavy tasks that require substantial system- and organization-specific context, such as explaining custom code or translating legacy code. Paper V shows that, to realize generative AI support in these settings, developers instantiate a different set of development practices. They align generative AI use with company-level objectives, design enhanced contextualization that provides generative AI with the additional system and organizational context required, thereby creating contextualized generative AI-based artifacts, and control the quality and reliability of these artifacts through practices such as piloting, testing, and ongoing monitoring based on user feedback.

Paper V further shows that enacting and expanding mechanisms are intertwined. Expanding mechanisms, guided by company-level goals in context-heavy task domains, produce contextualized generative AI-based artifacts, for example, artifacts that embed legacy code, architecture knowledge, or security rules, thereby opening up additional ways of acting for individual developers. These newly created possibilities for action can then be realized through enacting mechanisms, provided they align with developers' individual-level goals and local task requirements. In this way, expanding mechanisms reshape how the underlying general-purpose LLM can be worked with in the organization, broadening the space of action possibilities that can subsequently be enacted in everyday software development work.

In summary, Paper V answers RQ4 by specifying enacting and expanding as two mechanisms,

each instantiated through distinct software development practices, through which developers realize and shape generative AI-augmented ways for accomplishing software development tasks in organizational ISD settings.

4.3.2 Paper VI: Developers' Job Crafting and Individual Work Outcomes in Artificial Intelligence-augmented Information Systems Development

Employees who are satisfied with their jobs, feel engaged in their work, and experience their work as meaningful tend to show higher performance (Christian et al., 2011; Hackman & Oldham, 1976; Judge et al., 2001). For software developers, non-technical factors such as job enthusiasm and job satisfaction are among the strongest factors correlating with their productivity (Murphy-Hill et al., 2019; Storey et al., 2019). Yet, most empirical research on the implications of generative AI in ISD examines productivity effects using technical performance indicators, such as the number of completed development tasks or time to complete standardized programming assignments (Cui et al., 2024; Peng et al., 2023; Ziegler et al., 2024). Therefore, Paper VI investigates, in an organizational ISD setting, how generative AI use relates to developers' individual-level work outcomes.

Because generative AI is widely discussed as fundamentally transforming developers' jobs and work practices (Nguyen-Duc et al., 2023; Sætra, 2023; Ulfsnes et al., 2024), Paper VI adopts a job crafting perspective to capture how developers actively reshape their work as they integrate generative AI (see Section 2.4.4). Job crafting is understood as employees' self-initiated changes to their jobs (Bruning & Campion, 2018; Wrzesniewski & Dutton, 2001). Building on the approach-avoidance, role-resource model of job crafting (Bruning & Campion, 2018), Paper VI differentiates between approach and avoidance forms of job crafting and examines four dimensions: approach role crafting, approach resource crafting, avoidance role crafting, and avoidance resource crafting. Prior research shows that job crafting is related to job satisfaction, work engagement, and work meaning (Bruning & Campion, 2018; Lichtenthaler & Fischbach, 2019; Petrou et al., 2012). Accordingly, Paper VI examines not only how generative AI use relates to developers' individual-level work outcomes, but also whether and how developers' job crafting behaviors mediate these relationships.

In addressing this question, Paper VI conducts a cross-sectional survey of software developers in a large European telecommunications organization who had begun using GitHub Copilot in their daily work. Following established guidelines for survey-based research and structural equation modeling (SEM) (Hair et al., 2019), the structural model is estimated with partial least squares SEM (PLS-SEM) on 98 complete responses. Generative AI use is conceptualized as the extent of GitHub Copilot use and is measured with three indicators, which are duration, frequency, and intensity of use (Venkatesh et al., 2008). Individual-level work outcomes are drawn on the concepts of work engagement, job satisfaction, and work meaning, and are measured using

validated scales from prior literature (Lawler et al., 1979; Schaufeli et al., 2006; Spreitzer, 1995). Additionally, prior job crafting research conceptualizes job complexity as a central job characteristic that shapes employees' crafting behavior and can relate to both approach and avoidance forms of job crafting (Bai et al., 2021; Tian et al., 2022). Building on this research, Paper VI extends this perspective to generative AI-augmented ISD by focusing on the job complexity of the work for which developers use generative AI. Job complexity here refers to the extent to which the generative AI-augmented work is non-routine, lacks repetitiveness, requires independent judgment, problem-solving, and creativity, and is measured based on the job complexity scale adapted from Gould (1979). Paper VI theorizes that when developers apply generative AI to more complex parts of their work, the association between generative AI use and job crafting becomes stronger, both in approach forms and in avoidance forms of job crafting. Accordingly, job complexity of generative AI-augmented work is modeled as a moderator of the relationship between generative AI use and the four job crafting dimensions.

The findings provide partial support for the hypothesis that generative AI use is positively related to the developers' individual-level work outcomes. Generative AI use shows a significant positive relationship with work engagement, whereas its relationships with job satisfaction and work meaning are not significant. In the study sample, where developers had an average around six months of experience with GitHub Copilot, Paper VI thus finds that higher generative AI use is associated with feeling more absorbed, vigorous, and dedicated at work, but not with feeling more satisfied or perceiving work as more meaningful.

The hypotheses that approach job crafting positively mediates, and avoidance job crafting negatively mediates the relationship between generative AI use and the three individual-level outcomes are not supported. The moderating effect of job complexity on the relationship between generative AI use and job crafting, however, shows an effect. The interaction between generative AI use and high job complexity shows a significant and positive relationship with avoidance resource crafting, whereas no such effect is found for low job complexity. This indicates that when developers use generative AI for high-complexity work, generative AI use is positively related to avoidance resource crafting.

Finally, Paper VI finds that proactive personality and job autonomy are predictors of job crafting. Proactive personality is positively related to approach role crafting and approach resource crafting, whereas job autonomy is positively related to avoidance resource crafting. Furthermore, Paper VI finds that approach and avoidance job crafting are differently related to individual-level work outcomes. Approach role crafting shows a significant positive relationship to job satisfaction, work engagement, and work meaning, while avoidance role and avoidance resource crafting show significant negative relationships to work meaning. The results of the PLS-SEM for the full set of path estimates, including controls and interaction effects are shown in Table 9.

In summary, Paper VI answers RQ5 by showing that generative AI use is positively related to

Table 9: Coefficients for PLS-SEM

Variables	AppRoC	AppReC	AvRoC	AvReC	Job Satis- faction	Work En- gagement	Work Meaning
Generative AI use	-0.010	-0.070	0.099	0.173	-0.022	0.159*	0.030
Generative AI use*Job complexity high	-0.108	-0.022	-0.011	0.269**			
Generative AI use*Job complexity low	-0.073	-0.016	-0.203	-0.162			
Generative AI use*Acceptance rate	-0.026	0.065	0.142	-0.166			
Generative AI use*Professional role identity	-0.048	-0.061	0.247	0.110			
Proactivity	0.393***	0.250**	0.021	0.068			
Workload	-0.019	-0.011	0.207	0.043			
Job autonomy	-0.015	0.038	0.196	0.309**			
Self-efficacy	0.142	0.107	-0.229	-0.089			
Attitude	0.067	0.026	0.075	-0.090			
Job role level	0.000	-0.091	0.045	0.155			
Organizational tenure	0.094	0.173	0.108	-0.081			
Approach role crafting (AppRoC)					0.486*	0.635**	0.644**
Approach resource crafting (AppReC)					-0.041	0.062	-0.062
Avoidance role crafting (AvRoC)					-0.012	0.051	-0.208*
Avoidance resource crafting (AvReC)					-0.097	0.004	-0.220**
Approach role crafting*Monitoring behavior					-0.136	-0.071	-0.203
Approach resource crafting*Monitoring behavior					0.068	-0.055	0.150
Avoidance role crafting*Monitoring behavior					-0.114	-0.023	0.064
Avoidance resource crafting*Monitoring behavior					-0.059	-0.073	0.011

Notes. N=98, ***p<0.001, **p<0.01, *p<0.05

work engagement but not to job satisfaction or work meaning. Developers' job crafting does not mediate the relationship between generative AI use and individual-level work outcomes in the study sample. Instead, the findings indicate that when generative AI is used for high-complexity work, its use is positively related to avoidance resource crafting.

5 Discussion

This dissertation addresses the overarching research question: *How does artificial intelligence reshape information systems development in organizations?* Across six papers, this dissertation answers this question and extends knowledge on ISD in the age of AI by clarifying how ISD is reshaped when developing AI-based systems as agentic IS artifacts, and how ISD is reshaped when AI augments developers' work in ISD processes. Section 5.1 outlines the theoretical implications of the dissertation's findings and shows how they advance IS research and theory. Section 5.2 presents the practical implications and offers guidance for organizations.

5.1 Theoretical Implications

This dissertation offers several contributions to research and theory, organized along the five research questions. Section 5.1.1 (RQ1) develops a triadic perspective on developing AI-based systems. Section 5.1.2 (RQ2) theorizes how developers align AI-based systems with joint human-AI problem-solving through coupled development practices directed at the system and at users. Section 5.1.3 (RQ3) advances a delegation perspective on AI-based systems development by conceptualizing delegation-in-development and showing how, through this, developers construct user-AI delegation in use. Section 5.1.4 (RQ4) explains how generative AI-augmented software development reshapes ISD processes. Finally, section 5.1.5 (RQ5) discusses implications of AI-augmented ISD for developers' individual-level work outcomes and their job crafting behavior.

5.1.1 Development of AI-based Systems from a Triadic Perspective

The first research question (RQ1) asks: *How are development practices enacted within the triad of developers, users, and the AI-based system during systems development?* **Paper I** addresses this by adopting a knowledge boundary and boundary object perspective on the systems development of AI-based systems.

First, Paper I extends ISD research beyond the traditional user-developer dyad. In ISD, prior work has largely focused on a dyad of human agents, where knowledge is assumed to flow from users to developers so that human decision logic can be encoded in rule-based information systems (Berente et al., 2021; Byrd et al., 1992; Herbst, 1996; Newman & Robey, 1992).

Paper I extends this view by conceptualizing the development of AI-based systems as a triad of agents, that are users, developers, and the agentic AI-based system, in which knowledge-related boundaries must be managed across all three agents.

This triadic view offers a more comprehensive understanding of where and how boundaries arise when developing AI-based systems. Existing research on AI-based system development has primarily emphasized integrating user domain knowledge into data as input for AI-based systems (e.g., Lebovitz et al., 2021; Sundberg & Holmström, 2024). By mapping knowledge-related boundaries across the triad, Paper I shows that this is only one part of the challenge. In addition, user task and decision logic required for task completion must be integrated into the AI-based system via business rules and key performance indicators (KPIs) that frame, for example, how model outputs are used. These rules and KPIs encode domain- and task-specific constraints and success criteria, enabling users to act on AI outputs in line with organizational goals and task requirements.

Second, Paper I problematizes the assumption that developers and the AI-based system form a single technical knowledge domain. Prior studies on the development of AI-based systems largely treat developers and the AI-based system as belonging to one technical domain, emphasizing workflows, pipelines, and quality assurance (Amershi et al., 2019a; Pfeuffer et al., 2023; Sculley et al., 2015; Studer et al., 2021). While developers and AI-based systems indeed operate on shared algorithms, data, and infrastructure, Paper I shows that a knowledge-related boundary nonetheless exists between developers and the AI-based system. It is primarily developers' choices, rather than user specifications, that determine the algorithms and learning architectures and thus shape how the AI-based system can learn. At the same time, the findings show that developers themselves struggle with the agentic and partly opaque characteristics of the AI-based system and must enact practices to overcome this boundary and manage their own uncertainty during the development of the AI-based system. In this way, Paper I highlights that boundary work in AI-based systems also occurs at the developer-AI interface.

Third, Paper I specifies which artifacts function as boundary objects when bridging knowledge-related boundaries in the triad. Existing IS and organization studies underscore the centrality of data and data objects in developing AI-based systems, showing how data is produced, curated, and assembled into organizational resources (Aaltonen et al., 2021; Alaimo & Kallinikos, 2021; Muldoon et al., 2024). Building on this work, Paper I shows that different boundaries during the development of AI-based systems require additional boundary objects. In the algorithm implementation phase, the AI-based system itself, together with simulations and what-if analyses of input-output relationships, serves as a boundary object. These system artifacts make algorithmic behavior visible on concrete cases, allowing users and developers to see how algorithms behave on their data and to refine model settings. In the output interpretation and actionability phase, business knowledge objects in the form of business rules and KPIs become essential boundary objects. They govern where and how outputs of the AI-based systems may inform action

and how their impact is evaluated. These findings qualify data objects as one class of boundary objects and foreground additional boundary objects, such as system artifacts and business knowledge objects, through which developers and users manage knowledge-related boundaries in the development of AI-based systems.

5.1.2 Development of AI-based Systems for Joint Human-AI Problem-Solving

The second research question (RQ2) asks: *How do developers enact development practices to align AI-based systems with joint human-AI problem-solving tasks, and how do they thereby shape users' understanding of the system?* **Paper I** addresses this question by showing that, as the development of AI-based systems moves through the phases of data provisioning, algorithm implementation, learning, and output interpretation and actionability, developers repeatedly encounter knowledge-related boundaries between users and the AI-based system. To keep joint human-AI problem-solving viable, developers enact boundary-bridging practices that alternately target the AI-based system and the users. On the one hand, developers align the AI-based system with task demands. On the other hand, they support users' understanding of the system. Paper I conceptualizes the development of AI-based systems as a boundary-bridging process, in which developers need to align the system's learned decision logic with users' task and decision logic. This extends ISD research on AI-based systems, which has often focused either on shaping data and models of the AI-based system (e.g., Asatiani et al., 2021; Kane et al., 2021; Studer et al., 2021) or on shaping the user understanding (e.g., Arrieta et al., 2020; Lebovitz et al., 2021). Paper I shows that these two strands are intertwined in practice. On this basis, Paper I provides the foundation for Paper II, which investigates in more detail how developers' practices towards the AI-based system and towards users jointly construct alignment in human-AI problem-solving tasks in organizations.

Paper II answers RQ2 by showing how cognitive fit is constructed for agentic, AI-based systems in collaborative tasks and, in doing so, advances understanding of ISD for AI-based systems. Cognitive fit for classical rule-based systems posits that problem-solving performance improves when the external problem representation provided by a system matches the information-processing demands of the task (Vessey & Galletta, 1991). Paper II extends cognitive fit from the setting of deterministic, rule-based technologies to agentic AI-based systems engaged in human-AI problem-solving. In collaborative settings, heterogeneous contributions from the AI-based system and the human must be integrated (Dellermann et al., 2019; Raisch & Krakowski, 2021). This entails the translation and transformation of knowledge, that is, building understanding of the meaning behind the contributions of the AI-based system, as well as learning the consequences of the collaboration in how they align with user goals and interests (Carlile, 2004). Paper II demonstrates that, under these conditions, exposing system outputs alone, such as scores or labels, as the system's external problem representation is insufficient to achieve cognitive fit. Developers must also externalize aspects of the AI-based system's internal func-

tioning, such as input-output relations, and the consequences of engaging in joint human-AI problem-solving, for example, through business outcomes. This enables users to construct internal mental representations for joint problem-solving with the AI-based system. In this way, Paper II adapts cognitive fit to a collaborative human-AI task setting.

Additionally, Paper II advances the understanding of ISD for AI-based systems. Because the AI-based system's problem representations are learned and partly opaque, Paper II shows that developers must enact two key streams of development practices: practices aimed at aligning the AI-based system's problem representations with task demands and practices aimed at rendering the system's learned problem representations intelligible and actionable to the users. Importantly, these practices enable users to apply their domain knowledge during development. Prior research has established the importance of domain knowledge for the development of AI-based systems (Gronsund & Aanestad, 2020; van den Broek et al., 2021). Paper II adds to this research by specifying how domain expertise becomes relevant. Only once developers have enabled user understanding of the contributions of the AI-based system for the joint human-AI problem-solving task can users apply their domain knowledge to determine the knowledge requirements for the task, and to determine whether the learned problem representation of the AI-based system matches what the task requires. This yields user feedback that developers then use to further refine the AI-based system.

Since the problem representations of the AI-based system are results of AI-based system's learning rather than specifications, and cannot be specified *ex ante* (Burrell, 2016; Zhang et al., 2021), the alignment of the AI-based system with task demands has to be achieved iteratively. Developers must repeatedly enact development practices towards users to enable understanding and towards the AI-based system to align its problem representations to better match task demands. In doing so, Paper II moves the focus of ISD for AI-based systems from system specification (Byrd et al., 1992; Herbst, 1996; Watson & Frolick, 1993) to a process of iterative alignment to achieve cognitive fit. Cognitive fit is constructed and maintained through developers' iteratively coupled development practices of aligning the AI-based system's problem representation and aligning users' mental representation.

Paper III addresses the second research question by studying the development of generative AI-based systems for human-AI problem-solving. It contributes to research on the development of AI-based systems by demonstrating that, for generative AI-based systems (LLM-based systems) developed without modifying the underlying model, development shifts from a model-centric development to a development centered on the user-AI interaction. In model-centric development, the focal development activities center on the machine learning (ML) pipeline, including data preparation, feature engineering, model and algorithm selection, model training, as well as model evaluation and monitoring (Amershi et al., 2019a; Breck et al., 2017; Polyzotis et al., 2018; Sculley et al., 2015). The learned model functions as a key component of the AI-based system that represents learned causalities of how variables are predicted to interact (Russell

& Norvig, 2016). The model guides the decision-making of the AI-based systems (Baird & Maruping, 2021) and may be opaque (Burrell, 2016). Within the model-centric development, the implications of AI-based systems on humans are addressed at the model level. For example, existing research shows how inscrutable models are enveloped with sociotechnical boundary conditions to ensure human accountability (Asatiani et al., 2021), or how explainability techniques render models more understandable to humans (Arrieta et al., 2020).

Paper III extends this research by showing that, for generative AI-based systems supporting organizational tasks, the development focus shifts from the model to the user-AI interaction. On the system side, beyond enablement via access to task-relevant context knowledge at inference time, control practices become central to steer the system's behavior so that its interaction and output generation adhere to business rules and task logic. On the user side, developers likewise enact enablement and control practices to improve the quality of goal provisioning, ensuring that what users provide in interaction and the resulting system outputs align with task requirements. Paper III conceptualizes this interaction-centric development as a development approach for LLM-based generative AI systems.

A further contribution of Paper III concerns how the alignment of the AI-based system with the human-AI problem-solving task is evaluated. In a model-centric development approach, evaluation draws on model-level performance indicators (e.g., Lebovitz et al., 2021) or on the interpretability of the model, explaining how inputs are related to outputs (e.g., van den Broek et al., 2021). In contrast, for the development of generative AI-based systems, Paper III shows that the locus of evaluation expands to the user-AI interaction. Developers rely on situated user evaluation, where users validate during user-AI interaction whether outputs fit task requirements. Paper III thus identifies situated evaluation during user-AI interaction as an important element of developing LLM-based generative AI systems.

5.1.3 Development of AI-based Systems from a Delegation Perspective

The third research question (RQ3) asks: *How do developers enact development practices when developing AI-based systems whose behavior they cannot fully specify or inspect, and how do these practices condition how users engage with these systems?* **Paper I** makes an initial contribution by showing that developers themselves face a knowledge-related boundary with the AI-based system during development, particularly in the learning phase. Here, the AI-based system acquires a probabilistic decision logic that is only partially intelligible to developers and cannot be fully specified or inspected. This challenges prior studies on the development of AI-based systems, which largely treat developers and the AI-based system as belonging to a single technical domain (Amershi et al., 2019a; Pfeuffer et al., 2023; Sculley et al., 2015). Paper I shows instead that a developer-AI boundary emerges during systems development. Developers experience uncertainty about the system's agentic and partly opaque behavior and must

find ways to engage with, evaluate, and control the learned logic to judge its task fit and retain control. Paper I contributes to the literature on ISD for AI-based systems by highlighting that development work involves managing a knowledge-related boundary at the developer-AI interface. These insights lay the basis for Paper IV, which examines in more detail how developers deal with these challenges and how their ways of working with the AI-based systems shape how users later engage with these systems in use.

Paper IV answers RQ3 by reframing the development of AI-based systems as a site where human-AI delegation is constructed and, in doing so, advances the delegation perspective, the literature on algorithmic learning, and research on ISD for AI-based systems.

First, Paper IV extends IS delegation theory (Baird & Maruping, 2021) from a dyadic, use-time perspective to a triadic view of delegation that spans developers, users, and AI-based systems across development and use. Prior research predominantly conceptualizes delegation as a dyadic phenomenon between users and the agentic AI artifact at use time (e.g., Baird & Maruping, 2021; Jussupow et al., 2021; Lebovitz et al., 2022; Zhang et al., 2021). Paper IV reveals that, under conditions where AI-based systems learn their own decision models from data and remain partly opaque (Burrell, 2016; Faraj et al., 2018; Zhang et al., 2021), delegation already begins during development. Developers themselves enter into a delegation relationship with the AI-based system by letting the system learn a decision logic they can neither fully specify nor fully inspect. Paper IV theorizes this as delegation-in-development, specifying how developers delegate to the AI-based system via appraisal-, distribution-, and coordination-in-development. Drawing on temporally embedded agency (Emirbayer & Mische, 1998), Paper IV demonstrates how the three temporal orientations (iterational, projective, and practical-evaluative) shape how developers delegate to the AI-based system and, in turn, how they construct the user-AI delegation in different ways. Paper IV shows that delegation in AI-based systems unfolds as two cycles of appraisal, distribution, and coordination that are linked, spanning development and use, rather than a process at use time alone. This triadic framing advances the delegation perspective in IS by explaining how the conditions for user-AI delegation are constructed through developer-AI delegation.

Second, Paper IV contributes to the literature on algorithmic learning in AI-based systems by demonstrating that the learning capacity of these systems is not merely technologically given, but is shaped by how developers transfer responsibility to the AI-based system and regulate this transfer. Prior work emphasizes that machine learning systems construct their own decision logic from data, resulting in partial opacity and unpredictability in organizational settings (e.g., Burrell, 2016; Faraj et al., 2018; Seidel et al., 2019; Zhang et al., 2021). Paper IV shows that how much responsibility developers are willing to hand over to the AI-based system, and thus how broadly and autonomously it is allowed to learn, depends on the developers' temporal orientation (Emirbayer & Mische, 1998). In an iterational orientation, developers delegate their responsibility only narrowly for decision logic construction, keep the system's learning

constrained, rely on interpretable models, and close user review to maintain control. In a projective orientation, they delegate more expansive responsibility for optimizing outcomes of the AI-based system, broaden its learning scope, and govern it primarily through outcome metrics and performance monitoring. In a practical-evaluative orientation, particularly with LLM-based generative AI models, developers accept that they cannot meaningfully shape the internal decision model and instead involve users in the situated evaluation and correction of individual outputs during the user-AI interaction, while retaining responsibility for designing interaction patterns and retrieval mechanisms. In this way, Paper IV shows that the self-learning characteristic of AI-based systems is not an inherent property of the technology but an outcome of developers' delegation to these systems. The extent of autonomous learning capacity of AI-based systems is co-constructed through developers' delegation and regulation practices.

Third, Paper IV advances research on ISD for AI-based systems by specifying how development work changes when system behavior cannot be fully specified and inspected. Traditional ISD emphasizes requirements engineering (Browne & Rogich, 2001; Byrd et al., 1992; Hickey & Davis, 2004) and the specification of rules or decision logic (Berente et al., 2021; Herbst, 1996; Ram & Khatri, 2005). In AI-based systems, by contrast, developers hand over parts of the decision logic construction to the AI-based system and then coordinate this delegation rather than solely encoding decision rules. ISD thus shifts from specifying system logic to designing human-AI delegation. Paper IV further extends work on human-in-the-loop design in AI-based systems development, which highlights the role of domain expertise (Gronsund & Aanestad, 2020; van den Broek et al., 2021). While existing literature generally recommends involving domain experts, Paper IV refines the notion of human-in-the-loop by showing that its extent and locus differ with developers' temporal orientations in delegation-in-development. The extent of human-in-the-loop ranges from high (iterational and practical-evaluative) to low (projective), and the focus shifts from internal logic alignment (iterational) to outcome monitoring (projective) to instance-level input-output evaluation (practical-evaluative). These contributions reshape ISD for AI-based systems from a process of specifying and implementing decision logic to a practice of designing and governing the delegation of responsibility across the triad of developers, users, and the AI-based system.

5.1.4 Implication of AI-augmented ISD for ISD Processes

The fourth research question (RQ4) asks: *How do software developers realize and shape generative AI-augmented ways of accomplishing their development tasks in organizations?* **Paper V** answers this question by theorizing two mechanisms, enacting and expanding, through which developers, via development practices, realize and shape generative AI-augmented ways of working in their ISD processes. In doing so, Paper V explains how generative AI augmentation emerges in ISD and becomes embedded in software developers' ISD processes. This extends the literature on AI-augmented ISD beyond examining which development tasks gen-

erative AI can support (Banh et al., 2025; Barke et al., 2023; Ozkaya, 2023; Santos et al., 2025) by specifying how developers realize and shape generative AI-augmented action possibilities.

A second contribution of Paper V is to show how generative AI-augmented software development reshapes ISD processes. Paper V demonstrates that generative AI differs from previous automation tools in ISD, which typically implement pre-defined rules and support well-defined software development tasks (Banker & Kauffman, 1991; Bessey et al., 2010; Sunitha & Samuel, 2019). With generative AI, developers face the tension between, on the one hand, a general-purpose technology that is versatile but not optimized for specific organizational contexts and tasks, and, on the other hand, the need to account for organizational and project-specific constraints in ISD in organizational settings. To make use of generative AI in their ISD processes while managing this tension, developers must engage in additional development practices, which realize and shape generative AI-augmented ISD, alongside classic ISD activities of specifying, designing, implementing, testing, and maintaining software systems (Meyer et al., 2017; Sommerville, 2015).

In the everyday work of developers, enacting mechanisms add additional practices of realizing generative AI-augmented software development in addition to classic ISD processes: developers interact with generative AI, in line with their individual goals, to structure and refine task instructions and validate generative AI outputs. Expanding mechanisms introduce further development practices on top of this: developers design contextualization that supplies generative AI with organization- and system-specific contexts and control the resulting contextualized generative AI-based artifacts to ensure quality and reliability. Crucially, these expanding mechanisms broaden the action possibilities of generative AI in software development, creating new possibilities that developers can subsequently enact in their everyday ISD practice.

Taken together, Paper V contributes to the ISD literature by theorizing how enacting and expanding mechanisms add new development practices for realizing and shaping generative AI-augmented ISD alongside classic ISD activities. Through this, Paper V shows how generative AI reshapes ISD processes and, in particular, how generative AI reshapes how software development tasks are accomplished in organizational settings.

5.1.5 Implication of AI-augmented ISD for Developers' Individual Work Outcomes and Job Crafting

The fifth research question (RQ5) asks: *How does developers' use of generative AI influence their individual-level work outcomes (work engagement, job satisfaction, and work meaning), and what is the mediating role of developers' job crafting in this relationship?* **Paper VI** addresses this question and contributes to research on AI-augmented ISD in several ways.

First, it extends the emerging IS and software engineering literature on generative AI in software development beyond its predominant focus on technical productivity factors. Prior studies

show that generative AI can increase output and reduce completion time on programming tasks (Cui et al., 2024; Peng et al., 2023; Ziegler et al., 2022, 2024), but provide little insight into how generative AI use impacts developers' individual-level work outcomes. By theorizing and empirically examining work engagement, job satisfaction, and work meaning as outcomes of generative AI use in an organizational ISD setting, Paper VI shows that AI augmentation in ISD is positively associated with developers' work engagement, but not with job satisfaction or work meaning. This refines the current understanding of productivity gains in AI-augmented ISD by demonstrating that generative AI use is positively related not only to technical performance metrics but also to work engagement as an individual-level work outcome that is an important antecedent of performance and developers' productivity in ISD (Christian et al., 2011; Murphy-Hill et al., 2019; Storey et al., 2019). Paper VI thus contributes to the literature on AI-augmented ISD and broadens our understanding of its impact on developers' productivity from a narrow focus on technical metrics to a more differentiated perspective that incorporates non-technical factors. It provides initial evidence that generative AI use in ISD is positively associated with work engagement as a key individual-level work outcome.

Second, Paper VI contributes to research on generative AI and the reshaping of software developers' jobs. Existing work argues that generative AI will substantially transform developers' roles and tasks (Sætra, 2023; Sauvola et al., 2024; Ulfsnes et al., 2024). Furthermore, emerging studies suggest that generative AI can influence employees' job crafting behavior (Freise et al., 2025; Mayer et al., 2025a; Ritz et al., 2025). Contrary to these expectations, Paper VI finds no evidence that developers' job crafting mediates the relationship between generative AI use and individual-level work outcomes. Instead, the results show that proactive personality is positively related to approach role and approach resource crafting, and that job autonomy is positively related to avoidance resource crafting. This finding is in line with job crafting research that identifies individual and job-level factors as central antecedents of job crafting behavior (Bruning & Campion, 2018; Rudolph et al., 2017; Tims & Bakker, 2010). The results further demonstrate that approach role crafting is positively related to work engagement and work meaning, whereas avoidance resource crafting has a negative relationship with work meaning. This also confirms findings from the job crafting literature (Lichtenthaler & Fischbach, 2019; Petrou et al., 2012; Zhang & Parker, 2019).

Together, these findings indicate that, in the studied setting, developers' job crafting is primarily shaped by individual and job-level characteristics rather than by the extent of generative AI use. In the large telecommunications organization examined, software developers work in an incumbent organizational environment, and, at the time of the study, had an average of about six months of experience using GitHub Copilot. In this context, job crafting is not identified as a mediator in the relationship between generative AI use and individual-level work outcomes, nor is generative AI use significantly related to any of the four job crafting dimensions. Paper VI thus advances prior work on generative AI-enabled job crafting by showing that in certain

real-world organizational contexts, individual and job-level characteristics remain the dominant antecedents of how developers shape their jobs.

Third, Paper VI contributes to the job crafting research and the perspective on job complexity in the context of generative AI by introducing the concept of job complexity of generative AI-augmented work and by examining its role in the relationship between generative AI use and job crafting. Drawing on the job crafting research that conceptualizes job complexity as a central job characteristic that can be associated with both approach and avoidance forms of job crafting (Bai et al., 2021; Tian et al., 2022), Paper VI shifts the focus from overall job complexity to the complexity of those parts of the job for which developers use generative AI. The results show no general relationship between generative AI use and any of the four job crafting dimensions. However, when the generative AI-augmented work is highly complex, generative AI use is positively associated with avoidance resource crafting, whereas no such association is found for low-complexity AI-augmented work. Paper VI thus provides initial evidence that job complexity of generative AI-augmented work is a useful concept for job crafting research in the context of generative AI. It suggests that the impact of generative AI use on job crafting may be contingent on the complexity of the parts of the job that are AI-augmented.

5.2 Practical Implications

This dissertation offers practical implications for organizations that seek to leverage AI in two ways. First, it addresses how to develop AI-based systems as outputs of information systems development (ISD) that are deployed in business processes and decision-making. Second, it examines how generative AI can be used to augment software developers' work within ISD processes.

These insights are relevant for leaders, developers, and user representatives who are responsible for designing and governing AI-based systems, integrating generative AI into development work, and shaping conditions under which humans and AI-based systems jointly accomplish organizational tasks.

In the following, the practical implications from Paper I to IV for developing AI-based systems are synthesized first. Section 5.2.1 outlines how organizations can orchestrate the development of AI-based systems within the triad of developers, users, and AI-based systems. Section 5.2.2 provides insights on how to design delegation in the development of AI-based systems.

Sections 5.2.3 and 5.2.4 then present the practical implications of Paper V and VI, which address the use of generative AI to augment developers' work in ISD. Section 5.2.3 synthesizes how organizations can shape AI-augmented ISD by enabling developers to engage in generative AI-augmented development practices in their everyday work. Section 5.2.4 provides insights into the impact of AI-augmented ISD in organizational settings and discusses how organizations can assess this impact.

5.2.1 Orchestrating AI-based Systems Development in the Triad

Across Papers I to IV, this dissertation conceptualizes AI-based systems development as taking place in a triad of interdependent agents: developers, users, and the AI-based system. Orchestrating work in this triad means coordinating development practices and artifacts across these agents and across development phases so that knowledge, insights, problem representation, and task context can be aligned. Specifically, this needs to explicitly account for the agentic characteristics of AI-based systems and the new challenges they create during development. **Paper I** raises practitioners' awareness of the knowledge-related boundaries that arise between developers, users, and the AI-based systems at each development phase. It identifies phase-specific practices and artifacts that function as boundary objects, such as data objects in model training, system artifacts like simulations and what-if analyses in algorithm implementation, and business rules and KPI artifacts in output interpretation and decision-making. Organizations developing AI-based systems can draw on these findings on boundaries, practices, and artifacts to systematically identify where knowledge-related boundaries emerge in the triad of developers, users, and the AI-based system, and to understand how to address them in each phase of development.

Paper II deepens this perspective by specifying how developers align AI-based systems with joint human-AI problem-solving tasks while, in parallel, shaping users' understanding of the system during development. It shows that such systems emerge from recurrent alignment practices enacted by developers. In these practices, developers iteratively adjust the AI-based system's learned problem representation to task demands, and support users in building mental representations of how the system functions and how to act with it. For practice, this implies that development work must cover both streams. Alongside model engineering and contextualization of the AI-based system, developers need to enact development practices that render the system's learned problem representation intelligible and actionable for users in accomplishing the joint human-AI problem-solving tasks. On this basis, users can apply their domain knowledge to assess whether the system's problem representations match task requirements and provide informed feedback, which developers again use to refine and contextualize the AI-based system.

For the development of generative AI-based systems, **Paper III** shows that when organizations build on pre-trained models without modifying the underlying model, development shifts from a model-centric approach to a development centered on the user-AI interaction. Developers shape the user-AI interaction through the practices of enablement and control so that the system outputs align with the organizational task context and requirements. Enablement and control practices are enacted both towards users and towards the generative AI-based system. For practitioners, this implies that the development of generative AI-based systems should focus on designing the user-AI interaction as the central development activity. Paper III further shows that the locus of evaluation expands to situated evaluation during user-AI interaction, where users continuously judge whether outputs fit task requirements. Organizations, therefore, need ways to capture

and analyze such situated user evaluations. In this way, Paper III shows that user-AI interaction and situated evaluation become key levers for aligning generative AI-based systems with organizational tasks.

5.2.2 Designing Delegation in AI-based Systems Development

Paper IV introduces the concept of delegation-in-development and shows that developers hand over parts of their traditional responsibility for constructing decision logic to the learning and partly opaque AI-based system. AI-based systems development can thus be understood as the design of delegation. How developers allocate responsibilities to the system and govern them – specifically, whether they adopt an iterative, projective, or practical-evaluative orientation – directly shapes how user-AI delegation can subsequently unfold in use. These design-time choices determine what the AI-based system is authorized to decide, what users can appraise and contest, and which coordination levers are available during use. Accordingly, a key implication for practitioners is that such choices should not remain within the purview of technical teams alone but be made jointly with key business stakeholders, including users and leaders, so that the organization can intentionally determine the extent and form of delegation to the AI-based system it is willing to operationalize and govern. Making the development orientation explicit, knowing how it configures the conditions of use, enables both *ex ante* alignment of the delegation design and, *ex post*, targeted adjustments when use reveals mismatches between the intended way of delegation and the actual conditions for user-AI delegation constructed during development.

Paper IV further qualifies the common assumption that domain experts must always be closely involved in building AI-based systems. It specifies when and where expert input has the greatest effect. Expertise is most valuable when aligning system behavior with established practices or when interpreting ambiguous outputs that depend on professional judgment. It is less central when outcomes can be clearly specified and measured. And it moves toward the interaction layer when the outputs of the AI-based system can be assessed and adjusted at the moment of user-AI interaction. Rather than involving experts by default, the findings suggest that organizations should be aware of when and how expert involvement is most needed.

Overall, Paper IV helps organizations recognize the development of the AI-based system as a setting in which delegation to the system is designed. Delegation choices during development should be made explicit as design decisions rather than left as implicit by-products of model training and system integration, as they construct the role of the AI-based system and condition what users can appraise, contest, and coordinate in use.

5.2.3 Shaping AI-augmented ISD

Paper V shows that generative AI-augmented ISD is not only a matter of giving developers access to new generative AI-based tools. Because general-purpose LLMs are only partially aligned with organization- and system-specific contexts, generative AI adds value in ISD only when developers shape how it is used through additional development practices. A first implication for organizations is therefore to recognize the enacting mechanism, instantiated through development practices of AI-augmented ISD, in everyday software development work as an integral part of developers' ISD processes. Developers need time and guidance to identify where generative AI fits their individual tasks and responsibilities, to learn how to structure and refine task instructions, and to validate outputs against coding standards and quality expectations. Organizations should support these development practices, for example, by providing training on task-specific instruction and validation, establishing guidelines for integrating generative AI use, and supporting team practices that treat the review and testing of generative AI outputs as standard parts of the development process.

A second implication concerns the expanding mechanism instantiated through development practices through which organizations can unlock generative AI support for context-heavy, high-value development tasks. Paper V shows that realizing generative AI support for tasks such as the explanation of custom code or legacy code translation requires additional contextualization and governance work. Organizations should therefore align expanding efforts with company-level priorities and identify where generative AI can create the greatest organizational value.

Finally, Paper V shows that expanding and enacting practices are intertwined, with consequences for how organizations should organize generative AI-augmented ISD. Contextualized generative AI-based artifacts created through expanding work only reshape ISD if they are taken up in developers' everyday work through enacting practices. Accordingly, the expanding efforts should not be driven solely by top-down company-level goals but should be co-shaped by the developers who will later use the resulting artifacts. This implies involving developers early when selecting use cases and designing contextualization, and feeding back developers' experiences from enactment, where generative AI helps, fails, or changes how development tasks are done, into subsequent expanding efforts. In doing so, organizations can unlock the potential of generative AI to reshape how software development work is accomplished: expanding practices broaden the action possibilities of generative AI, while enacting practices in everyday work determine which of these possibilities become value-adding ways for individual developers to accomplish software development tasks. In effect, organizations can support generative AI-augmented software development by deciding which parts of development work are augmented by generative AI and by enabling developers to embed this augmentation into their everyday development practices.

5.2.4 Assessing the Impact of AI-augmented ISD

Paper VI has practical implications for organizations that introduce generative AI into their ISD processes to support software developers in their work. The findings suggest that generative AI-augmented ISD should be evaluated not only through technical productivity indicators but also through developers' individual-level work outcomes. In the studied setting, a greater extent of generative AI use is associated with greater work engagement of developers (feeling more absorbed, vigorous, and dedicated in their work), but not with higher job satisfaction or perceived work meaning. This indicates that generative AI can support certain facets of developers' work experience while leaving others unchanged. To obtain a more comprehensive picture of the impact of generative AI-augmented ISD, organizations should therefore complement technical productivity metrics with regular monitoring of key individual-level work outcomes. This helps avoid situations in which technical indicators improve while individual-level outcomes stagnate or deteriorate, potentially undermining productivity gains.

Paper VI further indicates that, in the incumbent organization studied and at the current stage of generative AI adoption (on average, six months of GitHub Copilot use), developers' job crafting is driven primarily by individual and job-level characteristics rather than by the extent of generative AI use. This result stands in contrast to existing work that argues that generative AI is fundamentally reshaping developers' jobs and job crafting. For practice, this suggests that introducing generative AI into ISD processes alone may not achieve substantial changes in how developers proactively shape their jobs in real-world organizational settings. Organizations seeking such changes may therefore need to complement tool rollout with opportunities for developers to reflect, experiment, and exchange with peers, so they can identify where generative AI adds value to their tasks and under which conditions it can be integrated into their everyday work practices to support desirable work outcomes.

6 Limitations and Future Research

This dissertation has certain limitations, which at the same time offer opportunities for future research. The insights generated in this dissertation are shaped by specific empirical settings and methodological choices. These boundary conditions set the scope of the work and are reflected in the following overarching limitations, which in turn inform directions for future research.

6.1 Limitations

First, the findings in Paper I to V are based on a qualitative case study research design, drawing on in-depth studies in a large European telecommunications organization. This design enables rich insights into how development practices unfold in the triad of developers, users, and the AI-

based system, and how developers engage in AI-augmented ISD with generative AI. The cases are used to develop and refine concepts that can inform similar contexts (Eisenhardt, 1989b; Yin, 2018). In line with the discussion of Lee and Baskerville (2012) of generalizing to new settings, the applicability of these findings to other organizational settings cannot be presumed but needs to be examined and demonstrated in future empirical research. In particular, the studied organization has a broad AI project portfolio across business units and provides software engineers with extensive access to generative AI for augmented work, which may not be representative of smaller companies or public organizations.

Second, this dissertation focuses on AI-based systems and applications that support internal experts with domain expertise in accomplishing work within the organization, such as diagnostics, incident management, campaign design, and software development. This focus is well-suited to theorizing AI in ISD as an agentic IS artifact for joint human-AI problem-solving in internal work processes. The empirical studies thus examine AI-based systems used by internal experts rather than systems that interact directly with customers or operate in highly safety-critical contexts. The theoretical insights developed here are therefore grounded in this application context, and their applicability to other application domains or AI-based systems, such as customer-facing or safety-critical AI-based systems, needs to be examined in future research.

Third, this dissertation examines the development and early use of AI-based systems and the early adoption of generative AI in ISD. For AI-based systems, Paper I to IV investigate how development practices are enacted and how they shape early use, addressing a gap in IS research, which has largely analyzed users' adoption and use-time interaction while leaving the development work comparatively underexplored. For generative AI-augmented ISD, Paper V and VI investigate the early adoption of generative AI by developers, reflecting that the use of generative AI in development work is a recent phenomenon and can currently be examined primarily in its early adoption. As a result, the findings of this dissertation are most informative for the development and early adoption phases, while long-term dynamics, such as how development practices, delegation conditions, or generative AI-related job crafting and individual work outcomes stabilize or change over extended periods of use, remain outside the temporal scope of this dissertation.

Fourth, in Paper II to IV, which theorize development practices for AI-based systems, the analysis foregrounds the perspective of developers. This responds to the state of existing research, which has predominantly examined users' adoption and use of AI-based systems, while developers often remain in the background as technical implementers. This dissertation extends this literature by detailing how developers confront emerging challenges in development due to the agentic characteristics of AI-based systems, how they construct user-AI delegation, and how they enact alignment and coordination practices in the triad of developers, users, and the AI-based system. This focus means that users' perspectives on development-time decisions and conditions are incorporated in less depth. The findings of this dissertation thus offer a particu-

larly detailed account of how developers enact development practices in the triad. In doing so, these findings complement the IS literature that has predominantly focused on the use and user adoption of AI-based systems.

Finally, this dissertation examines the development of AI-based systems and AI-augmented ISD as two analytical perspectives. Paper I to IV theorize how developers enact development practices in the triad of developers, users, and the AI-based system. Papers V and VI investigate how developers use generative AI to augment their development work when developing traditional IS artifacts and how this relates to job crafting and individual work outcomes. With these analyses, this dissertation develops conceptual frameworks for the triadic development of AI-based systems on the one hand, and an understanding of the implications for generative AI-augmented ISD on the other hand. This dissertation does not examine settings in which these two perspectives are combined, that is, cases in which developers develop AI-based systems while using generative AI during development.

6.2 Future Research

The findings of this dissertation offer several promising avenues for future research. In the following, four research directions are outlined that aim to extend the theorizing on the development of AI-based systems and generative AI-augmented ISD. These directions can further deepen our understanding of how AI reshapes ISD.

Investigating triadic development of AI-based systems across organizational settings and application domains. Future research can build on the findings of this dissertation by examining how the relationships between developers, users, and AI-based systems unfold in other organizational and application contexts. While the studies in this dissertation are situated in a large telecommunications provider with a broad AI project portfolio, studies in smaller firms, public organizations, or start-ups could analyze how differences in resources, governance, and expertise shape the development practices in the triad. In addition, research on AI-based systems operating in other application domains, such as autonomous agents in physical environments, safety-critical contexts, or customer-facing services could show how increased risk, regulation, and proximity to customers alter the roles and interactions between developers, users, and AI-based systems. As the triadic understanding of developing AI-based systems is still nascent in IS research, such broader investigations across organizational settings and application domains are particularly promising for refining the development practices identified in this dissertation.

Investigating long-term trajectories of the development of AI-based systems and AI-augmented information systems development. Future research can build on this dissertation by studying how AI-based systems development and AI-augmented ISD evolve over time. For AI-based systems development, building on the findings of Paper I to IV, longitudinal studies could examine the AI-based systems from initial development through everyday use and

later redesigns. A central focus would be to examine how users' experiences in everyday use feed back into development. Such research could show how development practices, such as alignment practices or coordination practices, are reinforced, adapted, or reworked over time, thereby deepening the understanding of how triadic practices in AI-based systems evolve across development and use.

For AI-augmented ISD, building on the findings of Paper V and VI, longitudinal research could investigate how generative AI-augmented ways of working develop beyond the early adoption phase. Building on Paper V, longitudinal case studies could examine how enacting and expanding mechanisms develop across projects and releases. Such research could, for example, analyze which contextualized generative AI-based artifacts become stabilized, how expanding efforts evolve, and how enacting practices in everyday development work change as developers gain experience with generative AI-augmented ISD. Building on Paper VI, multi-wave panel surveys could examine whether the positive association between generative AI use and work engagement persists, strengthens, or weakens over time, and whether effects on job satisfaction and work meaning emerge as generative AI becomes more deeply embedded in developers' work. Such longitudinal designs would also allow researchers to investigate how different dimensions of job crafting evolve as mediators between generative AI use and individual-level work outcomes.

Investigating how triadic AI-based systems development relates to AI vendors and customers. Building on the triadic view advanced in this dissertation (developers, users, AI-based systems), future research can examine how AI-based systems development relates to two additional important stakeholder groups: AI vendors and customers. First, future research can investigate the role of AI vendors. AI vendors design and provide models and services that internal developers later configure and integrate (Saffarizadeh et al., 2024). Studying how internal developers and external vendors jointly shape AI-based systems and delegation conditions and guardrails could reveal how design decisions outside the focal organization influence development practices within the developer-user-AI triad, for example, in terms of accountability, controllability, and trust.

Second, future research can consider customers as additional stakeholders who are served through AI-based systems. Organizational users with domain expertise may interact with customers of the organization via AI-based systems (Hanelt et al., 2025). Examining how customers' experiences and concerns feed back into the developer-user-AI triad could yield insights into how development practices are adapted when AI-based systems are used in interactions with customers. Taken together, these extensions could build on the triadic view developed in this dissertation while showing how practices in the developer-user-AI triad are shaped when AI-based systems are co-shaped by external vendors and deployed in customer-facing contexts.

Investigating delegation in the development of AI-based systems within generative AI-augmented ISD settings. Building on the two perspectives developed in this dissertation, fu-

ture research can examine the development of AI-based systems in settings where developers are additionally supported by generative AI during development. In such settings, three related delegation cycles could be studied. In a first cycle, developers delegate development tasks to generative AI, for example generating code, suggesting data transformations, and exploring alternative designs. In a second cycle, in the delegation-in-development as theorized in Paper IV, developers delegate a part of their responsibility for constructing the decision model to the AI-based system by letting it learn from data and governing this learning through appraisal-, distribution-, and coordination-in-development. In a third cycle, users delegate tasks and responsibilities to the AI-based system in use under the delegation conditions that were constructed through delegation-in-development.

Studying projects in which all three cycles are present would allow future research to analyze how delegation from developers to generative AI in the first cycle influences how developers then delegate the decision model construction to the AI-based system in the second cycle, and how both together shape user-AI delegation in the third cycle. Such studies could build on the findings of this dissertation and reveal, for example, how the extent of delegation of development tasks to generative AI affects developers' temporal orientation (iterational, projective, practical-evaluative) in delegation-in-development, and how it conditions the user-AI delegation at use time.

7 Conclusion

This cumulative dissertation offers novel insights into ISD in the age of AI and how AI reshapes ISD. The findings advance our understanding of how the development of AI-based systems as agentic IS artifacts requires a triadic perspective that considers developers, users, and the AI-based system as distinct agents in the development process. Drawing on a knowledge boundary lens, the dissertation surfaces new boundaries that arise across the triad that must be managed by leveraging new development practices and boundary objects. The dissertation further applies the cognitive fit lens to examine how to develop AI-based systems that are effective in supporting joint human-AI problem-solving. It shows that it is no longer sufficient to align the system's problem representation with task demands, as is the case for traditional deterministic systems. Instead, in developing agentic and probabilistic AI-based systems, the effective performance of joint human-AI problem-solving tasks depends on iteratively aligning task demands, the AI-based system's problem representations, and users' internal representations relevant to the task solution during systems development. By conceptualizing these interdependent development practices, this dissertation adapts cognitive fit to the setting of developing agentic AI-based systems.

In addition, based on the triadic perspective, the dissertation investigates how developers manage the tension of developing agentic AI-based systems that can neither be fully specified ex

ante nor fully inspected ex post. In examining this tension, the dissertation adopts a delegation lens. It theorizes the development of AI-based systems as delegation-in-development from the developers' viewpoint, shows how different developers' temporal orientations shape delegation-in-development, and how this, in turn, shapes the user-AI delegation in use. In doing so, the dissertation extends IS delegation theory from a dyadic, use-time perspective to a triadic perspective of delegation that spans developers, users, and the AI-based systems across development and use.

Finally, the dissertation contributes to a deeper understanding of how the augmentation of developers' work by generative AI reshapes ISD. It shows that generative AI reshapes ISD processes through additional development practices that developers engage in to realize generative AI-augmented ways of working in their ISD processes and, in the organizational context, to create new possibilities of AI-augmented ISD. The dissertation further provides initial evidence that, beyond technical productivity effects, generative AI-augmented ISD is positively related to developers' work engagement as an individual-level work outcome. In this way, the dissertation enriches existing findings on AI-enabled productivity gains and advances a more comprehensive understanding of the impact of generative AI-augmented ISD on developers and their work outcomes. At the same time, contrary to arguments in existing research that generative AI transforms developers' jobs, this dissertation reveals that, in real-world organizational settings, individual and job-level characteristics remain the dominant antecedents of how developers shape their jobs and that generative AI use alone is not related to broader proactive changes to developers' jobs.

By providing a comprehensive account of AI as both an ISD output and an augmentation technology within ISD processes, this dissertation shows how AI reshapes ISD. This understanding is critical, as the ways organizations develop AI-based systems and embed AI into ISD work shape whether AI-based systems are aligned with human goals and needs and whether AI augmentation in ISD becomes a value-adding part of everyday development practice.

References

- Aaltonen, A., Alaimo, C., & Kallinikos, J. (2021). The making of data commodities: Data analytics as an embedded process. *Journal of Management Information Systems*, 38(2), 401–429.
- Abdel-Karim, B. M., Pfeuffer, N., Carl, K. V., & Hinz, O. (2023). How AI-Based Systems Can Induce Reflections: The Case of AI-Augmented Diagnostic Work. *MIS Quarterly*, 47(4), 1395–1424.
- Ågerfalk, P. J., Conboy, K., Crowston, K., Lundström, J. E., Jarvenpaa, S. L., Ram, S., & Mikalef, P. (2022). Artificial Intelligence in Information Systems: State of the Art and Research Roadmap. *Communications of the AIS*, 50, 420–438.
- Ågerfalk, P. J., & Fitzgerald, B. (2006). Session details: Flexible and distributed software processes: Old petunias in new bowls? *Communications of the ACM*, 49(10), 27–34.
- Ahmed, T., & Devanbu, P. (2022). Few-shot training llms for project-specific code summarization. *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, 1–5.
- Alaimo, C., & Kallinikos, J. (2021). Managing by data: Algorithmic categories and organizing. *Organization Studies*, 42(9), 1385–1407.
- Alavi, M., Leidner, D. E., & Mousavi, R. (2024). A Knowledge Management Perspective of Generative Artificial Intelligence. *Journal of the Association for Information Systems*, 25(1), 1–12.
- Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmermann, T. (2019a). Software engineering for machine learning: A case study. *IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 291–300.
- Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*.
- Ang, S., & Slaughter, S. A. (2001). Work Outcomes and Job Design for Contract Versus Permanent Information Systems Professionals On Software Development Teams^{1, 2}. *MIS Quarterly*, 25(3), 321–350.
- Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, 82–115.
- Asatiani, A., Malo, P., Nagbol, P. R., Penttinen, E., Rinta-Kahila, T., & Salovaara, A. (2021). Sociotechnical Envelopment of Artificial Intelligence: An Approach to Organizational Deployment of Inscrutable Artificial Intelligence Systems. *Journal of the Association for Information Systems*, 22(2), 325–352.

- Auernhammer, J. (2020). Human-centered AI: The role of Human-centered Design Research in the development of AI. *Synergy - DRS International Conference 2020*, 1315–1333.
- Bai, J. Y., Tian, Q., & Liu, X. (2021). Examining job complexity on job crafting within conservation of resources theory: A dual-path mediation model. *Frontiers in Psychology*, 12, 1–12.
- Baird, A., & Maruping, L. M. (2021). The Next Generation of Research on IS Use: A Theoretical Framework of Delegation to and from Agentic IS Artifacts. *MIS Quarterly*, 45(1), 315–341.
- Bakker, A. B., Tims, M., & Derks, D. (2012). Proactive personality and job performance: The role of job crafting and work engagement. *Human Relations*, 65(10), 1359–1378.
- Balijepally, V., Mahapatra, R., Nerur, S., & Price, K. H. (2009). Are two heads better than one for software development? The productivity paradox of pair programming. *MIS Quarterly*, 33(1), 91–118.
- Banh, L., Holldack, F., & Strobel, G. (2025). Copiloting the future: How generative AI transforms Software Engineering. *Information and Software Technology*, 183, 1–14.
- Banker, R. D., Datar, S. M., Kemerer, C. F., & Zweig, D. (1993). Software complexity and maintenance costs. *Communications of the ACM*, 36(11), 81–94.
- Banker, R. D., & Kauffman, R. J. (1991). Reuse and productivity in integrated computer-aided software engineering: An empirical study. *MIS Quarterly*, 15(3), 375–401.
- Banker, R. D., & Slaughter, S. A. (2000). The Moderating Effects of Structure on Volatility and Complexity in Software Enhancement. *Information Systems Research*, 11(3), 219–240.
- Barke, S., James, M. B., & Polikarpova, N. (2023). Grounded copilot: How programmers interact with code-generating models. *Proceedings of the ACM on Programming Languages*, 7, 85–111.
- Barki, H., & Hartwick, J. (1989). Rethinking the concept of user involvement. *MIS Quarterly*, 13(1), 53–63.
- Barki, H., & Hartwick, J. (1994). Measuring user participation, user involvement, and user attitude. *MIS Quarterly*, 18(1), 59–82.
- Baskerville, R. L., & Stage, J. (1996). Controlling Prototype Development through Risk Analysis. *MIS Quarterly*, 20(4), 481–504.
- Baylor, D., Breck, E., Cheng, H.-T., Fiedel, N., Foo, C. Y., Haque, Z., Haykal, S., Ispir, M., Jain, V., & Koc, L. (2017). Tfx: A tensorflow-based production-scale machine learning platform. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1387–1395.
- Beaudry, A., & Pinsonneault, A. (2005). Understanding user responses to information technology: A coping model of user adaptation. *MIS Quarterly*, 29(3), 493–524.
- Benbasat, I., Goldstein, D. K., & Mead, M. (1987). The case research strategy in studies of information systems. *MIS Quarterly*, 11(3), 369–386.

- Benbasat, I., & Zmud, R. W. (2003). The identity crisis within the IS discipline: Defining and communicating the discipline's core properties. *MIS Quarterly*, 27(2), 183–194.
- Benbya, H., Davenport, T. H., & Pachidi, S. (2020). Artificial intelligence in organizations: current state and future opportunities. *MIS Quarterly Executive*, 19(4), 1–15.
- Benbya, H., Pachidi, S., & Jarvenpaa, S. (2021). Special issue editorial: artificial intelligence in organizations: implications for information systems research. *Journal of the Association for Information Systems*, 22(2), 281–303.
- Benbya, H., Strich, F., & Tamm, T. (2024). Navigating Generative Artificial Intelligence Promises and Perils for Knowledge and Creative Work. *Journal of the Association for Information Systems*, 25(1), 23–36.
- Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing Artificial Intelligence. *MIS Quarterly*, 45(3), 1433–1450.
- Berg, J. M., Dutton, J. E., & Wrzesniewski, A. (2013). Job crafting and meaningful work. American Psychological Association.
- Berg, J. M., Raj, M., & Seamans, R. (2023). Capturing value from artificial intelligence. *Academy of Management Discoveries*, 9(4), 424–428.
- Bessey, A., Block, K., Chelf, B., Chou, A., Fulton, B., Hallem, S., Henri-Gros, C., Kamsky, A., McPeak, S., & Engler, D. (2010). A few billion lines of code later: using static analysis to find bugs in the real world. *Communications of the ACM*, 53(2), 66–75.
- Bird, C., Ford, D., Zimmermann, T., Forsgren, N., Kalliamvakou, E., Lowdermilk, T., & Gazit, I. (2023). Taking flight with Copilot. *Communications of the ACM*, 66(6), 56–62.
- Bommasani, R. (2021). On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.
- Bostrom, R. P., & Heinen, J. S. (1977a). MIS Problems and Failures: A Socio-Technical Perspective, Part II: The Application of Socio-Technical Theory. *MIS Quarterly*, 1(4), 11–28.
- Bostrom, R. P., & Heinen, J. S. (1977b). MIS problems and failures: A socio-technical perspective. Part I: The causes. *MIS Quarterly*, 1(3), 17–32.
- Boussioux, L., Lane, J. N., Zhang, M., Jacimovic, V., & Lakhani, K. R. (2024). The crowdless future? Generative AI and creative problem-solving. *Organization Science*, 35(5), 1589–1607.
- Bowling, N. A., & Hammond, G. D. (2008). A meta-analytic examination of the construct validity of the Michigan Organizational Assessment Questionnaire Job Satisfaction Subscale. *Journal of Vocational Behavior*, 73(1), 63–77.
- Breck, E., Cai, S., Nielsen, E., Salib, M., & Sculley, D. (2017). The ML test score: A rubric for ML production readiness and technical debt reduction. *2017 IEEE International Conference on Big Data (Big Data)*, 1123–1132.

- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., & Askell, A. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
- Browne, G. J., & Rogich, M. B. (2001). An empirical investigation of user requirements elicitation: Comparing the effectiveness of prompting techniques. *Journal of Management Information Systems*, 17(4), 223–249.
- Bruning, P. F., & Campion, M. A. (2018). A role–resource approach–avoidance model of job crafting: A multimethod integration and extension of job crafting theory. *Academy of Management Journal*, 61(2), 499–522.
- Brynjolfsson, E., Li, D., & Raymond, L. R. (2023). Generative AI at work.
- Bürkland, S., & Zachariassen, F. (2014). Developing an ERP technology: Handling incompleteness of the system. *Scandinavian Journal of Management*, 30(4), 409–426.
- Burrell, J. (2016). How the Machine ‘Thinks’: Understanding Opacity in Machine Learning Algorithms. *Big Data & Society*, 3(1), 1–12.
- Byrd, T. A., Cossick, K. L., & Zmud, R. W. (1992). A Synthesis of Research on Requirements Analysis and Knowledge Acquisition Techniques. *MIS Quarterly*, 16(1), 117–138.
- Campion, M. A., & McClelland, C. L. (1993). Follow-up and extension of the interdisciplinary costs and benefits of enlarged jobs. *Journal of Applied Psychology*, 78(3), 339–351.
- Carlile, P. R. (2004). Transferring, translating, and transforming: An integrative framework for managing knowledge across boundaries. *Organization Science*, 15(5), 555–568.
- Castelfranchi, C., & Falcone, R. (1998). Towards a theory of delegation for agent-based systems. *Robotics and Autonomous Systems*, 24(3-4), 141–157.
- Chakraborty, S., Sarker, S., & Sarker, S. (2010). An Exploration into the Process of Requirements Elicitation: A Grounded Approach. *Journal of the Association for Information Systems*, 11(4), 212–249.
- Chalmers, R. (2006). Methodology for customer relationship management. *Journal of Systems and Software*, 79(7), 1015–1024.
- Chen, W., & Hirschheim, R. (2004). A paradigmatic and methodological examination of information systems research from 1991 to 2001. *Information Systems Journal*, 14(3), 197–235.
- Chin, W. W. (1998). Commentary: Issues and Opinion on Structural Equation Modeling. *MIS Quarterly*, 22(1), vii–xvi.
- Chollet, F. (2021). *Deep learning with Python*. Simon; Schuster.
- Christian, M. S., Garza, A. S., & Slaughter, J. E. (2011). Work engagement: A quantitative review and test of its relations with task and contextual performance. *Personnel Psychology*, 64(1), 89–136.
- Chuang, Y.-T., Chiang, H.-L., & Lin, A.-P. (2025). Insights from the Job Demands–Resources Model: AI’s dual impact on employees’ work and life well-being. *International Journal of Information Management*, 83, 1–17.

- Chui, M., Hazan, E., Roberts, R., Singla, A., Smaje, K., Sukharevsky, A., Yee, L., & Zemmell, R. (2023a). The economic potential of generative AI: The next productivity frontier. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier>
- Conboy, K. (2009). Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development. *Information Systems Research*, 20(3), 329–354.
- Cousot, P., Cousot, R., Feret, J., Mauborgne, L., Miné, A., Monniaux, D., & Rival, X. (2005). The ASTREE analyzer. *Proceedings of the 14th European Conference on Programming Languages and Systems*, 21–30.
- Cui, Z. K., Demirer, M., Jaffe, S., Musolff, L., Peng, S., & Salz, T. (2024). The effects of generative AI on high skilled work: Evidence from three field experiments with software developers. *Available at SSRN 4945566*.
- Dellermann, D., Ebel, P., Söllner, M., & Leimeister, J. M. (2019). Hybrid intelligence. *Business & Information Systems Engineering*, 61(5), 637–643.
- Demerouti, E. (2014). Design your own job through job crafting. *European Psychologist*.
- Deng, Y., Zheng, J., Huang, L., & Kannan, K. (2023). Let artificial intelligence be your shelf watchdog: The impact of intelligent image processing-powered shelf monitoring on product sales. *MIS Quarterly*, 47(3), 1045–1072.
- Dhar, V. (2013). Data science and prediction. *Communications of the ACM*, 56(12), 64–73.
- Dubé, L., & Paré, G. (2003). Rigor in information systems positivist case research: current practices, trends, and recommendations. *MIS Quarterly*, 27(4), 597–636.
- Edmondson, A. C., & McManus, S. E. (2007). Methodological fit in management field research. *Academy of Management Review*, 32(4), 1246–1264.
- Eisenhardt, K. M. (1989a). Agency theory: An assessment and review. *Academy of Management Review*, 14(1), 57–74.
- Eisenhardt, K. M. (1989b). Building theories from case study research. *Academy of Management Review*, 14(4), 532–550.
- Eisenhardt, K. M., & Graebner, M. E. (2007). Theory building from cases: Opportunities and challenges. *Academy of Management Journal*, 50(1), 25–32.
- Eloundou, T., Manning, S., Mishkin, P., & Rock, D. (2023). Gpts are gpts: An early look at the labor market impact potential of large language models. *arXiv preprint arXiv:2303.10130*.
- Emirbayer, M., & Mische, A. (1998). What is agency? *American Journal of Sociology*, 103(4), 962–1023.
- Fadel, K. J., & Brown, S. A. (2010). Information systems appraisal and coping: The role of user perceptions. *Communications of the AIS*, 26(1), 6.
- Faraj, S., Pachidi, S., & Sayegh, K. (2018). Working and organizing in the age of the learning algorithm. *Information and Organization*, 28(1), 62–70.
- Faraj, S., & Sproull, L. (2000). Coordinating expertise in software development teams. *Management Science*, 46(12), 1554–1568.

- Feuerriegel, S., Hartmann, J., Janiesch, C., & Zschech, P. (2024). Generative AI. *Business & Information Systems Engineering*, 66(1), 111–126.
- Fitzgerald, B. (1997). The use of systems development methodologies in practice: a field study. *Information Systems Journal*, 7(3), 201–212.
- Fornell, C., & Larcker, D. F. (1981). Evaluating structural equation models with unobservable variables and measurement error. *Journal of Marketing Research*, 18(1), 39–50.
- Foster, D. (2022). *Generative Deep Learning*. O'Reilly Media, Inc.
- Freise, L. R., Bruhin, O., Ritz, E., Li, M. M., & Leimeister, J. M. (2025). Code and Craft: How Generative AI Tools Facilitate Job Crafting in Software Development. *Hawaii International Conference on System Sciences (HICSS) Proceedings*.
- Fügener, A., Grahl, J., Gupta, A., & Ketter, W. (2022). Cognitive challenges in human–artificial intelligence collaboration: Investigating the path toward productive delegation. *Information Systems Research*, 33(2), 678–696.
- Gallivan, M. J., & Keil, M. (2003). The user–developer communication process: a critical case study. *Information Systems Journal*, 13(1), 37–68.
- Gerlich, M. (2025). AI tools in society: Impacts on cognitive offloading and the future of critical thinking. *Societies*, 15(6), 1–28.
- Glaser, V. L., Pollock, N., & D'Adderio, L. (2021). The biography of an algorithm: Performing algorithmic technologies in organizations. *Organization Theory*, 2(2), 1–27.
- Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1). MIT press Cambridge.
- Graves, T. L., Karr, A. F., Marron, J. S., & Siy, H. (2002). Predicting fault incidence using software change history. *IEEE Transactions on Software Engineering*, 26(7), 653–661.
- Gronlund, T., & Aanestad, M. (2020). Augmenting the algorithm: Emerging human-in-the-loop work configurations. *Journal of Strategic Information Systems*, 29(2), 1–16.
- Gulati, R., Wohlgezogen, F., & Zhelyazkov, P. (2012). The two facets of collaboration: Cooperation and coordination in strategic alliances. *Academy of Management Annals*, 6(1), 531–583.
- Hackman, J. R., & Oldham, G. R. (1976). Motivation through the design of work: Test of a theory. *Organizational Behavior and Human Performance*, 16(2), 250–279.
- Hadfield-Menell, D., & Hadfield, G. K. (2019). Incomplete contracting and AI alignment. *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 417–422.
- Hahn, J., & Lee, G. (2021). The Complex Effects of Cross-Domain Knowledge on IS Development: A Simulation-Based Theory Development¹. *MIS Quarterly*, 45(4), 2023–2054.
- Hair, J. F. (2014). *A primer on partial least squares structural equation modeling (PLS-SEM)*. Sage.
- Hair, J. F., Hult, G. T. M., Ringle, C. M., Sarstedt, M., Danks, N. P., & Ray, S. (2021). *Partial least squares structural equation modeling (PLS-SEM) using R: A workbook*. Springer Nature.

- Hair, J. F., Matthews, L. M., Matthews, R. L., & Sarstedt, M. (2017). PLS-SEM or CB-SEM: updated guidelines on which method to use. *International Journal of Multivariate Data Analysis*, 1(2), 107–123.
- Hair, J. F., Risher, J. J., Sarstedt, M., & Ringle, C. M. (2019). When to use and how to report the results of PLS-SEM. *European Business Review*, 31(1), 2–24.
- Halloran, D., Manchester, S., Moriarty, J., Riley, R., Rohrman, J., & Skramstad, T. (1978). Systems Development Quality Control. *MIS Quarterly*, 2(4), 1–13.
- Hanelt, A., Wiesche, M., Benlian, A., & Hinz, O. (2025). Opening The Network of Trust: How Domain Experts in Triadic Relationships Build Trust in Ai-Based Counterparts. *MIS Quarterly*, 32(1), 1–32.
- Harter, D. E., Krishnan, M. S., & Slaughter, S. A. (2000). Effects of process maturity on quality, cycle time, and effort in software product development. *Management Science*, 46(4), 451–466.
- He, J., & King, W. R. (2008). The role of user participation in information systems development: Implications from a meta-analysis. *Journal of Management Information Systems*, 25(1), 301–331.
- Henseler, J., Ringle, C. M., & Sarstedt, M. (2015). A new criterion for assessing discriminant validity in variance-based structural equation modeling. *Journal of the Academy of Marketing Science*, 43(1), 115–135.
- Herbst, H. (1996). Business rules in systems analysis: a meta-model and repository system. *Information Systems*, 21(2), 147–166.
- Hickey, A. M., & Davis, A. M. (2004). A unified model of requirements elicitation. *Journal of Management Information Systems*, 20(4), 65–84.
- Hirschheim, R., Klein, H. K., & Newman, M. (1991). Information systems development as social action: Theoretical perspective and practice. *Omega*, 19(6), 587–608.
- Hsu, J. S.-C., Lin, T.-C., Zheng, G.-T., & Hung, Y.-W. (2012). Users as knowledge co-producers in the information system development project. *International Journal of Project Management*, 30(1), 27–36.
- Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., & Qin, B. (2025). A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2), 1–55.
- Hunton, J. E., & Beeler, J. D. (1997). Effects of user participation in systems development: a longitudinal field experiment. *MIS Quarterly*, 21(4), 359–388.
- Iivari, J., & Maansaari, J. (1998). The usage of systems development methods: are we stuck to old practices? *Information and Software Technology*, 40(9), 501–510.
- Jain, H., Padmanabhan, B., Pavlou, P. A., & Raghu, T. S. (2021). Editorial for the special section on humans, algorithms, and augmented intelligence: The future of work, organizations, and society. *Information Systems Research*, 32(3), 675–687.

- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An introduction to statistical learning - second edition*. Springer.
- Jenkin, T. A., Chan, Y. E., & Sabherwal, R. (2019). Changes Within and Across Projects. *MIS Quarterly*, 43(2), 649–A17.
- Judge, T. A., Thoresen, C. J., Bono, J. E., & Patton, G. K. (2001). The job satisfaction–job performance relationship: A qualitative and quantitative review. *Psychological Bulletin*, 127(3), 376.
- Jussupow, E., Spohrer, K., Heinzl, A., & Gawlitza, J. (2021). Augmenting medical diagnosis decisions? An investigation into physicians’ decision-making process with artificial intelligence. *Information Systems Research*, 32(3), 713–735.
- Kane, G. C., Young, A. G., Majchrzak, A., & Ransbotham, S. (2021). Avoiding an oppressive future of machine learning: A design theory for emancipatory assistants. *MIS Quarterly*, 45(1), 371–396.
- Kaplan, B., & Duchon, D. (1988). Combining qualitative and quantitative methods in information systems research: a case study. *MIS Quarterly*, 12(4), 571–586.
- Kemp, A. (2024). Competitive advantage through artificial intelligence: Toward a theory of situated AI. *Academy of Management Review*, 49(3), 618–635.
- Kim, M., Zimmermann, T., DeLine, R., & Begel, A. (2016). The emerging role of data scientists on software development teams. *Proceedings of the 38th International Conference on Software Engineering*, 96–107.
- Kirsch, L. J. (1996). The management of complex tasks in organizations: Controlling the systems development process. *Organization Science*, 7(1), 1–21.
- Krakowski, S. (2025). Human-AI agency in the age of generative AI. *Information and Organization*, 35(1–25), 100560.
- Kreuzberger, D., Kühl, N., & Hirschl, S. (2023). Machine learning operations (mlops): Overview, definition, and architecture. *IEEE Access*, 11, 31866–31879.
- Krishnan, M. S., Mukhopadhyay, T., & Zubrow, D. (1999). Software process models and project performance. *Information Systems Frontiers*, 1(3), 267–277.
- Kudaravalli, S., Faraj, S., & Johnson, S. L. (2017). A Configural Approach to Coordinating Expertise in Software Development Teams. *MIS Quarterly*, 41(1), 43–64.
- Laato, S., Mäntymäki, M., Minkinen, M., Birkstedt, T., Islam, A. K. M., & Dennehy, D. (2022). Integrating machine learning with software development lifecycles: Insights from experts. *European Conference on Information Systems (ECIS) Proceedings*.
- Law, M., & Varanasi, R. A. (2025). Generative AI and Changing Work: Systematic Review of Practitioner-Led Work Transformations Through the Lens of Job Crafting. *International Conference on Human-Computer Interaction*, 131–152.
- Lawler, E., Cammann, C., Nadler, D., & Jenkins, D. (1979). Michigan organizational assessment questionnaire. *Journal of Vocational Behavior*.

- Lawrence, N. D. (2017). Living together: Mind and machine intelligence. *arXiv preprint arXiv: 1705.07996*.
- Lebovitz, S., Levina, N., & Lifshitz-Assaf, H. (2021). Is AI Ground Truth Really ‘True’? The Dangers of Training and Evaluating AI Tools Based on Experts’ Know-What. *MIS Quarterly*, 45(3), 1501–1525.
- Lebovitz, S., Lifshitz-Assaf, H., & Levina, N. (2022). To engage or not to engage with AI for critical judgments: How professionals deal with opacity when using AI for medical diagnosis. *Organization Science*, 33(1), 126–148.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.
- Lee, A. S. (1989). A scientific methodology for MIS case studies. *MIS Quarterly*, 13(1), 33–50.
- Lee, A. S., & Baskerville, R. L. (2003). Generalizing generalizability in information systems research. *Information Systems Research*, 14(3), 221–243.
- Lee, A. S., & Baskerville, R. L. (2012). Conceptualizing Generalizability: New Contributions and a Reply. *MIS Quarterly*, 36(3), 749–761.
- Lee, G., & Xia, W. (2010). Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. *MIS Quarterly*, 34(1), 87–114.
- Levina, N., & Vaast, E. (2005). The Emergence of Boundary Spanning Competence in Practice: Implications for Implementation and Use of Information Systems. *MIS Quarterly*, 29(2), 335–363.
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., & Rocktäschel, T. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459–9474.
- Li, M., Peters, C., Poser, M., Eilers, K., & Elshan, E. (2022). ICT-enabled job crafting: How Business Unit Developers use Low-code Development Platforms to craft jobs. *International Conference on Information Systems (ICIS) Proceedings*.
- Lichtenthaler, P. W., & Fischbach, A. (2019). A meta-analysis on promotion-and prevention-focused job crafting. *European Journal of Work and Organizational Psychology*, 28(1), 30–50.
- Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9), 1–35.
- Lou, B., & Wu, L. (2021). AI on Drugs: Can Artificial Intelligence Accelerate Drug Development? Evidence from a large-scale Examination of Bio-Pharma Firms. *MIS Quarterly*, 45(3), 1451–1482.
- Lyytinen, K., Nickerson, J. V., & King, J. L. (2021). Metahuman systems= humans+ machines that learn. *Journal of Information Technology*, 36(4), 427–445.
- MacCormack, A., & Sturtevant, D. J. (2016). Technical debt and system architecture: The impact of coupling on defect-related activity. *Journal of Systems and Software*, 120, 170–182.

- Mahmood, M. A. (1987). System Development Methods-A Comparative Investigation. *MIS Quarterly*, 11(3), 293–311.
- Majchrzak, A., More, P. H. B., & Faraj, S. (2012). Transcending knowledge differences in cross-functional teams. *Organization Science*, 23(4), 951–970.
- Malone, T. W., & Crowston, K. (1994). The interdisciplinary study of coordination. *ACM Computing Surveys (CSUR)*, 26(1), 87–119.
- Matook, S., Lee, G., & Fitzgerald, B. (2021). MISQ research curation on information systems development. *MIS Quarterly*, 1–20.
- Mayer, A.-S., Baygi, R. M., & Buwalda, R. (2025a). Generation AI: Job Crafting by Entry-Level Professionals in the Age of Generative AI: A.-S. Mayer et al. *Business & Information Systems Engineering*, 1–19.
- Meyer, A. N., Barton, L. E., Murphy, G. C., Zimmermann, T., & Fritz, T. (2017). The work life of developers: Activities, switches and perceived productivity. *IEEE Transactions on Software Engineering*, 43(12), 1178–1193.
- Mikalef, P., & Gupta, M. (2021). Artificial intelligence capability: Conceptualization, measurement calibration, and empirical study on its impact on organizational creativity and firm performance. *Information & Management*, 58(3), 1–20.
- Miles, M. B., & Huberman, A. M. (1994). *Qualitative Data Analysis: An Expanded Sourcebook*. Sage.
- Minelli, R., Mocci, A., & Lanza, M. (2015). I know what you did last summer-an investigation of how developers spend their time. *2015 IEEE 23rd international conference on program comprehension*, 25–35.
- Mingers, J. (2001). Combining IS research methods: towards a pluralist methodology. *Information Systems Research*, 12(3), 240–259.
- Monaghan, B. D., & Bass, J. M. (2020). Redefining legacy: a technical debt perspective. *International Conference on Product-Focused Software Process Improvement*, 254–269.
- Muldoon, J., Cant, C., Wu, B., & Graham, M. (2024). A typology of artificial intelligence data work. *Big Data & Society*, 11(1), 1–13.
- Murphy-Hill, E., Jaspan, C., Sadowski, C., Shepherd, D., Phillips, M., Winter, C., Knight, A., Smith, E., & Jorde, M. (2019). What predicts software developers' productivity? *IEEE Transactions on Software Engineering*, 47(3), 582–594.
- Murray, A., Rhymer, J., & Sirmon, D. G. (2021). Humans and technology: Forms of conjoined agency in organizations. *Academy of Management Review*, 46(3), 552–571.
- Newman, M., & Robey, D. (1992). A social process model of user-analyst relationships. *MIS Quarterly*, 16(2), 249–266.
- Nguyen-Duc, A., Cabrero-Daniel, B., Przybylek, A., Arora, C., Khanna, D., Herda, T., Rafiq, U., Melegati, J., Guerra, E., & Kemell, K.-K. (2023). Generative Artificial Intelligence for Software Engineering—A Research Agenda. *arXiv preprint arXiv:2310.18648*.
- Nicolini, D. (2012). *Practice theory, work, and organization: An introduction*. OUP Oxford.

- Nitzl, C., Roldan, J. L., & Cepeda, G. (2016). Mediation analysis in partial least squares path modeling: Helping researchers discuss more sophisticated models. *Industrial Management & Data Systems*, 116(9), 1849–1864.
- Okhuysen, G. A., & Bechky, B. A. (2009). 10 coordination in organizations: An integrative perspective. *Academy of Management Annals*, 3(1), 463–502.
- Ozkaya, I. (2023). Application of large language models to software engineering tasks: Opportunities, risks, and implications. *IEEE Software*, 40(3), 4–8.
- Paleyes, A., Urma, R.-G., & Lawrence, N. D. (2022). Challenges in deploying machine learning: a survey of case studies. *ACM Computing Surveys*, 55(6), 1–29.
- Patnayakuni, R., Rai, A., & Tiwana, A. (2007). Systems development process improvement: A knowledge integration perspective. 54(2), 286–300.
- Pawlowski, S. D., & Robey, D. (2004). Bridging user organizations: Knowledge brokering and the work of information technology professionals. *MIS Quarterly*, 28(4), 645–672.
- Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of AI on developer productivity: Evidence from GitHub Copilot. *arXiv preprint arXiv:2302.06590*.
- Petrou, P., Demerouti, E., Peeters, M. C. W., Schaufeli, W. B., & Hetland, J. (2012). Crafting a job on a daily basis: Contextual correlates and the link to work engagement. *Journal of Organizational Behavior*, 33(8), 1120–1141.
- Pfeuffer, N., Baum, L., Stammer, W., Abdel-Karim, B. M., Schramowski, P., Bucher, A. M., Hügel, C., Rohde, G., Kersting, K., & Hinz, O. (2023). Explanatory interactive machine learning: establishing an action design research process for machine learning projects. *Business & Information Systems Engineering*, 65(6), 677–701.
- Pinski, M., Adam, M., & Benlian, A. (2023). AI knowledge: Improving AI delegation through human enablement. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 1–17.
- Pinsonneault, A., & Kraemer, K. (1993). Survey research methodology in management information systems: an assessment. *Journal of Management Information Systems*, 10(2), 75–105.
- Polyzotis, N., Roy, S., Whang, S. E., & Zinkevich, M. (2018). Data lifecycle challenges in production machine learning: a survey. *ACM Sigmod Record*, 47(2), 17–28.
- Preacher, K. J., & Hayes, A. F. (2008). Asymptotic and resampling strategies for assessing and comparing indirect effects in multiple mediator models. *Behavior Research Methods*, 40(3), 879–891.
- Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*. San Francisco, CA, USA.
- Raisch, S., & Krakowski, S. (2021). Artificial Intelligence and Management: The Automation-Augmentation Paradox. *Academy of Management Review*, 46(1), 192–210.
- Raisch, S., & Fomina, K. (2025). Combining human and artificial intelligence: Hybrid problem-solving in organizations. *Academy of Management Review*, 50(2), 441–464.

- Ram, S., & Khatri, V. (2005). A comprehensive framework for modeling set-based business rules during conceptual database design. *Information Systems*, 30(2), 89–118.
- Ravichandran, T., & Rai, A. (1999). Total quality management in information systems development: key constructs and relationships. *Journal of Management Information Systems*, 16(3), 119–155.
- Ravichandran, T., & Rai, A. (2000). Quality management in systems development: an organizational system perspective. *MIS Quarterly*, 24(3), 381–415.
- Recker, J. (2021). *Scientific research in information systems: a beginner's guide*. Springer Nature.
- Ritz, E., Freise, L. R., & Li, M. M. (2025). Offloading to Digital Minds: How Generative AI Can Help to Craft Jobs. *Information Systems and Neuroscience: NeuroIS Retreat 2024, Vienna, Austria*, 66, 28–37.
- Rosenkranz, C., Stray, V., & Wiesche, M. (2025). Navigating the new frontier of information systems engineering. *Business & Information Systems Engineering*, 67(1), 1–5.
- Rosenkranz, C., Vranešić, H., & Holten, R. (2014). Boundary interactions and motors of change in requirements elicitation: A dynamic perspective on knowledge sharing. *Journal of the Association for Information Systems*, 15(6), 306–345.
- Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Sauvestre, R., & Remez, T. (2023). Code Llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.
- Rudolph, C. W., Katz, I. M., Lavigne, K. N., & Zacher, H. (2017). Job crafting: A meta-analysis of relationships with individual differences, job characteristics, and work outcomes. *Journal of Vocational Behavior*, 102, 112–138.
- Rus, I., Lindvall, M., & Sinha, S. (2002). Knowledge management in software engineering. *IEEE Software*, 19(3), 26–38.
- Russell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach*. Pearson.
- Sætra, H. S. (2023). Generative AI: Here to stay, but for good? *Technology in Society*, 75, 1–5.
- Saffarizadeh, K., Keil, M., & Maruping, L. (2024). Relationship Between Trust in the Artificial Intelligence Creator and Trust in Artificial Intelligence Systems: The Crucial Role of Artificial Intelligence Alignment and Steerability. *Journal of Management Information Systems*, 41(3), 645–681.
- Saidani, I., Ouni, A., Chouchen, M., & Mkaouer, M. W. (2020). Predicting continuous integration build failures using evolutionary search. *Information and Software Technology*, 128, 1–16.
- Saldaña, J. (2009). *The Qualitative Coding Manual*. Thousand Oaks.
- Santos, I., Magalhaes, C., & Santos, R. d. S. (2025). Model-Assisted and Human-Guided: Perceptions and Practices of Software Professionals Using LLMs for Coding. *arXiv preprint arXiv:2510.09058*.

- Sarker, S., Chatterjee, S., Xiao, X., & Elbanna, A. (2019). The sociotechnical axis of cohesion for the IS discipline: Its historical legacy and its continued relevance. *MIS Quarterly*, 43(3), 695–720.
- Sarker, S., Xiao, X., Beaulieu, T., & Lee, A. S. (2018). Learning from first-generation qualitative approaches in the IS discipline: An evolutionary view and some implications for authors and evaluators (PART 1/2). *Journal of the Association for Information Systems*, 19(8), 752–774.
- Sauvola, J., Tarkoma, S., Klemettinen, M., Riekk, J., & Doermann, D. (2024). Future of software development with generative AI. *Automated Software Engineering*, 31(1), 1–26.
- Schaufeli, W. B., Bakker, A. B., & Salanova, M. (2006). The measurement of work engagement with a short questionnaire: A cross-national study. *Educational and Psychological Measurement*, 66(4), 701–716.
- Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in machine learning systems. *Advances in Neural Information Processing Systems*, 28.
- Seidel, S., Berente, N., Lindberg, A., Lyytinen, K., & Nickerson, J. V. (2019). Autonomous tools and design: a triple-loop approach to human-machine learning. *Communications of the ACM*, 62(1), 50–57.
- Shaft, T. M., & Vessey, I. (2006). The Role of Cognitive Fit in the Relationship between Software Comprehension and Modification. *MIS Quarterly*, 30(1), 29–55.
- Singla, A., Sukharevsky, A., Yee, L., Chui, M., & Hall, B. (2025). The state of AI: How organizations are rewiring to capture value. *McKinsey & Company*, 12.
- Sommerville, I. (2015). *Software engineering*. 10th.
- Spreitzer, G. M. (1995). Psychological empowerment in the workplace: Dimensions, measurement, and validation. *Academy of Management Journal*, 38(5), 1442–1465.
- Star, S. L. (2010). This is not a boundary object: Reflections on the origin of a concept. *Science, Technology, & Human Values*, 35(5), 601–617.
- Star, S. L., & Griesemer, J. R. (1989). Institutional Ecology, 'Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907–39. *Social Studies of Science*, 19(3), 387–420.
- Storey, M.-A., Zimmermann, T., Bird, C., Czerwinka, J., Murphy, B., & Kalliamvakou, E. (2019). Towards a theory of software developer job satisfaction and perceived productivity. *IEEE Transactions on Software Engineering*, 47(10), 2125–2142.
- Strauss, A., & Corbin, J. (1998). *Basics of qualitative research techniques*. Sage Publications.
- Stray, V., Barbala, A., & Wivestad, V. T. (2025). Human-AI Collaboration in Software Development: A Mixed-Methods Study of Developers' Use of GitHub Copilot and ChatGPT. *Proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering*, 1325–1332.

- Strich, F., Mayer, A.-S., & Fiedler, M. (2021). What do I do in a world of artificial intelligence? Investigating the impact of substitutive decision-making AI systems on employees' professional role identity. *Journal of the Association for Information Systems*, 22(2), 9.
- Strong, D. M., Volkoff, O., Johnson, S. A., Pelletier, L. R., Tulu, B., Bar-On, I., Trudel, J., & Garber, L. (2014). A theory of organization-EHR affordance actualization. *Journal of the Association for Information Systems*, 15(2), 53–85.
- Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., & Müller, K.-R. (2021). Towards CRISP-ML (Q): a machine learning process model with quality assurance methodology. *Machine Learning and Knowledge Extraction*, 3(2), 392–413.
- Subramanyam, R., & Krishnan, M. S. (2003). Empirical analysis of ck metrics for object-oriented design complexity: Implications for software defects. *IEEE Transactions on Software Engineering*, 29(4), 297–310.
- Sundberg, L., & Holmström, J. (2024). Innovating by prompting: How to facilitate innovation in the age of generative AI. *Business Horizons*.
- Sunitha, E. V., & Samuel, P. (2019). Automatic code generation from UML state chart diagrams. *IEEE Access*, 7, 8591–8608.
- Swanson, E. B., & Beath, C. M. (1989). Reconstructing the systems development organization. *MIS Quarterly*, 13(3), 293–307.
- Tarafdar, M., & Saunders, C. (2022). Remote, Mobile, and Blue-Collar: ICT-Enabled Job Crafting to Elevate Occupational Well-Being. *Journal of the Association for Information Systems*, 23(3), 707–749.
- Teodorescu, M. H. M., Morse, L., Awwad, Y., & Kane, G. C. (2021). Failures of Fairness in Automation Require a Deeper Understanding of Human-ML Augmentation. *MIS Quarterly*, 45(3), 1483–1499.
- Tesch, D., Sobol, M. G., Klein, G., & Jiang, J. J. (2009). User and developer common knowledge: Effect on the success of information system development projects. *International Journal of Project Management*, 27(7), 657–664.
- Tian, Q., Bai, J., & Wu, T. (2022). Should we be” challenging” employees? A study of job complexity and job crafting. *International Journal of Hospitality Management*, 102, 1–15.
- Tim, Y., Pan, S. L., Ractham, P., & Kaewkitipong, L. (2017). Digitally enabled disaster response: the emergence of social media as boundary objects in a flooding disaster. *Information Systems Journal*, 27(2), 197–232.
- Tims, M., & Bakker, A. B. (2010). Job crafting: Towards a new model of individual job redesign. *SA Journal of Industrial Psychology*, 36(2), 1–9.
- Tims, M., Bakker, A. B., & Derks, D. (2012). Development and validation of the job crafting scale. *Journal of Vocational Behavior*, 80(1), 173–186.

- Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., & Bhosale, S. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- Ullsnes, R., Moe, N. B., Stray, V., & Skarpen, M. (2024). Transforming software development with generative AI: Empirical insights on collaboration and workflow. In *Generative ai for effective software development* (pp. 219–234). Springer.
- van den Broek, E., Sergeeva, A., & Huysman, M. (2021). When The Machine Meets The Expert: An Ethnography of Developing AI for Hiring. *MIS Quarterly*, 45(3), 1557–1580.
- Vara, J. M., & Marcos, E. (2012). A framework for model-driven development of information systems: Technical decisions and lessons learned. *Journal of Systems and Software*, 85(10), 2368–2384.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
- Venkatesh, V., Brown, S. A., Maruping, L. M., & Bala, H. (2008). Predicting different conceptualizations of system use: The competing roles of behavioral intention, facilitating conditions, and behavioral expectation. *MIS Quarterly*, 32(3), 483–502.
- Vessey, I., & Galletta, D. (1991). Cognitive fit: An empirical study of information acquisition. *Information Systems Research*, 2(1), 63–84.
- Waardenburg, L., & Huysman, M. (2022). From coexistence to co-creation: Blurring boundaries in the age of AI. *Information and Organization*, 32(4), 1–11.
- Wagner, S., & Ruhe, M. (2018). A systematic review of productivity factors in software development. *arXiv preprint arXiv:1801.06475*.
- Walz, D. B., Elam, J. J., & Curtis, B. (1993). Inside a software design team: knowledge acquisition, sharing, and integration. *Communications of the ACM*, 36(10), 63–77.
- Wan, Z., Xia, X., Lo, D., & Murphy, G. C. (2019). How does machine learning change software development practices? *IEEE Transactions on Software Engineering*, 47(9), 1857–1871.
- Watson, H. J., & Frolick, M. N. (1993). Determining Information Requirements for an EIS. *MIS Quarterly*, 17(3), 255–269.
- Weber, T., Brandmaier, M., Schmidt, A., & Mayer, S. (2024). Significant productivity gains through programming with large language models. *Proceedings of the ACM on Human-Computer Interaction*, 8(256), 1–29.
- Wrzesniewski, A., & Dutton, J. E. (2001). Crafting a job: Revisioning employees as active crafters of their work. *Academy of Management Review*, 26(2), 179–201.
- Wu, S., Liu, Y., Ruan, M., Chen, S., & Xie, X.-Y. (2025). Human-generative AI collaboration enhances task performance but undermines human’s intrinsic motivation. *Scientific Reports*, 15(1–30), 15105.

- Yang, J., Jin, H., Tang, R., Han, X., Feng, Q., Jiang, H., Zhong, S., Yin, B., & Hu, X. (2024). Harnessing the power of LLMs in practice: A survey on ChatGPT and beyond. *ACM Transactions on Knowledge Discovery from Data*, 18(6), 1–32.
- Yin, R. K. (2018). *Case study research and applications* (Vol. 6). Sage Thousand Oaks, CA.
- Yuan, Z., Lou, Y., Liu, M., Ding, S., Wang, K., Chen, Y., & Peng, X. (2023). No more manual tests? evaluating and improving chatgpt for unit test generation. *arXiv preprint arXiv: 2305.04207*.
- Zhang, F., & Parker, S. K. (2019). Reorienting job crafting research: A hierarchical structure of job crafting concepts and integrative review. *Journal of Organizational Behavior*, 40(2), 126–146.
- Zhang, J., & Norman, D. A. (1994). Representations in Distributed Cognitive Tasks. *Cognitive Science*, 18(1), 87–122.
- Zhang, J., Chen, Y., Liu, C., Niu, N., & Wang, Y. (2023). Empirical Evaluation of ChatGPT on Requirements Information Retrieval Under Zero-Shot Setting. *2023 International Conference on Intelligent Computing and Next Generation Networks (ICNGN)*, 1–6.
- Zhang, Z., Yoo, Y., Lyytinen, K., & Lindberg, A. (2021). The unknowability of autonomous tools and the liminal experience of their use. *Information Systems Research*, 32(4), 1192–1213.
- Zhao, X., Lynch Jr, J. G., & Chen, Q. (2010). Reconsidering Baron and Kenny: Myths and truths about mediation analysis. *Journal of Consumer Research*, 37(2), 197–206.
- Ziegler, A., Kalliamvakou, E., Li, X. A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., & Aftandilian, E. (2022). Productivity assessment of neural code completion. *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, 21–29.
- Ziegler, A., Kalliamvakou, E., Li, X. A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., & Aftandilian, E. (2024). Measuring github copilot’s impact on productivity. *Communications of the ACM*, 67(3), 54–63.

Paper I

Developing AI-based Systems – Bridging Knowledge Boundaries in the Triad

Mei Wu-Gehbauer
University of Cologne

In: Academy of Management Proceedings (Vol. 2025, No. 1)

Developing AI-based Systems – Bridging Knowledge Boundaries in the Triad

Abstract

As artificial intelligence (AI) increasingly supports knowledge work within organizations, understanding how AI systems are developed becomes essential. The development of AI systems marks a significant paradigm shift from creating rule-based IT artifacts to designing AI systems characterized by data-driven and agentic properties. AI systems construct their own decision logic, which can diverge from the knowledge and decision-making of human experts. We draw on the concept of knowledge boundaries to explore the practices that facilitate knowledge integration, bridging the knowledge domains of the AI system, the developer, and users in AI systems development. Our findings reveal that this triad of agents leverages different boundary objects to enact core practices that bridge the boundaries.

Keywords: Artificial intelligence, IS development, knowledge boundaries, data objects

1 Motivation

With the rapid proliferation of advanced artificial intelligence (AI) systems and their impact on how work is done in organizations, AI has become a top priority on the agenda of many management boards (Chui et al., 2023a). Spending on AI is expected to triple in the next years, with an annual growth rate of almost 30 percent (IDC, 2024). At the same time, firms struggle to realize benefits from AI (Bellefonds et al., 2024). Only a fraction of AI projects advance beyond the proof-of-concept stage and fail twice as often as traditional information technology (IT) projects (Baig et al., 2024; Bojinov, 2023; Fountaine et al., 2019). This raises the question of how the development of AI systems is different from traditional IT systems. As tech leaders put it in a recent global CIO Network Summit², the time has come for companies to leave the window of experimenting behind. It is timely and important to unpack how the unique characteristics of AI, with their capacity to learn, adapt, and act autonomously (Berente et al., 2021), impact the development of AI systems.

Developing AI systems is different from developing traditional IT systems (Rosenkranz et al., 2025). AI systems learn a *probabilistic decision logic* from data captured by the system's decision model (Foster, 2022; Schuetz & Venkatesh, 2020). This decision logic is generated by applying machine learning algorithms to learn rules and patterns from data autonomously. As a result, the outputs of AI systems based on a probabilistic decision logic might deviate from the human expert decision logic in a business context (Murray et al., 2021).

The shift to a self-learned decision logic represents a paradigm change for systems development

²<https://www.wsj.com/articles/companies-had-fun-experimenting-with-ai-now-they-have-to-show-the-returns-2a683592>

(Amershi et al., 2019a; Zhang et al., 2020). In traditional IT systems, the focus often is on the *dyadic relationship* between the developers and users, relying on existing, codified knowledge flowing from users to developers. Developers represent knowledge into rule-based algorithms for execution by the IT system to generate answers that support users in their tasks (Chakraborty et al., 2010). Here, the underlying assumption is that knowledge resides in different human actors and needs to be brought together for the IT system development (Alavi & Leidner, 2001; Patnayakuni et al., 2007). The advent of AI systems as agentic IS artifacts, which can learn, adapt, and act autonomously, challenges this assumption (Baird & Maruping, 2021; Chollet, 2021). AI systems are increasingly treated as actors in their own right (Baird & Maruping, 2021). Consequently, we argue that AI systems development hinges on coordinating the triad of developers, users, and the AI system.

We posit that AI systems can be treated as knowledge agents for three reasons: (1) AI systems learn from a data basis that differs from that of human actors (Lawrence, 2017; Lyytinen et al., 2021; Raisch & Fomina, 2025), as a result, (2) AI systems construct their own probabilistic decision logic from these data (Chollet, 2019; LeCun et al., 2015), and (3) their outputs are used as insights in knowledge work (Berg et al., 2023; Eloundou et al., 2023). This framing helps in clarifying implications for developing AI systems. It does not attribute human-like intentionality or epistemic states, such as belief, to the AI systems. These characteristics mean that, at the interface with human actors, AI systems introduce knowledge-related boundaries in the development process. Accordingly, this study asks the following research question: “*How are development practices enacted within the triad of developers, users, and the AI system to manage knowledge-related boundaries during systems development?*”

We leverage the concept of knowledge boundaries to conceptualize the knowledge-related boundaries between developers, users, and the AI system and to examine the practices through which these boundaries are bridged within the triadic relationship. We investigate interactions in this triad during the development of AI systems and conduct a case study across five AI development projects. We find that AI systems development can be understood as a process of managing knowledge-related boundaries at the interfaces among developers, users, and the AI system. This study thereby contributes to research on information systems development (ISD) of agentic IS artifacts.

2 Theoretical Background

2.1 From Deterministic Rule-based Systems to Probabilistic Agentic AI Systems

The development of traditional rule-based expert systems relies on human knowledge being transferred from users to developers. Developers represent this knowledge through determin-

istic logic that the IT system executes in order to provide answers effectively supporting users in their organizational tasks (Chakraborty et al., 2010; Newman & Robey, 1992). To achieve this goal, a variety of development practices are discussed in IS literature, ranging from user-centered approaches (e.g., Mao et al., 2005), user involvement practices (e.g., Maruping & Matook, 2020), prototyping (e.g., Naumann & Jenkins, 1982), and agile practices (e.g., Lee & Xia, 2010) where users iteratively test whether the outcomes are meaningful in practice and can engage in redesign activities. As a result, the deterministic decision logic is closely coupled with the expert knowledge and real-world practices of the users of the IT system. The underlying core assumption of these development practices is that knowledge resides in different human experts (developers or users) that need to be brought together for the IT system development to be useful and effective (Alavi & Leidner, 2001; Patnayakuni et al., 2007).

However, with the probabilistic and agentic nature of AI systems, this assumption is challenged (Chollet, 2019; Zhang et al., 2020). AI systems are based on a probabilistic decision logic grounded in connectionist principles from cognitive science, enabling them to autonomously learn and adapt (Fahlman & Hinton, 1987). AI systems trained through machine learning are not programmed with explicit expert rules. Instead, AI systems autonomously learn patterns from data and generalize from these patterns. Instead of representing human expert knowledge through *deterministic decision logic*, AI systems learn a own decision logic from data, resulting in a decision model that represents causalities and that enables the systems in their decision-making (Baird & Maruping, 2021). This marks a paradigm change for information systems development.

2.2 Developing AI Systems

AI systems are advanced information systems (IS) imbued with the capacity to learn, adapt, and act autonomously based on probabilistic decision logic to solve complex problems not being thought of as being amenable to IS support before (Baird & Maruping, 2021; Berente et al., 2021; Eloundou et al., 2023; Russell & Norvig, 2016; Schuetz & Venkatesh, 2020). Developing AI systems involves (1) inputs in the form of data, (2) an AI model learned from data with a probabilistic decision logic, and (3) AI outputs (OECD, 2024; von Krogh, 2018).

The starting point for developing AI systems is *data* as the principal material on which critical actions, commitments, and services are made (Alaimo & Kallinikos, 2022; Hartmann & Henkel, 2020). Discovering, sourcing, managing, and versioning data is inherently more complex and time-consuming than doing the same with software code (Amershi et al., 2019a). Data is used to train an *AI model* applying a combination of machine learning algorithms (Rokach, 2010). This process usually starts with an untrained algorithmic learning algorithm, such as linear regression, decision trees, support vector machines, or deep neural networks (LeCun et al., 2015).

Machine learning approaches can be broadly categorized into supervised and unsupervised learn-

ing (LeCun et al., 2015). Supervised learning relies on labeled data, where each input corresponds to an output label, allowing the model to learn a decision logic based on probabilistic reasoning between inputs and outputs (James et al., 2021). This method is often used for tasks such as classification and prediction, the most prominent use case for organizations. Take for example the use case of predicting the next best action for a customer calling at the service desk. The AI model would be trained by developers on specific customer data provided by users that aggregates all information, such as the customer call history, products in use, and usage behaviors, to learn a decision model to predict the next best action. In contrast, unsupervised learning involves the discovery of hidden patterns or structures within the data. Common unsupervised learning tasks include clustering, where data points are grouped based on similarity, and dimensionality reduction, which simplifies data by reducing the number of features while retaining key information (James et al., 2021).

The outputs of AI systems are the results produced by applying the trained model to new data. These outputs can take various forms, depending on the type of task and model used. For instance, in predictive models, outputs are typically probabilistic scores or classifications, such as predicting whether a customer is likely to churn. These outputs, in turn, become inputs for decision-making processes in practice or are used to automate a step in knowledge work.

AI systems with today's learning architectures are superior in handling uncertainty and incomplete information compared to traditional IT systems, but at the same time make the learned probabilistic decision logic less coupled with the user domain (Alaimo & Kallinikos, 2022). We know that the resulting probabilistic decision logic learned from data is only an approximation of the knowledge domains to which they are applied (Alaimo & Kallinikos, 2022). It may only capture explicit knowledge, but may not capture the rich know-how practices of experts in the field (Lebovitz et al., 2021) or the complexities of the work context (Asatiani et al., 2021). As a result, the link between the human expert knowledge and the probabilistic decision model learned from data is loosened. The disjunction between the learned representation of the AI system's decision model and the human expert knowledge depends on the context sensitivity of the task, the context specificity of the domain, and the data with which the AI model has been trained.

Therefore, the learned decision model with a probabilistic decision logic in AI systems development in an organizational context is a two-edged sword. On the one hand, it is a core feature of AI systems that enables learning, adaptation, and decision-making under uncertainty. On the other hand, it creates its own representation based on a learned decision model that is always only an approximation of the knowledge domains to which it is applied, thereby shifting the focus of developing AI systems from a dyad to a triad of agents that fundamentally alters the interaction and collaboration processes among developers, users, and the AI system.

2.3 Knowledge Boundaries and Boundary Objects in AI Systems Development

Building on the relational view of knowledge from Carlile (2004), knowledge boundaries arise when actors (1) hold different knowledge, (2) face novel circumstances, and (3) are mutually dependent on each other to accomplish their tasks. When these conditions are present, differences in expertise become consequential for coordinating work. Prior IS and management research has shown that integrating specialized and distributed knowledge across such boundaries is critical for effective systems development (Faraj & Sproull, 2000; Patnayakuni et al., 2007; Rus et al., 2002; Walz et al., 1993).

In the context of AI systems, the involvement of an agentic IS artifact means that these conditions also arise at the interface between humans and the AI systems, creating an additional knowledge-related boundary in Carlile's sense in ISD. First, there are differences in the amount and type of knowledge. AI systems can handle vast amounts of data at high speed and are trained on a variety of data that may span multiple organizational units and external sources, exceeding human capabilities (Baird & Maruping, 2021). They can uncover patterns from data that remain hidden to humans (Dhar, 2013), making the resulting insights AI systems create different both in amount and type.

Second, AI systems introduce novelty in how problems are represented and solved. Through machine learning, they acquire a probabilistic decision logic that is not explicitly programmed by developers but learned from data (Chollet, 2021). The resulting algorithmic rules are often opaque to humans (Burrell, 2016) and can produce recommendations or patterns that diverge from established human decision logics (Murray et al., 2021). For users and developers, this means that AI systems can frame problems in unfamiliar ways, surface unexpected causalities, and suggest actions that are difficult to reconcile with existing domain knowledge. This novelty is a core source of knowledge boundaries in AI systems development.

Third, these differences and novelties become consequential because humans and AI systems are increasingly mutually dependent on each other's contributions to accomplish knowledge-intensive tasks (Raisch & Krakowski, 2021). While Carlile (2004) distinguishes between syntactic, semantic, and pragmatic boundaries, we focus in this study on the relational conditions under which boundaries become consequential, rather than on categorizing boundary types.

Taken together, AI systems introduce knowledge-related boundaries at the interfaces among users, developers, and the AI system that must be actively managed during AI systems development. Therefore, in this study, we analyze how developers enact development practices to bridge knowledge-related boundaries within the triad of developers, users, and the AI system.

To understand how the agents in the triad of AI systems development bridge knowledge-related boundaries in practice, we draw on the concept of boundary objects (Star, 2010; Star & Gries-

mer, 1989). Boundary objects are artifacts that are sufficiently plastic to adapt to the different needs of parties employing them, while being robust enough to maintain a common identity across the parties (Star & Griesemer, 1989).

Star (2010) emphasizes that boundary objects are part of an arrangement of work and information that emerges from information and work requirements. In IS and organization studies, boundary objects have been used to explain how diverse communities of practice coordinate work using documentation, prototypes, data, or systems (Levina & Vaast, 2005; Majchrzak et al., 2012; Pawlowski & Robey, 2004; Tim et al., 2017). In line with this work, we focus our study on boundary objects that are relevant to bridge knowledge-related boundaries in the triad of developers, users, and the AI system during systems development.

3 Research Method

We employ an embedded case study design at a leading international telecommunications enterprise. Case studies are well established in IS research to generate an understanding of the socio-technical system and allow for developing an in-depth understanding of information systems in their contexts (Yin, 2009). We study the development of five AI systems that support users in problem solving from the developer perspective as well as from the perspective of users and explore the interactions among the developers, users and the AI systems. This enables us to gain an in-depth understanding of the practices to develop the AI systems from a socio-technical perspective. Specifically, we aim to generate theoretical understanding on how to bridge the knowledge-related boundaries in the triad of developers, users, and the AI system in the development of those systems.

As our case organization, we selected a leading international telecommunications enterprise. The company is regarded by industry analysts as one of the most advanced telecommunications operators in terms of AI adoption. It has established extensive data and AI platforms and capabilities, and runs AI development projects across multiple business areas that implement AI systems to translate the potential of AI into business value.

3.1 Data Collection

Our primary data sources were interviews, project documentation, and video material. We sampled projects based on two criteria: (1) they developed AI systems that support users in their business domain work, and (2) the AI systems were either already in productive use or in a late stage of development before launch at the time of data collection.

To gain a thorough understanding of how the AI systems are developed to support knowledge work in organizations, we interviewed 24 informants from five AI systems projects in the time-frame between January 2023 and August 2023. As informants we focused on two groups of

human actors per project: the developers and the users. With user, we refer to domain experts knowledgeable about the user domain and who are involved in the AI systems development process. To address our research question, it is essential to include both perspectives. Consequently, we sampled interviewee pairs of developers and users for every project. Each interview lasted between 38 and 68 minutes and took place either onsite at the case company or via online video conferencing. We followed a semi-structured approach to conduct the interviews, using an interview protocol and at the same time leaving space for probing based on the informants' responses. In the interviews, we aimed to understand knowledge-related boundaries during development as well as actions taken to develop the AI systems from the perspectives of developers and users. Table 1 provides an overview of the interview data.

Table 1: Interview data

Project ID	# Developers	# Domain Experts	Total # of Interviews	Interview Minutes	No of Docs & Video
A	4	3	7	363	15
B	2	4	6	307	4
C	3	1	4	213	12
D	2	1	3	170	14
E	2	2	4	200	2
Total	13	11	24	1.253	47

Complementing the interviews, we accessed and studied additional data sources in form of project documentation and video materials describing the development of the AI systems and the resulting systems. These additional data sources enabled a validation of our understanding of the development process of the AI systems as well as the systems features, allowing us to triangulate our findings. Using the interviews and the additional materials, we constructed a detailed description of each AI system. Table 2 provides a summary of the AI systems in scope including the decision-making prior to AI versus decision-making with AI.

3.2 Data Analysis

For data analysis, we used a form of grounded theorizing (Strauss & Corbin, 1998) drawing on data from our revelatory case study. We organized our analysis around the interactions among the developers, users, and the AI system to identify practices that bridge the knowledge-related boundaries in the triad. We focus on development practices enacted by developers, examining the actions they take in interaction with users and the AI system. In doing so, we also consider users' actions, such as providing feedback, as part of the practices through which AI systems are developed. These actions form the core of getting a deeper understanding of how AI systems are developed for supporting knowledge work.

Table 2: Overview of AI systems: Decision making prior to versus with AI

Project	Task & Previous Decision Making of Human Experts	AI System	Decision Making with AI
Project A: Campaign personalization	Target customer audience selection for campaigns and offers using customer scoring and product affinity based on manually defined rules Campaign managers create the offer and use the scores to decide on customer selection	AI system as decision engine for target audience selection for personalized customer offers based on both offline and online customer data AI system developed using algorithms employing decision tree ensemble methods, reinforcement learning, deep neural network	Campaign managers create the offer AI decision engine decides on customer selection
Project B: Predictive and prescriptive mobile network maintenance	In case of incidents radio field optimizer analyzes mobile network data of incident cases to identify root causes Based on root cause analysis, rules are manually specified and implemented to resolve current incidents as well as to mitigate future incidents resulting from the same root causes	AI system predicts incidents based on probability for call drops, prioritizes incident criticality, and identifies root causes as well as mitigation rules AI system developed using algorithms employing supervised, non-linear prediction and classification methods such as decision tree ensemble methods	AI system prioritizes incident cases as basis for proactive maintenance and optimization by radio field optimizers AI system assists root cause analysis by classifying root causes and identifying mitigation rules, thereby providing a holistic view instead of case-by-case analysis; AI-identified mitigation rules are verified by radio field optimizer before implementation
Project C: Predictive fixed-line network maintenance	Based on customer reported disruptions in the fixed-line network, technicians initiate troubleshooting measures, and analyze network data to address the issues	AI system predicts incidents based on anomalies in the performance data of the fixed-line network AI system developed using algorithms employing unsupervised machine learning methods and supervised classification methods	AI system identifies anomalies, predicts fixed-line network issues, and automatically generates problem tickets Technicians proactively initiate troubleshooting measures using AI-generated insights without requiring customers to report the issue
Project D: Predictive and prescriptive home network optimization	Based on customer reported Wi-Fi internet issues, technicians use data of current home network topology and additional information requested from customers for troubleshooting	AI system predicts home network issues based on current as well as historical home network topology and device parameters and past incident diagnostics AI system developed using algorithms employing unsupervised machine learning methods and supervised classification methods	AI system predicts home network issues, performs diagnostics, and provides recommendations for troubleshooting Technicians proactively initiate troubleshooting using AI-generated insights for faster root cause analysis and personalized recommendations for improving customer experience in the home network
Project E: Predictive and prescriptive customer service	Customer-reported issues and tickets are handled by the first-level support If the first-level support team cannot resolve the ticket, it is forwarded based on manually defined rules to the technical customer support team, then to the field service team, if required	AI system as decision engine for ticket routing based on ticket data, historical customer interaction data, as well as technical network data AI system developed using algorithms employing supervised classification methods	AI system predicts which of the support teams – first-level support, technical customer support, or the field service team – is best suited to address the issue Based on this prediction, the AI system automatically routes the customer ticket to the identified team, enabling faster issue resolution

Coding is a cyclical act (Saldaña, 2009), and our coding process was highly iterative. Emerging concepts and relationships informed further theoretical sampling, data analysis, and concept refinement. In the initial analysis stage (open coding), we identified initial concepts such as “data feature engineering” or “understanding of data attributes and feature importance” from the collected data. We focused on interactions among developers, users, and the AI system, and in particular on the practices the actors applied to address knowledge-related boundaries. We generated open codes that captured which knowledge-related boundaries occurred during development, and how the actors responded to them. Following the data reduction process from Miles and Huberman (1994), we progressively focused, simplified, and abstracted the data while remaining open to salient concepts in the data. We coded all interviews line by line, producing open codes that we then further grouped and integrated into first-order concepts. For example, the concept “user feedback collection” emerged from codes such as “user feedback on required data”, “user feedback on incorrect data input”, and “user feedback on data sources”.

During this open coding, it became evident that boundary objects played a central role as artifacts in the development practices. We therefore began to code boundary objects explicitly, alongside knowledge-related boundaries and development practices. We triangulated our coding with additional materials, such as AI system documentation and demo presentations, to increase the validity of our results.

In axial coding, we further developed categories and systematically related these to the identified first-order concepts from our open coding process. For example, the category “integrating AI decision logic with human understanding” relates to overcoming knowledge boundaries between the human decision logic and the AI system for humans (both users and developers) to grasp the AI’s learned decision logic. The axial coding step was informed by the theoretical lens of knowledge boundaries and boundary objects. This lens suggests focusing on integrating the knowledge domains. We used the lens to identify practices in the triadic interaction to elaborate on the axial categories, thereby relating our analysis to our research question.

During selective coding, we centered on four core categories of bridging the knowledge-related boundaries. We found that the four core categories are interdependent during the development process, iteratively addressing the knowledge-related boundaries within the triad.

4 Findings

4.1 Knowledge Boundaries in AI Systems Development

Our findings show that developers, users, and the AI system encounter several knowledge boundaries that emerge among all three agents during the development process of the AI system. We illustrate the knowledge boundaries that emerged in our case study along the stages of the AI systems development process that are input to the AI system, learning by the AI system, and

output of the AI system influencing its environment (OECD, 2024; von Krogh, 2018).

AI input: data provisioning. The most important input for AI systems is data. As one developer noted, *“one of the important exchanges we have with users was on deciding which data and which attributes we have to use for our modeling approach.”* (Peter, developer). The expertise for domain- and task-relevant data lies with users, who can point developers to data sources and factors they consider important based on their decision logic. At the same time, they usually lack transparency about how much data is needed, how it will be processed by the AI system, or which data types are suitable for a given algorithm. In the campaign management project, for instance, developers had to combine historical campaign data with real-time online data. In the mobile network project, large-scale data logs were needed to predict failures proactively. Discovering and sourcing such data *“from many data sources sitting in different corners of the organization on old and new platforms and databases”* required months of work and coordination and still left *“a lot of noise in the data”* (Amir, developer). Users therefore took over much of the data preparation: *“We do the data preparation because we have the domain expertise to understand and interpret the data”* (Torsten, user). Developers then use analytical approaches to focus on a statistically valid data scope. This knowledge-related boundary in data provisioning can hinder developers in identifying appropriate training data and, in turn, limit the AI system’s ability to learn from relevant data.

AI input: algorithm implementation. In the second phase, the boundary runs in the opposite direction: users struggle to understand the algorithms that developers implement and to grasp how AI systems differ from traditional IT systems they are used to. To process data, developers *“decide on the AI algorithms based on data volume, type of data, and distribution of available data.”* (Amir, developer). These choices are rarely specified by users. Instead, developers rely on their technical knowledge and experience in selecting the algorithms and optimization objectives. As one developer explained, they draw on their *“intuition that this family of decision trees works better with the type of data we have with a large imbalance between the positive and negative labels.”* Users, by contrast, are accustomed to deterministic rules and “perfect solutions”, as one user put it: *“Just a year or two ago, we couldn’t accept that there are also solutions that are only 80 percent accurate and 20 percent incorrect”* (Thomas, user). In the marketing project, for example, developers recommended starting with *“predicting 20 target categories instead of 200 categories as proposed by the user”* (Peter, developer) to obtain statistically more reliable results. Users, however, had *“doubts with this approach and were concerned with missing out key target customers”* (John, user). In this phase, the knowledge-related boundary arises because algorithm implementation embeds trade-offs, for example, between coverage and reliability, that are grounded in specialized developer expertise but that users must understand and integrate with their business goals to meaningfully work with the AI system.

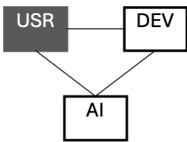
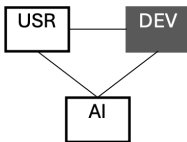
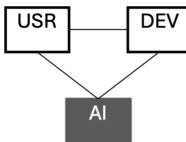
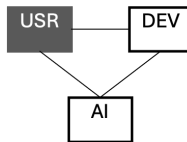
Learning: algorithmic opacity. In the third phase, algorithmic opacity becomes central. The AI system’s decision logic is learned from data and is often not fully understandable, even to

developers. Both users and developers struggle to interpret how the system reaches its decisions. A user stressed that *“it is indeed important to understand how the algorithm generates the outputs. If I have a result and have no idea how that result came about, I can’t interpret it properly. I don’t know which levers to adjust to get the right result”* (Leo, user). Users also questioned *“why certain patterns have been identified and why certain data features are important”* (Adam, developer). Developers likewise struggle with this opacity, as one developer explained: *“A neural network is usually a kind of black box. It is so complex that it’s hard to say what is going on, which is why we used a proxy model to have at least a proxy understanding of the parameters. To really see how the information flows inside the net is another area of science. It’s a lot of work, and our knowledge is limited”* (Max, developer). In this phase, the knowledge-related boundary arises because the probabilistic, data-driven decision logic of the AI system remains partially opaque, impeding users’ and developers’ ability to interpret outputs and understand how to influence the system’s behavior.

Output: interpretation and actionability. In the fourth phase, the boundary becomes a practical boundary. Even when users understand the AI output, acting on it can conflict with existing goals, organizational constraints, or task-related constraints. Working with AI outputs, therefore, requires *“a rethinking of how our work is done”* (John, user). The AI system optimizes globally, whereas users are used to more decentralized, local decisions. This also triggered a rethinking in AI projects, as this quote shows: *“In the future we want to move away from the decentralized decisions of campaign managers and come to a situation where we increasingly use the central decisions that come from the AI tool [...] but this is hard to accept for many campaign managers”* (John, user). Across projects, existing business rules relevant for task accomplishment further constrained the use of AI outputs. Developers noted that *“marketers have their own business objectives, which we need to consider [...] even when we understand how the AI outputs were generated”* (Peter, developer). In the service department, the AI system’s predictions produced strongly fluctuating call volumes, but managers worried that *“if they are not handling any calls for two hours, what are the agents doing? That does not work with how the organization is currently set up”* (John, user). In this phase, the knowledge-related boundary arises because algorithmic recommendations must be reconciled with human goals and task requirements, which can impede users’ ability to act on AI outputs in human-AI collaboration.

Across the AI systems development phases, different knowledge-related boundaries become salient, not only between human actors, but also between humans and the AI system, depending on who holds critical knowledge and who must access it to move development forward. Table 3 summarizes these shifting knowledge-related boundaries for each phase of AI systems development.

Table 3: Knowledge-related boundaries along key phases of AI systems development

Knowledge Boundary	Phases of AI Systems Development			
	Input: Data Provisioning	Input: Algorithms	Learning: Algorithmic Opacity	Output Interpretation and Actionability
Knowledge boundary description	Developers and the AI system depend on domain-relevant input data held and interpreted by users	Algorithm implementation depends on the developers' expertise inaccessible to users	AI's decision logic, learned from data, is opaque to users and partially to developers	AI outputs resulting from algorithmic optimization must be reconciled with domain-specific goals, business rules, and task constraints
Direction of effective boundary				
Impact of boundary	Impeding developers' ability to identify and prepare task-relevant data for the AI system to learn from	Impeding users' ability to comprehend key characteristics of AI system	Impeding ability of users and developers to comprehend AI system's decision logic	Impeding users' ability to act on the AI system's outputs in line with organizational goals and task requirements
Knowledge resides in x Knowledge to be accessed by y				

4.2 Development Practices and Boundary Objects

Our second set of findings concerns how knowledge-related boundaries during AI systems development are addressed in practice and what role data objects play as boundary objects in this process. For each phase of AI systems development, we identify development practices that bridge the respective knowledge-related boundary and the boundary objects through which this boundary work is accomplished. We begin with the data provisioning boundary.

Data provisioning: integrating user knowledge. In the data provisioning phase, developers depend on users with domain expertise to identify, collect, and iteratively refine the data that feeds AI systems. The key development practice here is *integrating user knowledge in the form of domain data*. Users and developers work together to decide which data sources to use, which attributes matter, and how these should be represented for model training. Users contribute their decision logic and contextual understanding of the tasks to be accomplished with AI systems, while developers bring knowledge of data quality and formats.

This practice is realized through two types of data used as boundary objects: *system-generated data*, where users are extraneous to data production, and *user-shaped data*, where users are involved in data production. On the system-generated side, for example, online trace data are captured automatically as customers interact with the shopping basket: *“A customer places a product in the shopping basket but cancels just before checkout. Later, the customer is contacted about unrelated products, and the original customer interest is ignored. This clearly shows that our customer communication must build on actual customer behavior. This online data shouldn’t be left unused”* (John, user). Here, system-generated logs become boundary objects that developers can analyze, but users did not shape the data structure upfront. Users are, however, essential in recognizing which traces are relevant to solve the business problem.

On the side of the user-shaped data, we found data where users actively shape how this data are organized before they are used for model training. In one project, users helped to create a household view by combining fixed-line and mobile service data: *“How can we better use household data for analytics and marketing? Fixed-line and mobile data are often viewed separately, but combining them, for example, the number of people or number of SIM cards, is key for deep selling. That’s the idea behind our approach for modelling a household view”* (Sarah, user). Users defined which contracts belong to a household, how to link individuals to a shared home line, and which attributes, such as household size, are meaningful. Developers then implemented these definitions in the data lake and feature pipeline as reusable data objects.

These practices turn both system-generated and user-shaped data into boundary objects that developers and users can work with. In doing so, user domain knowledge is applied to select and structure the data. This work bridges the knowledge-related boundary in data provisioning: it helps developers identify appropriate training data and enables the AI system to learn from data that reflects domain and task relevance.

Algorithm implementation: integrating developer knowledge. In the algorithm implementation phase, the focus shifts to developers and how their expertise is used both to select and configure suitable algorithms and to help users understand algorithmic behavior. Developers decide which model families and optimization objectives fit the problem and data.

To make these choices tangible for users, we found that developers used the *AI system itself and simulations* of input-output relationships as boundary objects. In one project, for example, the developers *“simply drew a few thousand samples and looked at how it’s shaping up. Under which rules does the data point fall? How likely is it to fall in or out? It was quite interesting for the users to see that these aren’t hard numbers. The users treated the rules [identified by the algorithms] with less respect and developed a healthy skepticism”* (Adam, developer). Another developer emphasized that there is not *“perfect”* model and that users must actively make trade-offs: *“I don’t get a perfect model that does everything. The model can be very good, but I still have to make a decision using a kind of slider. So we built a what-if model: ‘You set the slider, choose a cut-off, and see what happens when 100 tickets come in. How often are we right, how often wrong, and are you willing to pay that price?’”* (Yuki, developer).

By exposing algorithmic behavior through such simulations and interactive threshold setting of the AI system, developers bridge the knowledge-related boundary: users gain an improved understanding of how the chosen algorithm operates on their data, which opens space for questioning and refining model settings as well as the use of the AI system’s outputs.

Algorithmic opacity: integrating AI decision logic with developers and users. In the learning phase, where algorithmic opacity becomes central, both developers and users struggle to understand the internal decision logic of the AI system. Developers are aware that models, which perform well in development, can behave unpredictably in production. As one developer explained: *“When I hand over a model, I want it to work. I don’t want emails a month later saying ‘this is all rubbish’. If you let a model loose, you often get rubbish. I have countless examples of highly non-linear models that look great on paper but are complete nonsense in production. That’s why I rely on more explainable models where I can still look under the hood”* (Adam, developer).

To bridge this boundary for themselves and for users, developers make the model’s key attributes and their feature importance explicit and discuss them with users. Together, they explore which data features and feature combinations drive predictions and how these relate to users’ own practices: *“Together with the business users, we have very deep discussions on why the model delivers certain results, why we have spikes. We look together into which data features, and feature combinations play a role, and what the main drivers are. The users examine single cases and retrace what has happened.”* (Amir, developer).

Here, data objects generated by the AI system, such as feature importance rankings, act as boundary objects. User representatives in the developer team connect these features with domain terms

for users to integrate the AI decision logic with their domain knowledge: *“We derived the feature importance and used it to say, once again: ‘Look, here are the connections.’ It all made perfect sense, and of course, it helps when you can see, for example, that there’s a prepaid card in the household. The more prepaid cards there are, the more likely it is that you can successfully sell a Family Card. These are insights that everyone can understand, and that usually makes a difference”* (Sarah, user).

Explainability is also important for developers themselves. A user, who is representing the users and is a part of the developer team, described how going through feature importance together built confidence in their own development work: *“When we had the feature importances, we went through them as a team. We wrote down hypotheses about each relationship. Sometimes it was obvious, sometimes more questionable. You could see how it sparked discussion. [...] It was also reassuring to see that the overall direction made sense. It builds trust in our own work [in the development team]”* (Sarah, user).

By turning internal model feature importance into interpretable data and jointly scrutinizing them, developers manage their own uncertainty about complex models while integrating AI decision logic with users’ domain knowledge. This practice helps both developers and users in making sense of how the AI system arrives at its outputs and thereby bridges the knowledge-related boundary created by algorithmic opacity.

Output interpretation and actionability: integrating user knowledge for actionability. In the final phase of AI systems development, output interpretation and actionability, the boundary is less about understanding the AI system’s outputs and more about aligning them so that users can act on them. Even if an AI system suggests the technically optimal action, it may violate business goals and task constraints. We found that bridging the boundary required integrating user knowledge required for task performance in the form of business rules and key performance indicators (KPIs) to guide the AI system.

A central development practice is encoding domain- and task-specific constraints as business rules that guide how AI outputs may be used. In the campaign management project, for example, the AI system identifies customers who are likely to respond to a campaign, but certain customers must not be contacted for specific offers and are maintained on blacklists. As one product owner explained: *“Certain exclusions still apply [on AI outputs], such as blacklists. That still sits on top of it”* (Sarah, user).

Knowledge objects in the form of business rules act as boundary objects in this development practice: they encode domain- and task-related rules and constraints that have to be considered when performing a domain-specific task.

A second practice is linking AI use to business KPIs to control for the superiority of human-AI collaboration. Developers establish metrics to compare outcomes with and without AI outputs, using these KPI objects to decide when and how AI outputs should inform action. Such KPIs

help users assess whether following the AI system's suggestions actually improves performance and where they should act based on the AI system's outputs.

Through these practices, user knowledge is integrated with the AI system via business rules and KPI objects. These boundary objects provide a means to reconcile algorithmic decision logic with organizational goals and constraints, thereby bridging the knowledge-related boundary that arises when transforming AI outputs into actionable decisions.

5 Discussion

During AI systems development, knowledge-related boundaries arise not only between human actors but also between humans and the agentic AI system. For each phase of AI systems development, we identify such boundaries and show how developers enact practices to bridge them using different boundary objects. Table 4 summarizes these development practices and the boundary objects employed in each phase.

Table 4: Development practices and boundary objects

Knowledge Boundary	Phases of AI Systems Development			
	Input: Data Provisioning	Input: Algorithms	Learning: Algorithmic Opacity	Output Interpretation and Actionability
Development practice	Integrating user knowledge by selecting and structuring input data drawing on user domain knowledge	Integrating developer knowledge to configure algorithms and expose the algorithmic behavior of the AI system	Integrating AI decision logic by tracing which attributes the AI system relies on most strongly for its outputs	Integrating user knowledge in the form of business rules and key performance indicators into the AI system
Impact of boundary	System-generated data , where users are extraneous to the data production User-shaped data , where users are involved in the data production	AI system and the simulations / what-if analyses of input-output relationships	Data objects generated by the AI system , such as feature importance scores and key driver attributes, which make the model's main decision drivers visible	Knowledge objects in the form of business rules that guide how the AI system generates outputs and how these may be used, and in the form of KPIs that evaluate and control the task performance in human-AI collaboration

With this study, we contribute to the literature on information systems development (ISD). In systems development, prior work has largely focused on a dyad of human agents, where knowledge is assumed to flow from users to developers so that human decision logic can be encoded in rule-based information systems (Chakraborty et al., 2010; Newman & Robey, 1992). We extend this view by conceptualizing AI systems development as a triad of agents, that are users,

developers, and the agentic AI system, in which knowledge-related boundaries must be managed across all three agents.

This triadic conceptualization offers a more comprehensive understanding of where and how boundaries arise when developing AI systems. Existing research on AI systems development has primarily emphasized integrating user domain knowledge into data as input for AI systems (e.g., Lebovitz et al., 2021; Sundberg & Holmström, 2024). By mapping knowledge-related boundaries across the triad, our study shows that this is only one part of the challenge. In addition, in the phase output interpretation and actionability, user task and decision logic required for task completion must be integrated into the AI system via business rules and KPIs that frame, for example, how model outputs are used. These rules and KPIs encode domain- and task-specific constraints and success criteria, enabling users to act on AI outputs in line with organizational goals and task requirements.

Additionally, prior studies on the development of AI systems largely treat developers and the AI system as belonging to a single technical knowledge domain, emphasizing workflows, pipelines, and quality assurance for AI systems (Amershi et al., 2019a; Pfeuffer et al., 2023; Sculley et al., 2015; Studer et al., 2021). While developers and AI systems indeed operate within the same technical domain comprising models, data, and infrastructure, our study shows that a knowledge-related boundary nonetheless exists between developers and the AI system. First, it is primarily developers' choices, rather than user specifications, that determine the algorithms and learning architectures of the AI system and thus shape how it can learn. Second, our study shows that developers themselves struggle with the agentic and opaque characteristics of the AI system and must enact practices to overcome this boundary and manage their own uncertainty during the development of the AI system.

Another contribution of this study is to specify which artifacts function as boundary objects when bridging knowledge-related boundaries in the triad. Existing IS and organization studies underscore the centrality of data and data objects in AI systems development, showing how data is produced, curated, and assembled into organizational resources (Aaltonen et al., 2021; Alaimo & Kallinikos, 2021; Muldoon et al., 2024). Building on this work, we show that different boundaries during AI systems development require additional boundary objects beyond data objects.

First, in the algorithm implementation phase, we demonstrate how the AI system itself, together with simulations and what-if analyses of input-output relationships, serves as a boundary object. These system artifacts make algorithmic behavior visible on concrete cases, allowing users and developers to see how algorithms behave on their data and to refine model settings. Second, in the output interpretation and actionability phase, we show that users' business knowledge objects in the form of business rules and KPIs are essential boundary objects. While data objects serve as inputs from which the AI system learns and constructs its decision logic, business rules and KPI objects codify domain- and task-specific constraints and evaluation criteria that govern

how AI outputs can be transformed into action. These findings qualify data objects as one class of boundary objects and foreground additional model artifacts and business knowledge objects through which developers and users manage knowledge-related boundaries in systems development.

For practitioners, we sensitize organizational actors involved in AI systems development to the knowledge-related boundaries that arise across all development phases. We delineate phase-specific development practices and associated artifacts that function as boundary objects for addressing these boundaries. This helps practitioners more deliberately structure collaboration between users, developers, and AI systems, and to design AI systems, in which AI outputs can be meaningfully understood and acted upon.

This study has limitations that suggest avenues for future research. The findings are derived from a qualitative case study at a large European telecommunications provider. This design enables rich insights into how development practices unfold in the triad of developers, users, and the AI-based system. The applicability of the findings to other organizational settings, however, needs to be examined and demonstrated in future research. Future research could conduct case studies across industries (e.g., manufacturing, public sector) and organizational sizes to examine how different resources or governance arrangements shape which boundaries emerge in the triad, which development practices are enacted, and which boundary objects are drawn on in bridging these boundaries.

6 Conclusion

Developing information systems (IS) to support organizational knowledge work has long been a central concern in IS research (Markus et al., 2002). Improvements in IS development practices are tightly linked to organizational performance and can catalyze new forms of organizing and working (Bostrom & Heinen, 1977b; Clegg, 2000). With the rapid growth of data and the proliferation of advanced AI systems, work is again being reconfigured as AI increasingly augments humans in performing organizational tasks (Brynjolfsson et al., 2023; Chui et al., 2023a).

Against this backdrop, our study advances a view of AI systems as agentic IS artifacts and conceptualizes AI systems development as a triadic, boundary-spanning process involving developers, users, and the AI system. We show how different knowledge-related boundaries become salient across phases of AI systems development and identify the development practices and boundary objects, which are data objects, model artifacts, and business knowledge objects, through which these boundaries are bridged. In doing so, the study extends ISD research beyond the traditional user-developer dyad and specifies how the AI system's decision logic is aligned with domain and task requirements. For practitioners, the findings offer a structured account of where knowledge-related boundaries arise and which phase-specific practices and artifacts can be mobilized to manage them. This supports the design of AI systems that can be meaningfully

used in organizational work.

References

- Aaltonen, A., Alaimo, C., & Kallinikos, J. (2021). The making of data commodities: Data analytics as an embedded process. *Journal of Management Information Systems*, 38(2), 401–429.
- Alaimo, C., & Kallinikos, J. (2021). Managing by data: Algorithmic categories and organizing. *Organization Studies*, 42(9), 1385–1407.
- Alaimo, C., & Kallinikos, J. (2022). Organizations decentered: Data objects, technology and knowledge. *Organization Science*, 33(1), 19–37.
- Alavi, M., & Leidner, D. E. (2001). Knowledge management and knowledge management systems: Conceptual foundations and research issues. *MIS Quarterly*, 25(1), 107–136.
- Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmermann, T. (2019a). Software engineering for machine learning: A case study. *IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 291–300.
- Asatiani, A., Malo, P., Nagbol, P. R., Penttinen, E., Rinta-Kahila, T., & Salovaara, A. (2021). Sociotechnical Envelopment of Artificial Intelligence: An Approach to Organizational Deployment of Inscrutable Artificial Intelligence Systems. *Journal of the Association for Information Systems*, 22(2), 325–352.
- Baig, A., Merrill, D., & Sinha, M. (2024). Moving past gen AI's honeymoon phase: Seven hard truths for CIOs to get from pilot to scale. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/moving-past-gen-ais-honeymoon-phase-seven-hard-truths-for-cios-to-get-from-pilot-to-scale>
- Baird, A., & Maruping, L. M. (2021). The Next Generation of Research on IS Use: A Theoretical Framework of Delegation to and from Agentic IS Artifacts. *MIS Quarterly*, 45(1), 315–341.
- Bellefonds, N. d., Charanya, T., Franke, M. R., Apotheker, J., Forth, P., Grebe, M., Luther, A., Laubier, R. d., Lukic, V., Martin, M., Nopp, C., & Sassine, J. (2024). Where's the Value in AI? <https://www.bcg.com/publications/2024/wheres-value-in-ai>
- Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing Artificial Intelligence. *MIS Quarterly*, 45(3), 1433–1450.
- Berg, J. M., Raj, M., & Seamans, R. (2023). Capturing value from artificial intelligence. *Academy of Management Discoveries*, 9(4), 424–428.
- Bojinov, I. (2023). Keep Your AI Projects on Track. <https://hbr.org/2023/11/keep-your-ai-projects-on-track>
- Bostrom, R. P., & Heinen, J. S. (1977b). MIS problems and failures: A socio-technical perspective. Part I: The causes. *MIS Quarterly*, 1(3), 17–32.
- Brynjolfsson, E., Li, D., & Raymond, L. R. (2023). Generative AI at work.

- Burrell, J. (2016). How the Machine ‘Thinks’: Understanding Opacity in Machine Learning Algorithms. *Big Data & Society*, 3(1), 1–12.
- Carlile, P. R. (2004). Transferring, translating, and transforming: An integrative framework for managing knowledge across boundaries. *Organization Science*, 15(5), 555–568.
- Chakraborty, S., Sarker, S., & Sarker, S. (2010). An Exploration into the Process of Requirements Elicitation: A Grounded Approach. *Journal of the Association for Information Systems*, 11(4), 212–249.
- Chollet, F. (2019). On the measure of intelligence. *arXiv preprint arXiv:1911.01547*.
- Chollet, F. (2021). *Deep learning with Python*. Simon; Schuster.
- Chui, M., Hazan, E., Roberts, R., Singla, A., Smaje, K., Sukharevsky, A., Yee, L., & Zimmell, R. (2023a). The economic potential of generative AI: The next productivity frontier. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier>
- Clegg, C. W. (2000). Sociotechnical principles for system design. *Applied Ergonomics*, 31(5), 463–477.
- Dhar, V. (2013). Data science and prediction. *Communications of the ACM*, 56(12), 64–73.
- Eloundou, T., Manning, S., Mishkin, P., & Rock, D. (2023). Gpts are gpts: An early look at the labor market impact potential of large language models. *arXiv preprint arXiv:2303.10130*.
- Fahlman, S. E., & Hinton, G. E. (1987). Connectionist architectures for artificial intelligence. *Computer*, 20(01), 100–109.
- Faraj, S., & Sproull, L. (2000). Coordinating expertise in software development teams. *Management Science*, 46(12), 1554–1568.
- Foster, D. (2022). *Generative Deep Learning*. O’Reilly Media, Inc.
- Fountain, T., McCarthy, B., & Saleh, T. (2019). Building the AI-powered organization. *Harvard Business Review*, 97(4), 62–73.
- Hartmann, P., & Henkel, J. (2020). The rise of corporate science in AI: Data as a strategic resource. *Academy of Management Discoveries*, 6(3), 359–381.
- IDC. (2024). *IDC Unveils 2025 FutureScapes: Worldwide IT Industry Predictions*. <https://www.idc.com/getdoc.jsp?containerId=prUS52691924>
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An introduction to statistical learning - second edition*. Springer.
- Lawrence, N. D. (2017). Living together: Mind and machine intelligence. *arXiv preprint arXiv:1705.07996*.
- Lebovitz, S., Levina, N., & Lifshitz-Assaf, H. (2021). Is AI Ground Truth Really ‘True’? The Dangers of Training and Evaluating AI Tools Based on Experts’ Know-What. *MIS Quarterly*, 45(3), 1501–1525.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.
- Lee, G., & Xia, W. (2010). Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. *MIS Quarterly*, 34(1), 87–114.

- Levina, N., & Vaast, E. (2005). The Emergence of Boundary Spanning Competence in Practice: Implications for Implementation and Use of Information Systems. *MIS Quarterly*, 29(2), 335–363.
- Lyytinen, K., Nickerson, J. V., & King, J. L. (2021). Metahuman systems= humans+ machines that learn. *Journal of Information Technology*, 36(4), 427–445.
- Majchrzak, A., More, P. H. B., & Faraj, S. (2012). Transcending knowledge differences in cross-functional teams. *Organization Science*, 23(4), 951–970.
- Mao, J.-Y., Vredenburg, K., Smith, P. W., & Carey, T. (2005). The state of user-centered design practice. *Communications of the ACM*, 48(3), 105–109.
- Markus, M. L., Majchrzak, A., & Gasser, L. (2002). A design theory for systems that support emergent knowledge processes. *MIS Quarterly*, 26(3), 179–212.
- Maruping, L. M., & Matook, S. (2020). The Multiplex Nature of the Customer Representative Role in Agile Information Systems Development. *MIS Quarterly*, 44(3), 1411–1437.
- Miles, M. B., & Huberman, A. M. (1994). *Qualitative Data Analysis: An Expanded Sourcebook*. Sage.
- Muldoon, J., Cant, C., Wu, B., & Graham, M. (2024). A typology of artificial intelligence data work. *Big Data & Society*, 11(1), 1–13.
- Murray, A., Rhymer, J., & Sirmon, D. G. (2021). Humans and technology: Forms of conjoined agency in organizations. *Academy of Management Review*, 46(3), 552–571.
- Naumann, J. D., & Jenkins, A. M. (1982). Prototyping: the new paradigm for systems development. *MIS Quarterly*, 6(3), 29–44.
- Newman, M., & Robey, D. (1992). A social process model of user-analyst relationships. *MIS Quarterly*, 16(2), 249–266.
- OECD. (2024). *Scoping the OECD AI principles*. OECD. <https://legalinstruments.oecd.org/en/instruments/OECD-LEGAL-0449>
- Patnayakuni, R., Rai, A., & Tiwana, A. (2007). Systems development process improvement: A knowledge integration perspective. 54(2), 286–300.
- Pawlowski, S. D., & Robey, D. (2004). Bridging user organizations: Knowledge brokering and the work of information technology professionals. *MIS Quarterly*, 28(4), 645–672.
- Pfeuffer, N., Baum, L., Stammer, W., Abdel-Karim, B. M., Schramowski, P., Bucher, A. M., Hügel, C., Rohde, G., Kersting, K., & Hinz, O. (2023). Explanatory interactive machine learning: establishing an action design research process for machine learning projects. *Business & Information Systems Engineering*, 65(6), 677–701.
- Raisch, S., & Krakowski, S. (2021). Artificial Intelligence and Management: The Automation-Augmentation Paradox. *Academy of Management Review*, 46(1), 192–210.
- Raisch, S., & Fomina, K. (2025). Combining human and artificial intelligence: Hybrid problem-solving in organizations. *Academy of Management Review*, 50(2), 441–464.
- Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review*, 33, 1–39.

- Rosenkranz, C., Stray, V., & Wiesche, M. (2025). Navigating the new frontier of information systems engineering. *Business & Information Systems Engineering*, 67(1), 1–5.
- Rus, I., Lindvall, M., & Sinha, S. (2002). Knowledge management in software engineering. *IEEE Software*, 19(3), 26–38.
- Russell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach*. Pearson.
- Saldaña, J. (2009). *The Qualitative Coding Manual*. Thousand Oaks.
- Schuetz, S., & Venkatesh, V. (2020). The rise of human machines: How cognitive computing systems challenge assumptions of user-system interaction. *Journal of the Association for Information Systems*, 21(2), 460–482.
- Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in machine learning systems. *Advances in Neural Information Processing Systems*, 28.
- Star, S. L. (2010). This is not a boundary object: Reflections on the origin of a concept. *Science, Technology, & Human Values*, 35(5), 601–617.
- Star, S. L., & Griesemer, J. R. (1989). Institutional Ecology, 'Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907–39. *Social Studies of Science*, 19(3), 387–420.
- Strauss, A., & Corbin, J. (1998). *Basics of qualitative research techniques*. Sage Publications.
- Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., & Müller, K.-R. (2021). Towards CRISP-ML (Q): a machine learning process model with quality assurance methodology. *Machine Learning and Knowledge Extraction*, 3(2), 392–413.
- Sundberg, L., & Holmström, J. (2024). Innovating by prompting: How to facilitate innovation in the age of generative AI. *Business Horizons*.
- Tim, Y., Pan, S. L., Ractham, P., & Kaewkitipong, L. (2017). Digitally enabled disaster response: the emergence of social media as boundary objects in a flooding disaster. *Information Systems Journal*, 27(2), 197–232.
- von Krogh, G. (2018). Artificial intelligence in organizations: New opportunities for phenomenon-based theorizing. *Academy of Management Discoveries*, 4(4), 404–409.
- Walz, D. B., Elam, J. J., & Curtis, B. (1993). Inside a software design team: knowledge acquisition, sharing, and integration. *Communications of the ACM*, 36(10), 63–77.
- Yin, R. K. (2009). *Case study research: Design and methods* (Vol. 5). Sage Publications.
- Zhang, Z., Hummel, J. T., Nandhakumar, J., & Waardenburg, L. (2020). Addressing the Key Challenges of Developing Machine Learning AI Systems for Knowledge-Intensive Work. *MIS Quarterly Executive*, 19(4), 221–238.

Paper II

Towards a Framework for Developing Artificial Intelligence Systems for Human-Artificial Intelligence Augmentation

Mei Wu-Gehbauer
University of Cologne
Christoph Rosenkranz
University of Cologne

In: Proceedings of the European Conference on Information Systems (ECIS), 2024

Towards a Framework for Developing Artificial Intelligence Systems for Human-Artificial Intelligence Augmentation

Abstract

With artificial intelligence (AI) systems increasingly augmenting users, it becomes crucial to comprehend how AI systems are developed for effective human-AI augmentation. Developing an AI system needs to consider its unique agentic, probabilistic, and self-learning properties that represent a paradigm shift from the rule-based programmed code, which is subject of traditional information systems development (ISD). We draw on a dual lens of agentic IS artifacts and cognitive fit to explore the human-AI interactions during development. Our findings suggest that the data science team plays a critical role in shaping human-AI augmentation by enacting the core alignment mechanisms “aligning the agentic IS artifact” and “aligning the user mental representation”. We found that human-AI augmentation is not pre-defined, but rather an emergent property of the triadic interaction between users, the data science team, and the AI system supported by the identified alignment mechanisms.

Keywords: AI systems development, agentic IS artifact, cognitive fit, alignment mechanisms

1 Introduction

With the increasing integration of artificial intelligence (AI) systems working alongside and augmenting users instead of replacing them (Jain et al., 2021), it becomes crucial to comprehend how AI systems are developed for effective human-AI augmentation. We refer to AI systems as state-of-the-art information systems (IS) with embedded AI capabilities that can perceive their environment, and react to changes in the environment based on predictions, recommendations, or decisions from an AI model to solve complex problems (Russell & Norvig, 2016). These systems are also called cognitive systems because they can augment humans in approaching cognitive tasks such as reasoning, decision-making, or creativity (Schuetz & Venkatesh, 2020). To realize human augmentation, AI systems leverage machine learning (ML) algorithms such as supervised, unsupervised, reinforcement learning, or deep learning (LeCun et al., 2015).

For an AI system to effectively augment humans in problem solving, it needs to be employed in a way that helps humans to complete their problem-solving task (Burton-Jones & Grange, 2013). This requires a close collaboration during development of the AI system between the data science team, the users, and the IS artifact within the business context. While traditional ISD approaches have placed an emphasis on designing information systems (IS) artifacts with consistent functionality to be used for generating outcomes or completing tasks (Benbasat & Zmud, 2003), AI systems are agentic IS artifacts with cognitive capabilities (Baird & Maruping, 2021). We argue that traditional ISD practices are not sufficient in the context of developing AI

systems because they are unique in mainly two ways.

First, AI systems rely on AI models trained on data enabling them to perform cognitive tasks that were previously the exclusive domain of humans (Russell & Norvig, 2016; Schuetz & Venkatesh, 2020). Their capacity to perceive, self-learn, and interact with their environment in a probabilistic way creates a level of uncertainty and ambiguity in user collaboration previously absent in information systems development. Second, with AI systems, a key actor gains in importance: the data science team, consisting of data scientists, machine learning (ML) engineers, and IT experts (Davenport, 2021; Vaast & Pinsonneault, 2021). The data science team takes decisions during building the AI model that impact how the model infers from data, and it facilitates the interaction between the users and the AI model during developing AI systems (Kim et al., 2017).

Given these unique characteristics of AI systems, current ISD literature calls for a revised view for developing AI systems, where the lines between development and use are blurred and the human-AI interaction becomes the focus of theorizing across development and use (Schuetz & Venkatesh, 2020; Waardenburg & Huysman, 2022). To date, research on developing AI systems has either focused on developing ever more powerful ML techniques (e.g., Vaswani et al., 2017; Yang et al., 2024) or designing adequate human-AI collaboration configurations when AI systems are deployed (e.g., Asatiani et al., 2021; Fügener et al., 2022; Gronlund & Aanestad, 2020). We find this problematic because it limits our understanding of how to develop AI systems for effective human-AI augmentation, the most promising path towards achieving superior problem-solving performance (Benbya et al., 2021; Teodorescu et al., 2021). Therefore, we put forward the following research question:

“How do activities by the data science team during the development of AI systems support effective human-AI augmentation?”

In the following sections, we conceptualize AI systems development (Section 2), elaborate on our research approach (Section 3), present our empirical findings (Section 4), derive a theoretical framework for developing AI systems for human-AI augmentation (Section 5), and conclude with limitations of the paper and an outlook (Section 6).

2 Theoretical Background

2.1 A socio-technical conceptualization of AI systems

We approach the concept of an AI system from a socio-technical perspective comprising the technology, actors, problem solving task, and structure (Leavitt, 2013). A socio-technical perspective is well-suited for analyzing the ISD process considering the interrelationships between the technical and social system (Lyytinen & Newman, 2008). These interrelations are inherently recursive in a way that users shape the technology and technology shapes IS use (Orlikowski,

2000).

From a technical perspective, the focus is on *agentic systems* that are non-deterministic and probabilistic in their actions (Baird & Maruping, 2021; Russell & Norvig, 2016). In contrast to conventional IS that support users in simple task completion, agentic IS or AI systems are applied to more ambiguous, cognitive, and open-ended problem solving tasks (Baird & Maruping, 2021). AI systems are also called cognitive systems reflecting their ability to mimic more human-like cognitive abilities such as reasoning, decision-making, and creativity by perceiving their environments (Schuetz & Venkatesh, 2020). To be able to achieve human-level cognitive abilities, AI systems possess an AI model comprising ML algorithms (Matsuo et al., 2022). The ML algorithms, with the data, are used to train a model, which results in a mathematically constructed representation of the environment. This model can change over time and it can be applied to a variety of multiple tasks and problem domains (Berente et al., 2021; Seidel et al., 2019). Such AI models are inherently inscrutable with limited visibility and transparency about the functioning of the models due to the probabilistic nature of the underlying algorithms (Berente et al., 2021). In effect, the technical system becomes an actor on its own.

From a social perspective, the data science team emerged as a key actor and plays a key role in developing AI systems (Davenport, 2021; Vaast & Pinsonneault, 2021). The data science team is a multi-disciplinary team consisting of data and ML experts such as data scientists, ML engineers, as well as IT experts, who are responsible for the end-to-end delivery of the AI system to the users (Amershi et al., 2019a; Laato et al., 2022; Patil, 2011). Data science teams apply agile and DevOps practices for the delivery of AI systems (Amershi et al., 2019a; Laato et al., 2022). While we acknowledge that all roles are required to deliver the AI system, we specifically focus on the data and ML experts including the roles of data scientist, ML engineer, data architect and data engineer, who possess the core knowledge of developing the AI model as well as user domain understanding (Davenport, 2021; Laato et al., 2022). We chose this focus, since the data and ML experts are instrumental in taking decisions that impact how the AI model is developed and how it infers from data (Amershi et al., 2019a; Sturm et al., 2021). At the same time, given the inscrutability of such AI models, there is a need for a continuous interaction between the data and ML experts and the users, emphasizing the critical role of the data and ML experts within the data science team, and setting it apart from a traditional ISD team (Davenport, 2021). The users of AI systems can be composed of the (end) users, who use and interact with the AI system, and of user representatives who are domain experts and not only have an intricate knowledge about the users' business processes but are also familiar with systems analysis techniques. Typically, in agile deliveries the role of the user representative is assumed by the product owner (PO).

Due to the cognitive capabilities of AI systems, the bi-directional nature of the relationship between humans and the IS artifact is now considered an important aspect in the context of AI systems (Baird & Maruping, 2021). As a result, the lines between the technical and social sys-

tems are increasingly blurred for AI systems in that the technical system increasingly assumes responsibilities that used to be the prime agency of humans, and the social system need to understand how to continuously interact and collaborate with the technical system in a way that leverages the strengths of both humans and AI (Waardenburg & Huysman, 2022).

2.2 Human-AI augmentation

In line with the socio-technical research focusing on the dyadic nature of the interrelationship between the technical and social system, we define human-AI augmentation as humans collaborating with AI to address problem solving (Raisch & Krakowski, 2021). Augmentation is not a new concept. It describes a situation where machines increase the capability of humans to approach problem situations (Engelbart, 1965). The fundamental assumption of human-AI augmentation is that humans and AI systems enhance one another through combining their complementary strengths, such that the resulting problem solving performance, as a whole, is superior (Jain et al., 2021). AI systems are particularly good at tasks that require fast processing of large amount of information, at recognizing complex patterns, or solving repetitive tasks more efficient and with higher quality (Dellermann et al., 2019; Jain et al., 2021). On the other hand, humans remain superior in formulating problems worth solving, are creative, have unique experiences, can comparatively easily adapt to changing environments and have common sense that is challenging to achieve with AI (Paul et al., 2022). While we recognize the interdependence of augmentation and automation, automation is usually applied to (part of) the business process or specific tasks in organizations rather than AI automating complete roles (Brynjolfsson et al., 2018; Chui et al., 2023a).

Human-AI augmentation becomes effective when the AI system is aligned with human context and needs, which in turn enables the users to employ the AI system in a way that helps to enhance their problem-solving task performance (Burton-Jones & Grange, 2013). We consider the effectiveness of human-AI augmentation as arising from the interactions between the data science team, the users, and the AI system while addressing a problem-solving task. Understanding this process depends on comprehending the interactions among the three actors. Effective human-AI augmentation may lead to redesigned job profiles or result in new practices allowing humans to take on more meaningful and valuable tasks, as AI can take care of routine work or work that was previously not possible without AI (Baer et al., 2022; van den Broek et al., 2021).

2.3 Developing AI systems for human-AI augmentation

Developing AI systems for human-AI augmentation is different in two ways (Amershi et al., 2019a; Wan et al., 2019): (1) The importance of the role of the data science team in developing AI systems, and (2) the agentic and probabilistic nature of the IS artifact, which requires a mutual alignment between the users and the AI system for effective human-AI augmentation.

First, developing AI systems is an intensely human task and the data science team plays a critical role in this (Sturm et al., 2021). The data science team makes decisions during building the AI model that impact how the model infers from data. These decisions are impacted by their understanding of the business context and their deep technical expertise and know-how developing AI systems (Seidel et al., 2019). On the other hand, the data science team mediates the interaction between the users and the AI model (Baird & Maruping, 2021). This mediation is required because model outcomes might not meet users' expectations. This might be the case when the data does not adequately represent the business problem or when assumptions are made in model training that are not in line with users' expectations.

Second, the alignment of the interaction between the model and the users to approach a problem is an important aspect of developing AI systems. Only focusing on technically optimizing the AI system without the involvement of the users using the system is not sufficient to achieve effective human-AI augmentation. Most AI systems will exist and operate as part of broader socio-technical systems in which humans play key roles (Selbst et al., 2019). For example, Lebovitz et al. (2021) found in a study of AI systems augmenting diagnostic radiologists that although the models showed high accuracy rates after training with inputs from domain experts, none of them met expectations when applied in practice. They are incapable of recognizing what is relevant to the users in a particular context on their own and cannot reason or act on events they are not trained on (Seidel et al., 2020). Even worse, state-of-the-art AI systems can even confidently state things as if they were facts that are entirely made up (OpenAI, 2023). This might render these systems ineffective, inaccurate, and even misguided without complementation by human actors (Amershi et al., 2019a; Selbst et al., 2019; van den Broek et al., 2021).

We argue that to achieve an effective human-AI augmentation, alignment mechanisms between the AI system and the users are required to better consider the user context, needs, and expertise. We consider alignment mechanisms to be an integral part of the ISD process for AI systems, and to represent the part of the human-AI augmentation process that allows us to gain a deeper understanding of how human-AI augmentation occurs (Mingers & Standing, 2017). Specifically, we consider alignment mechanisms from the perspective of the AI system (AI to the human) as well as from the perspective of the users (human to AI). From an AI perspective the focus is developing a high-performing AI model to approach the problem-solving task. From the user perspective, the focus is on the user perception of how the AI system supports the problem solution. Consequently, we apply a dual theoretical lens to address these two perspectives.

In this study, our conceptualization of AI systems draws on the theoretical framework for agentic IS artifacts by Baird and Maruping (2021). The framework provides our starting point to explore the development of AI systems because it acknowledges the agentic capabilities of the IS artifact in their ability to learn from data and form a representation of the business problem. To conceptualize how users collaborate with the agentic IS artifact to complete their problem-solving

task, we leverage cognitive fit as a theoretical lens (Vessey & Galletta, 1991). Specifically, this allows us to explore the user perspective because users draw on their mental models, developed from past experiences and expert knowledge from their domain, when making decisions to solve a problem jointly with the agentic IS artifact. Mental models in the context of AI systems are conceptualized as (1) understanding the functioning of the AI system, (2) being able to explain how and why certain outputs are created and being able to detect and recover from errors, and (3) understanding how to best interact with AI systems to augment human cognitive capabilities (Carroll & Olson, 1988; Orlikowski & Gash, 1994; von Krogh, 2018). When cognitive fit exists, users perceive the agentic IS artifact to support problem-solving. When cognitive fit does not exist (i.e., when there is a mismatch of the problem solution representation), alignment mechanisms may help solve this misfit.

Considering the agentic characteristics of the IS artifact as well as the mental representation of the users how they perceive the support from the agentic IS artifact makes this dual lens ideal for exploring alignment mechanisms for an effective human-AI augmentation during the AI system development process. We complement our framework with the role of the data science team for enacting such mechanisms reflecting the importance of this role. The data science team is instrumental in supporting the alignment mechanisms given their expert knowledge of AI system development and their domain understanding of the user context. Since the data science team is continuously involved during the reciprocal interactions between the AI system and the users, we use the lenses to explore the activities enacted by the data science team both towards the AI system and towards the users.

3 Research Method

3.1 Research setting and data collection

We employ a revelatory single case study design at a leading multi-national telecommunications provider (hereinafter referred to as TelcoGroup). We study five AI development projects that augment users in problem solving from the data science team perspective as well as from the user perspective and explore the interactions among them. This enables us to gain an in-depth understanding of the alignment mechanisms between the data science team, the users, and the AI system in the context of the ISD process. As AI systems are growing in importance, such a design is best suited to investigate this in an in-depth manner within the real-life context of an organization. We follow established guidelines for conducting exploratory case study research (Eisenhardt, 1989b; Yin, 2009).

TelcoGroup is among the top telecom operators in the world by revenue in 2023 and serves both private customers with broadband, mobile, and Internet-based media products and services as well as business customers with ICT solutions. Over the last years, TelcoGroup has invested

extensively in AI capabilities. According to analysts' assessments, TelcoGroup is one of the most advanced operators when it comes to AI adoption. It deploys data science teams that apply a wide range of different AI technologies and follow integrated agile software development and DevOps practices. TelcoGroup was chosen as a revelatory case organization because it presents a unique opportunity to investigate our research question, with access to teams and projects across different business and IT areas, and the possibility to collect longitudinal data. The main units of analysis are individuals and teams involved in developing AI systems in the case organization.

Interviews were the main form of gathering data, next to firm-level and project data sources, including presentations and documentations. The selection of interviewees followed a combination of literal (conditions of the cases lead to predicting the same results) and theoretical (conditions of the cases lead to predicting contrasting results) replication logics to analyze the different setups in which the interviewees operate (Dubé & Paré, 2003). We conducted 24 semi-structured interviews with data science team members and users from five different AI development projects in the timeframe between January and July 2023. To address the research question on how the data science team supports human-AI augmentation during development of the AI system, we complemented the data science perspective with the user perspective to explore the research question holistically, considering both the activities taken by the data science team and how user perception evolved. We used an interview protocol for the interviews but relied on probing questions as data collection progressed.

Drawing on the lens of the agentic artifact, we asked questions to the data science team such as: "How do the users interact with the AI system?", "How does the AI model learn?", "In how far do you or the users control the outcomes of the AI system?", "Can you describe in detail how you addressed the needs, wishes, and concerns of users when developing the AI system?", "How does the AI system affect the users' way of working?". From the user perspective, we draw on the cognitive fit lens and addressed questions such as: "How familiar are you with the AI system?", "In how far does the AI system benefit and support you in accomplishing your task?", "Can you explain how the AI outputs are generated?". All interviews were recorded, and transcriptions were generated to minimize the threat of inaccuracies and to allow for a line-by-line coding of data. Table 1 provides an overview of the interview data.

The AI projects cover AI use cases for anticipatory and prescriptive maintenance and optimization of network and equipment, predictive ticket routing in customer service, and personalization of customer recommendation and communication applying machine learning including deep learning. The focus of these projects is to deliver complete AI solutions to the users, and the data science team is responsible for the design, development, implementation, and deployment of these solutions. We constructed a detailed description of each AI project. For this, we asked the interviewees to provide us with project presentations and we had access to the company intranet where project documentation and demo presentation could be accessed, which formed the

Table 1: Interview data

Project	No of DS Team Members	Role of Key Informants DS Team	No of Users	Total No of Interviews	Interview Time in Minutes
Project A	4	Data scientists ML engineer Data engineer	3	7	363min
Project B	2	Data scientist ML engineer	4	6	307min
Project C	3	Data scientist ML engineer	1	4	213min
Project D	2	Data scientist ML engineer	1	3	170min
Project E	2	Data scientist ML engineer	2	4	200min
Total	13		11	24	1.253min

basis for triangulation. Table 2 provides a summary of the AI development projects in scope.

3.2 Coding and analysis

For data analysis, we used a form of grounded theorizing (Strauss & Corbin, 1998) drawing on data from our revelatory case study. We organized our analysis around the interactions among the data science team, the users, and the AI system to identify alignment mechanisms that support human-AI alignment. We focus on activities by the data science team as our core unit of analysis, as these activities form the core part of getting a deeper understanding of how they support human-AI augmentation.

As coding can be seen as a “cyclical act” (Saldaña, 2009), our coding process was highly iterative where initial concepts and relationships guided further theoretical sampling, data analysis, and the ongoing process of model development. The initial stage of analysis (open coding) aimed at identifying initial concepts (such as feature engineering or visualizing AI outputs) based on the data collected. We were interested in the interactions among the data science team, the users, and the AI system and, in particular, the supporting activities of the data science team. We generated open codes describing what the data science team did during the development process using a data reduction process often used by qualitative researchers aimed at focusing, simplifying, abstracting, and transforming the data that appear in written up transcriptions (Miles & Huberman, 1994). At the same time, we remained open to concepts that were salient in the data. We coded all interviews line by line producing open codes, which were further grouped and integrated, resulting in a set of first order concepts. For example, *user feedback collection and analysis* is a concept that originated from the concepts *user feedback on required data*, *user feedback on incorrect data input*, and *user feedback on data sources*. We triangulated the coding with additional materials such as project documentations and project demo presentations from

Table 2: Overview of AI system development projects

Project	Description	ML Algorithm	Effective Human-AI Augmentation
Project A Campaign personalization	Developing an AI system to assist the campaign and selection manager in development of personalized campaigns, selection of the target audience, and execution of campaigns.	Deep neural network	The AI system augments the user in enabling the campaign manager to focus on more value-adding tasks, and in enabling the selection managers to evolve in their role. Moreover, a new role of a data analyst / data scientist is required for processing of domain data as basis for the AI system.
Project B Predictive mobile network maintenance	Developing an AI system to assist radio field optimizer in identifying disruption of mobile services (e.g. call drops), prioritizing criticality and identifying of root causes.	Supervised non-linear machine learning	The AI system augments the user in enhancing the work of the radio field optimizer by providing a holistic view and by clustering the root causes instead of single case analysis only and enabling a better basis for decision and action.
Project C Predictive and prescriptive fixed-line network maintenance	Developing an AI system to assist network technicians in identifying problems in the fixed-line network early, in automatically generating tickets, and in proactively and automatically resolving less complex problems.	Supervised non-linear machine learning	The AI system augments the user in supporting the network technicians by predicting fixed-line network issues and improve proactivity in troubleshooting for less complex problems, thereby relieving the network technicians from growing workload and enabling them to focus on resolving more complex problems.
Project D Predictive and prescriptive home network optimization	Developing an AI system to assist network technicians for home network optimization and campaign managers for personalized customer communication and offering. The system identifies issues in the customer home network to proactively suggest actions.	Unsupervised machine learning	The AI system augments the users in supporting both the network technicians and the campaign managers in providing next best actions and / or offers to the customer to improve the customer's home network quality, thereby improving customer experience. In addition, the AI system enables an enhanced product steering for the commercial managers.
Project E Predictive and prescriptive customer service	Developing an AI system for predictive ticket routing of customer service requests and suggestion for troubleshooting actions either to internal customer service or field service agents.	Supervised, unsupervised machine learning, and deep neural network	The AI system augments users in routing the customer tickets including suggestion for actions to the appropriate agent team (internal service or field service) and enable the agents to focus on the work that require their specific expertise, thereby improving efficiency and customer experience.

the projects to increase validity.

In the process of axial coding, we further developed categories and systematically related these to the identified first order concepts from our open coding process. For example, the category *alignment with AI system functioning* was linked to *aligning with key attributes and feature importance*, *aligning via simulation*, and *aligning via training*, suggesting the interplay of socio-technical aspects to support user understanding of the AI system functions. This step was informed by the theoretical lenses to take into account existing knowledge from the literature. For instance, the cognitive fit lens suggests a need to align the user mental representation with the AI system and informed the dimensions of *aligning the user mental representation*. We used the lens to identify activities by the data science team to elaborate on the axial categories, thereby relating our analysis to our research question.

During selective coding, we centered around the core categories of *aligning the agentic IS artifact* and *aligning the user mental representation*. These concepts integrated the other categories into a coherent larger theoretical framework. The core category of *aligning the agentic IS artifact* synthesizes *model engineering* and *contextualization*, which are both directed towards the AI system, one aligning the AI model itself and the other aligning the AI system with the user context beyond the AI model. The second core category of *aligning the user mental representation* comprises both alignment of the user understanding of the AI system functioning and behavior with its unique characteristics as well as how the AI system aligns with business problem solving. We term these sub concepts as *aligning the mental representation of the AI system* and *aligning the mental representation of human-AI problem solving*. We found that the two core categories are interdependent on each other during AI system development, iteratively shaping the agentic IS artifact and the user mental representation.

Based on these steps and through iterative coding, discussion, and revision in the researcher team, we finally converged to a shared understanding and agreed-upon theoretical explanation for the observed phenomena. Figure 1 shows a visualization of our data coding scheme (based on Gioia et al., 2013).

We elaborate on the identified alignment mechanisms shown in Figure 1 in Section 4.1, while the identified relationships between the core categories are outlined in Section 4.2.

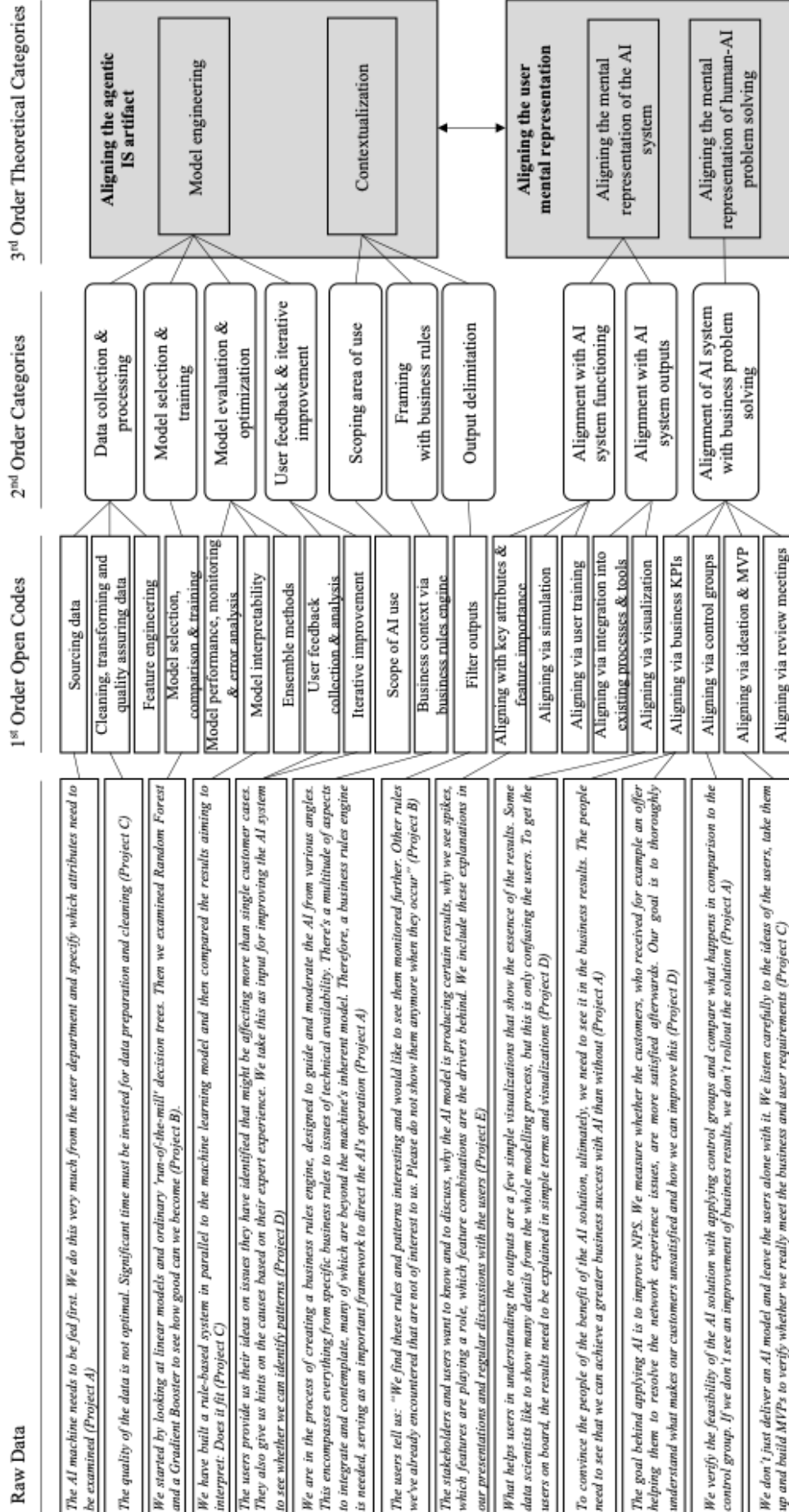


Figure 1: Excerpt of coding scheme

4 Findings

4.1 Alignment mechanisms

Our findings show that there is a close interaction between the data science team, the users, and the agentic IS artifact when developing AI systems, relying on two core mechanisms, which we describe as (1) *aligning the agentic IS artifact*, and (2) *aligning the user mental representation*. These mechanisms emerge as the data science team and the users are engaged in developing the agentic IS artifact to reach an effective human-AI augmentation. The interactions are cyclical and mutually interdependent. For example, the result from aligning the user mental representation might translate into input for aligning the IS artifact, such as in form of user feedback on additionally required data as basis for model engineering. This again adjusts the agentic IS artifact to support alignment of user mental representation. We first explain the mechanisms we found before elaborating further on the interdependencies among them. Example quotes are used to illustrate the mechanisms. All names of interviewees mentioned in this paper are pseudonyms to ensure the privacy of the participants.

Aligning the agentic IS artifact constitutes of the two major mechanisms *model engineering* and *contextualization*. Both are aligning the agentic IS artifact, with the distinction that model engineering is aimed at aligning the AI model in form of algorithms underlying the AI system, while contextualization involves aligning the AI system with the user context beyond the means of model engineering. We detail these mechanisms in the following.

Model engineering refers to the mechanism of developing, optimizing, maintaining, and iteratively improving the underlying model in form of algorithms, the very essence of the AI system, that enables the AI system to learn from data and perform cognitive functions such as decision making and problem solving. Model engineering ensures developing the AI system's capability that can augment the cognitive capabilities of humans. It includes data processing, model training, model evaluation optimization, as well as iterative improvement based on user feedback. Our findings on model engineering are in line with the existing literature and confirm the importance of model engineering for developing AI systems. However, we found that the mechanism of model engineering is often not being applied in isolation but in combination with the mechanism of contextualization, to ensure the AI system meets the requirements of the users, their business problems, and ultimately making an effective human-AI augmentation possible.

Contextualization refers to the mechanism of applying the AI system in a manner that is informed by and aligns with the specific context it is used in. This includes circumstances or nuances that are relevant to the business domain, which cannot be ensured by model engineering alone. We found three contextualization mechanisms, which we detail in the following with exemplary.

First, *scoping the area of use* comprises delineating the business parameters and conditions under which the AI system should be operated to support the users:

The user requirements on when it makes sense to apply AI are very specific. The AI system needs to identify a certain minimum number of customers that can be processed simultaneously applying the AI findings. An AI system identifying a target group below that threshold would not be sufficient to generate the required business revenues. (Erik, DS Team, Project D)

Second, *framing with business rules* is about framing the algorithmic logic with additional business-specific logic to cater to the specifics of the business domain:

We are in the process of creating a business rules engine, designed to guide and moderate the AI from various angles. This encompasses everything from specific business rules to issues of technical availability. There's a multitude of aspects to integrate and contemplate, many of which are beyond the machine's inherent model. Therefore, a business rules engine is needed, serving as an important framework to direct the AI's operation. (Michael, User, Project A)

While offering discounts to customers might be the optimal solution according to the machine's output, it may not be commercially sensible. Rather than relying solely on discounts, combining discounts with vouchers for products customers do not use yet may be the better solution. This is why a business rules engine is required on top of the AI solution. (Michael, User, Project A)

Third, *output delimitation* describes putting the output of the AI system in the context of the business domain. This includes delimiting outputs as illustrated by the following quote:

The AI model delivers scores across all customers. Based on scope and goal of the campaign, a cut has to be set for the customers in scope of the campaign. Where to set this cut has to be closely aligned with the campaign managers. (Sarah, User, Project A)

Aligning the user mental representation involves users assessing and verifying whether the representation by the AI system and how it is applied in the business context aligns with their mental representation and effectively augments their decision-making or problem-solving tasks. It ensures that the mental representation and problem-solving strategies of the human users are well-matched with the way the AI system represents and processes information. This mechanism is enacted when users start making sense of the outputs for problem solutions and developing human-AI augmentation strategies. We found two major alignment mechanisms: *aligning the mental representation of the AI system* and *aligning the mental representation of human-AI problem solving*.

Aligning the mental representation of the AI system is about users gaining an understanding of the AI system functioning and of the AI system outputs. These two mechanisms are described in the following.

First, *alignment with AI system functioning* concerns users gaining an understanding of the key characteristics of the probabilistic and self-learning nature of AI and understanding the key functions of the system. Users achieve this by comprehending the key attributes as well as the feature importance and comparing this with their own experiences and expertise. With simulations, the users review each run of the algorithm to stepwise understand the emergence of the outcomes and observe how features, weights and rules contribute to the final outputs. In addition, the users engage in trainings supported by the data scientists to acquire a foundational understanding of AI. The following quotes illustrate this mechanism:

Together with the business domain experts, we have very deep discussions on why the model delivers certain results, why we have spikes. We look together into which features, and feature combination play a role, and what the main drivers are. The domain experts examine single cases and retrace what has happened. (Amir, DS Team, Project E)

We did interactive coding sessions with the users where we took a call drop with all its attributes and performed simulations to evaluate the impact on the outputs. We somewhat said to the users, 'If we play around with the individual attributes and their values a bit, it's possible that a data point might fall under a different rule or be excluded from a rule.' We examined and found that some attributes tend to co-occur more strongly or have an anti-correlation. We then simply drew a few thousand samples and looked at how it's shaping up. Under which rules does the data point fall? How likely is it to fall in or out? It was quite interesting for the users to see that these aren't hard numbers. (Adam, DS Team, Project B)

We facilitate AI workshops with users to enhance their confidence and to equip them with a foundational understanding of the technology. (Elena, DS Team, Project C)

Second, through *alignment with AI system outputs* users gain understanding of the outputs of the AI system. It ensures that users comprehend the relevance of the AI outputs for their context so that the insights generated by the AI system are rendered actionable by the users. Users can be supported in achieving this by for example integration of the outputs of the AI system into existing processes and tools as well as intuitive visualization of outputs:

The users were not able to make sense of the AI outputs in relation to their own world. Then at some point we said we take the existing report they are using, put

in our results from the AI system. This was an enlightening moment for the users – suddenly, the findings from the AI made sense to them. (Adam, DS Team, Project B)

The AI solution identifies issues in the fixed-line network. The data scientist team developed a tool to visualize the findings in a very simple and intuitive way, like Google Maps. This made the results very tangible and accessible. After the presentation of the tool, several other units in the technology organization asked to use it. (David, User, Project C)

Aligning the mental representation of human-AI problem solving refers to the process of ensuring that the AI system is well-aligned with the specific problems and challenges that the users seek to address. This can be achieved by connecting the benefit of the AI system directly to business KPIs, making use of control groups to show the superiority of applying AI to solve the business problem, demonstrating the viability of applying AI to support the users with an MVP or continuously aligning the feasibility of applying AI in regular review meetings:

To create trust with the AI system, ultimately, we need to see the business impact. We need to show people the positive impact with concrete measurements. (Yuki, DS Team, Project C)

We verify the feasibility of the AI solution with applying control groups and compare what happens in comparison to the control group. (John, User, Project A)

We don't just deliver an AI model and leave the users alone with it. We listen carefully to the ideas of the users, take them up and build MVPs to verify whether we really meet the business and user requirements. (Elena, DS Team, Project C)

4.2 Interactions between alignment mechanisms

We found that an effective human-AI augmentation is a result of the intricate interplay between the two core alignment mechanisms *aligning the agentic IS artifact* and *aligning the user mental representation*. Together, these mechanisms constitute key processes for developing AI systems for human-AI augmentation that integrate with the users' activities in a collaborative way. The data science team enacts these mechanisms prompted by triggers as illustrated in the following project example:

In the predictive mobile network maintenance project (Project B), the objective was to proactively identify disruptions in the mobile network, identify root causes, and assess criticality. Initially, the data science team did extensive *model engineering* as part of *aligning the agentic*

IS artifact, sourcing the right data and trying out several ML algorithms to improve the model performance on this business problem:

We started by working with linear models and decision trees. Then we moved on to random forest and gradient boosting to see what the limit is. After that, we tweaked the attributes until we could predict certain problems with an accuracy of 90 percent. (Adam, DS Team, Project B)

The data science team converged finally on a supervised, non-linear ML algorithm. When they presented the AI findings to the users, the users initially did not grasp the probabilistic nature of AI. In addition, they found the features and the algorithmic logic of the AI model did not fully align with their understanding of the business problem and with their experience:

Our colleagues were always striving to find the perfect solution. Just a year or two ago, we couldn't accept that there are also solutions that are only 80 percent accurate and 20 percent incorrect. (Thomas, User, Project B)

The users were questioning why certain patterns have been identified and why certain features are important. They found certain patterns interesting and others less relevant. (Adam, DS Team, Project B)

Supported by the data science team, the users engaged in *aligning the user mental representation* both in terms of the AI system and the human-AI problem solving, such as participating in interactive coding sessions with simulations of how varying attribute values impact the outputs, retracing which features are driving the outputs or aligning how the use of the AI system influence the business KPIs. As a result, they gained understanding of the AI system enabling them to provide feedback to improve the system, changed their mental representation concerning AI and the interaction with the AI system as well as understanding the limitations of AI and started to engage in *aligning the agentic IS artifact*:

The users treated the results of the AI system with much less respect. They developed a 'healthy' skepticism that helped to identify areas for improvement in the system. They no longer accepted the outputs as face value, simply because they came from an algorithm. Instead, they began to discern, saying, 'This is nonsense again. But look, this could be something. Let's investigate further to see if it's an interesting pattern.' This new approach from the users, which I greatly appreciated, is key in collaboratively improving the AI system. (Adam, DS Team, Project B)

Now we are in the process of teaching the system what it should do. We also see the results of it. We had verification loops built into the system, where we can see what

has been worked on and its effect. Did one area improve in technical stability, or did nothing change, or did it possibly even get worse? We now have a comparison and can see that the results the AI system brings are not so bad. It's in our hands, and we can make it even better. (Thomas, User, Project B)

4.3 Facets of human-AI augmentation

In addition to the alignment mechanisms, we engaged with data science team members and the users across the five AI systems development projects to deepen our understanding of human-AI augmentation resulting from the ongoing alignment between the AI systems and its users. We found two facets that demonstrate effective human-AI augmentation: (1) within-role human-AI augmentation, and (2) role expanding human-AI augmentation.

Within-role human-AI augmentation refers to an augmentation where professionals are supported and enhanced by the AI system within their current role. For example, in the home network optimization project (Project D), the commercial managers, who are responsible for the sales of home network equipment, are now supported by the AI system in managing their salesforce based on the actual usage rather than only based on the amount of network equipment sold. This enables the commercial managers in more effectively achieving their business goals:

So far, commercial management has been guided by sold products but does not know whether and to what extent these products are used in the home network. The AI system enables exactly this and can, for example, show which products are rarely used after the sale is completed. This provides a completely new way of steering. (Erik, DS Team, Project D)

Role expanding human-AI augmentation refers to an augmentation where professionals are enhanced by the AI system in a way that transforms their current role. For example, in the campaign personalization project (Project A) the AI system takes over the identification and refinement of selection criteria, enabling the selection manager to transition into the campaign management role. This role evolution is illustrated by the following quote:

As selection manager, you still must manage the data part, but the task is different. You must make sure that the right data is there, and the AI system is running. You are no longer the one who clicks together the selection criteria. This leads to the role transforming more towards that of a campaign manager, allowing for a much deeper engagement with the content of the offering. (Michael, User, Project A)

5 Discussion and Contribution

Our study sets out to explore how data science teams develop AI systems for effective human-AI augmentation. The findings from our study highlight that developing AI systems for human-AI augmentation involves interactions between the users, the data science team, and the AI system relying on alignment mechanisms enacted by the data science team. The results underscore that optimizing the ML model for best performance is not sufficient for effective human-AI augmentation. Instead, the data science team plays an important role in mediating the interactions between the users and the AI system to align the agentic IS system with the user mental representation. Aligning the agentic IS artifact goes beyond model engineering aiming for a technically sound AI system to include contextualization. Contextualization, in turn, is enacted by the data science team to provide the AI systems with a nuanced understanding of the specific environments in which they operate. Since all AI systems in scope of the study support problem solving in specific business operational contexts, contextualization is a critical mechanism because it ensures that the AI system is not only technically feasible and resilient but also aligned to the complex and dynamic business context it is applied to. Aligning the user mental representation, on the other hand, is a critical mechanism for ensuring that the AI system is not only technically sound and adequately applied in the business context, but that this is aligned with the user mental representation in terms of the AI system being plausible, relevant, and useful to the users for approaching the business problem. This alignment is crucial so that the AI system is supporting and amplifying human capabilities, fostering a symbiotic relationship where both the AI system and humans work towards common objectives with enhanced problem-solving capabilities.

These alignment mechanisms constitute a highly iterative process with mutually reinforcing mechanisms. For instance, in the mobile network maintenance project, once the data science team was able to support the users in aligning with the AI system, the users started to actively engage in providing inputs for aligning the AI system with their needs and business context. This further advanced the users in aligning their mental representation of human-AI problem solving. As a result, during development of the AI systems, alignment mechanisms are enacted by the data science team with which users and the agentic IS artifact achieved an effective human-AI augmentation. Our findings are summarized in the theoretical framework in Figure 2, which is further discussed in the next section. The theoretical and practical contributions are outlined subsequently.

5.1 Framework for developing AI systems for human-AI augmentation

We draw on the dual lens of the framework for agentic IS artifacts and cognitive fit theory to explore the alignment mechanisms between the agentic IS artifact and the users (Baird & Maruping, 2021; Vessey & Galletta, 1991). Based on this dual lens and building on the findings of this

case, we offer a theoretical framework for how to develop AI systems for human-AI augmentation. Figure 2 shows that while approaching a problem-solving task, alignment mechanisms are enacted by the data science team between the agentic IS artifact and the user mental representations. Our framework explains that an effective human-AI augmentation is an outcome of an interaction of the users with the agentic IS artifact supported by alignment mechanisms enacted by the data science team. We conceptualize these mechanisms as aligning the agentic IS artifact and aligning the user mental representation.

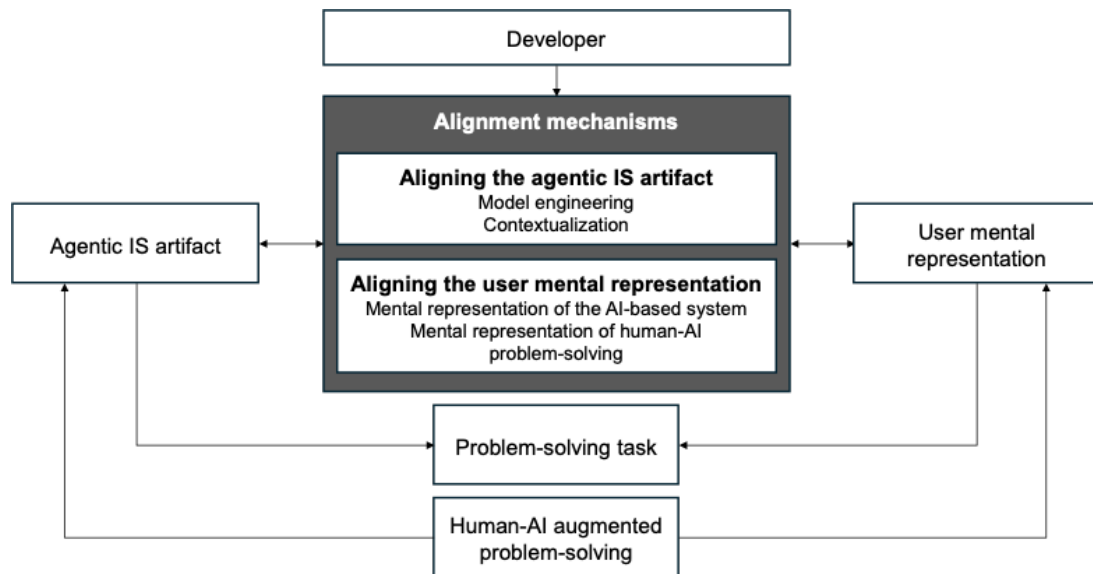


Figure 2: Framework for developing AI systems for human-AI augmentation (integrated and extended based on Baird and Maruping (2021) and Vessey and Galletta (1991)).

The theoretical framework highlights that, through the enactment of alignment mechanisms by the data science team, the agentic IS artifact improves its representation of the business problem, while the users reflect on their mental representation of the problem solution by applying the agentic IS artifact. In the AI projects we found that the data science team continuously enacted such alignment mechanisms. The iterative enactment of alignment mechanisms led to convergence on a complementary or synergistic human-AI augmentation. Our findings suggest, that rather than laying out a human-AI augmentation setup from the start, the human-AI augmentation emerges out of a recurrent process of enacting alignment mechanisms by the data science team during the development process of the AI system. Therefore, human-AI augmentation is an emergent property of developing AI systems resulting from the recurrent enactment of alignment mechanisms.

5.2 Contributions

Our study offers several contributions to the IS literature. First, our findings suggest that developing AI systems for human-AI augmentation involves a triadic interaction between users, the

data science team, and the AI system relying on alignment mechanisms enacted by the data science team. This goes beyond the long-standing tradition in ISD literature focusing on the dyadic interaction between user and the developer team during development of IS systems. With AI, the agentic IS artifact becomes an actor on its own, forms an own representation of the problem-solving space using the underlying AI model, self-learns and acts in a non-deterministic way. This requires mutual alignment of the AI system with the user mental representation. Second, the study highlights the critical role of the data science team in mediating the alignment between the users and the AI model, while enacting the core mechanisms of aligning the agentic IS artifact and aligning the user mental representation. By adding to the understanding of the role and activities of the data science team, we contribute to the framework of agentic IS artifacts and follow the call from Baird and Maruping (2021) for including the developer view. Third, we add to the research on human-AI augmentation by demonstrating that human-AI augmentation is not a predefined state but rather an emergent property of the alignment and interaction of the users with the agentic IS artifact.

Our findings offer several practical implications for leaders and data science teams spearheading AI systems development in organizations. We show that developing AI systems for human-AI augmentation involves an ongoing, intricate interplay between the data science team, the AI system and the users. The critical mechanisms identified to align both the AI system and the user mental representation can serve as a guide for the data science teams to focus their activities and direct their efforts. When aligning the AI system, the data science teams need to deeply reflect on the specifics, rules and boundaries of the business problem to purposefully contextualize the developed AI model with the business context in order to tailor the AI system to the specific business problem solving. In parallel, users need to be empowered to align with the AI system. The data science team plays a crucial role to enable the users in gaining a comprehensive understanding of the AI system functioning and outputs. This again provides the ground for users to be able to engage in providing input for aligning the AI system towards the joint human-AI problem solving, ultimately evolving to an effective human-AI augmentation and superior problem solving performance.

6 Limitations and Outlook

Several limitations to our study need to be considered. First, while a single case study design provides great depth and focus, it also provides less abstraction from the case organization compared to a multiple case study design. We believe this limitation to be necessary to be aware of but nonetheless to also be of only marginal nature, as this study is evaluating different AI development projects from different business areas. Second, our focus is on AI systems augmenting humans in problem-solving instead of automation systems replacing humans. However, also for autonomous AI systems, there is human input required to provide context. Therefore, findings

from this study may also be of relevance for automated systems. Since our focus of this study is on AI models that need to be trained by the data science teams, a fruitful way forward for future research is to enhance our understanding on human-AI alignment during developing AI systems by extending our study with projects developing AI systems based on pre-trained AI models for human-AI augmentation.

References

- Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmermann, T. (2019a). Software engineering for machine learning: A case study. *IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 291–300.
- Asatiani, A., Malo, P., Nagbol, P. R., Penttinen, E., Rinta-Kahila, T., & Salovaara, A. (2021). Sociotechnical Envelopment of Artificial Intelligence: An Approach to Organizational Deployment of Inscrutable Artificial Intelligence Systems. *Journal of the Association for Information Systems*, 22(2), 325–352.
- Baer, I., Waardenburg, L., & Huysman, M. (2022). What Are We Augmenting? A Multidisciplinary Analysis of AI-based Augmentation for the Future of Work. *International Conference on Information Systems (ICIS) Proceedings*.
- Baird, A., & Maruping, L. M. (2021). The Next Generation of Research on IS Use: A Theoretical Framework of Delegation to and from Agentic IS Artifacts. *MIS Quarterly*, 45(1), 315–341.
- Benbasat, I., & Zmud, R. W. (2003). The identity crisis within the IS discipline: Defining and communicating the discipline's core properties. *MIS Quarterly*, 27(2), 183–194.
- Benbya, H., Pachidi, S., & Jarvenpaa, S. (2021). Special issue editorial: artificial intelligence in organizations: implications for information systems research. *Journal of the Association for Information Systems*, 22(2), 281–303.
- Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing Artificial Intelligence. *MIS Quarterly*, 45(3), 1433–1450.
- Brynjolfsson, E., Mitchell, T., & Rock, D. (2018). What can machines learn, and what does it mean for occupations and the economy? *AEA Papers and Proceedings*, 108, 43–47.
- Burton-Jones, A., & Grange, C. (2013). From use to effective use: A representation theory perspective. *Information Systems Research*, 24(3), 632–658.
- Carroll, J. M., & Olson, J. R. (1988). Mental Models in Human-Computer Interaction. *Handbook of Human-Computer Interaction*, 45–65.
- Chui, M., Hazan, E., Roberts, R., Singla, A., Smaje, K., Sukharevsky, A., Yee, L., & Zimmel, R. (2023a). The economic potential of generative AI: The next productivity frontier. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier>
- Davenport, T. H. (2021). The analytics team. In *Informatics body of knowledge* (pp. 49–76). Wiley.
- Dellermann, D., Ebel, P., Söllner, M., & Leimeister, J. M. (2019). Hybrid intelligence. *Business & Information Systems Engineering*, 61(5), 637–643.
- Dubé, L., & Paré, G. (2003). Rigor in information systems positivist case research: current practices, trends, and recommendations. *MIS Quarterly*, 27(4), 597–636.

- Eisenhardt, K. M. (1989b). Building theories from case study research. *Academy of Management Review*, 14(4), 532–550.
- Engelbart, D. C. (1965). *Augmenting human intellect: Experiments, concepts, and possibilities*. Stanford Research Institute Menlo Park.
- Fügener, A., Grahl, J., Gupta, A., & Ketter, W. (2022). Cognitive challenges in human–artificial intelligence collaboration: Investigating the path toward productive delegation. *Information Systems Research*, 33(2), 678–696.
- Gioia, D. A., Corley, K. G., & Hamilton, A. L. (2013). Seeking qualitative rigor in inductive research: Notes on the Gioia methodology. *Organizational Research Methods*, 16(1), 15–31.
- Gronlund, T., & Aanestad, M. (2020). Augmenting the algorithm: Emerging human-in-the-loop work configurations. *Journal of Strategic Information Systems*, 29(2), 1–16.
- Jain, H., Padmanabhan, B., Pavlou, P. A., & Raghu, T. S. (2021). Editorial for the special section on humans, algorithms, and augmented intelligence: The future of work, organizations, and society. *Information Systems Research*, 32(3), 675–687.
- Kim, M., Zimmermann, T., DeLine, R., & Begel, A. (2017). Data scientists in software teams: State of the art and challenges. *IEEE Transactions on Software Engineering*, 44(11), 1024–1038.
- Laato, S., Mäntymäki, M., Minkinen, M., Birkstedt, T., Islam, A. K. M., & Dennehy, D. (2022). Integrating machine learning with software development lifecycles: Insights from experts. *European Conference on Information Systems (ECIS) Proceedings*.
- Leavitt, H. J. (2013). Applied organizational change in industry: Structural, technological and humanistic approaches. In *Handbook of organizations* (pp. 1144–1170). Routledge.
- Lebovitz, S., Levina, N., & Lifshitz-Assaf, H. (2021). Is AI Ground Truth Really ‘True’? The Dangers of Training and Evaluating AI Tools Based on Experts’ Know-What. *MIS Quarterly*, 45(3), 1501–1525.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.
- Lyytinen, K., & Newman, M. (2008). Explaining information systems change: a punctuated socio-technical change model. *European Journal of Information Systems*, 17, 589–613.
- Matsuo, Y., LeCun, Y., Sahani, M., Precup, D., Silver, D., Sugiyama, M., Uchibe, E., & Morimoto, J. (2022). Deep learning, reinforcement learning, and world models. *Neural Networks*, 152, 267–275.
- Miles, M. B., & Huberman, A. M. (1994). *Qualitative Data Analysis: An Expanded Sourcebook*. Sage.
- Mingers, J., & Standing, C. (2017). Why things happen—Developing the critical realist view of causal mechanisms. *Information and Organization*, 27(3), 171–189.
- OpenAI. (2023). *GPT-4 technical report*. arXiv 2303.08774 (tech. rep.). OpenAI.
- Orlikowski, W. J. (2000). Using technology and constituting structures: A practice lens for studying technology in organizations. *Organization Science*, 11(4), 404–428.

- Orlikowski, W. J., & Gash, D. C. (1994). Technological frames: making sense of information technology in organizations. *ACM Transactions on Information Systems (TOIS)*, 12(2), 174–207.
- Patil, D. J. (2011). *Building Data Science Teams*. O'Reilly Media, Inc.
- Paul, S., Yuan, L., Jain, H. K., Robert Jr, L. P., Spohrer, J., & Lifshitz-Assaf, H. (2022). Intelligence augmentation: Human factors in AI and future of work. *AIS Transactions on Human-Computer Interaction*, 14(3), 426–445.
- Raisch, S., & Krakowski, S. (2021). Artificial Intelligence and Management: The Automation-Augmentation Paradox. *Academy of Management Review*, 46(1), 192–210.
- Russell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach*. Pearson.
- Saldaña, J. (2009). *The Qualitative Coding Manual*. Thousand Oaks.
- Schuetz, S., & Venkatesh, V. (2020). The rise of human machines: How cognitive computing systems challenge assumptions of user-system interaction. *Journal of the Association for Information Systems*, 21(2), 460–482.
- Seidel, S., Berente, N., Lindberg, A., Lyytinen, K., Martinez, B., & Nickerson, J. V. (2020). Artificial intelligence and video game creation: A framework for the new logic of autonomous design. *Journal of Digital Social Research*, 2(3), 126–157.
- Seidel, S., Berente, N., Lindberg, A., Lyytinen, K., & Nickerson, J. V. (2019). Autonomous tools and design: a triple-loop approach to human-machine learning. *Communications of the ACM*, 62(1), 50–57.
- Selbst, A. D., Boyd, D., Friedler, S. A., Venkatasubramanian, S., & Vertesi, J. (2019). Fairness and abstraction in sociotechnical systems. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 59–68.
- Strauss, A., & Corbin, J. (1998). *Basics of qualitative research techniques*. Sage Publications.
- Sturm, T., Gerlach, J. P., Pumplun, L., Mesbah, N., Peters, F., Tauchert, C., Nan, N., & Buxmann, P. (2021). Coordinating Human and Machine Learning for Effective Organizational Learning. *MIS Quarterly*, 45(3), 1581–1602.
- Teodorescu, M. H. M., Morse, L., Awwad, Y., & Kane, G. C. (2021). Failures of Fairness in Automation Require a Deeper Understanding of Human-ML Augmentation. *MIS Quarterly*, 45(3), 1483–1499.
- Vaast, E., & Pinsonneault, A. (2021). When Digital Technologies Enable and Threaten Occupational Identity: The Delicate Balancing Act of Data Scientists. *MIS Quarterly*, 45(3), 1087–1112.
- van den Broek, E., Sergeeva, A., & Huysman, M. (2021). When The Machine Meets The Expert: An Ethnography of Developing AI for Hiring. *MIS Quarterly*, 45(3), 1557–1580.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

- Vessey, I., & Galletta, D. (1991). Cognitive fit: An empirical study of information acquisition. *Information Systems Research*, 2(1), 63–84.
- von Krogh, G. (2018). Artificial intelligence in organizations: New opportunities for phenomenon-based theorizing. *Academy of Management Discoveries*, 4(4), 404–409.
- Waardenburg, L., & Huysman, M. (2022). From coexistence to co-creation: Blurring boundaries in the age of AI. *Information and Organization*, 32(4), 1–11.
- Wan, Z., Xia, X., Lo, D., & Murphy, G. C. (2019). How does machine learning change software development practices? *IEEE Transactions on Software Engineering*, 47(9), 1857–1871.
- Yang, J., Jin, H., Tang, R., Han, X., Feng, Q., Jiang, H., Zhong, S., Yin, B., & Hu, X. (2024). Harnessing the power of LLMs in practice: A survey on ChatGPT and beyond. *ACM Transactions on Knowledge Discovery from Data*, 18(6), 1–32.
- Yin, R. K. (2009). *Case study research: Design and methods* (Vol. 5). Sage Publications.

Paper III

Between Enablement and Control – Generative Artificial Intelligence-based Systems Development

Mei Wu-Gehbauer
University of Cologne

Christoph Rosenkranz
University of Cologne

In: Proceedings of the Hawaii International Conference on System Sciences (HICSS), 2025

Between Enablement and Control – Generative Artificial Intelligence-based Systems Development

Abstract

The proliferation of generative artificial intelligence (GenAI)-based systems with embedded LLM has raised the interest of organizations in leveraging their potential to enhance work performance. However, for GenAI systems, the focus of systems development should extend beyond the artifact to managing the human-AI interaction. Through an exploratory case study of three GenAI systems development projects in a large enterprise we find that GenAI systems are shaped through an interplay of enablement and control of the user-AI interaction. We explain how these mechanisms practically unfold for systems supporting broad versus narrow user intents. Our findings demonstrate that through enablement the divergent nature of GenAI systems can be strengthened, while control ensures their convergence with requirements in the organizational context. With this we contribute to the understanding of how to develop GenAI systems that support the work in organizations.

Keywords: GenAI systems development, user-AI interaction, human-AI collaboration, case study

1 Introduction

Information systems (IS) are increasingly embedded with *artificial intelligence* (AI) to execute and process tasks in organizations (Russell & Norvig, 2016). Until recently, *discriminative AI* – such as predictive models, recommender systems, and anomaly detection – has primarily been used to support organizational goals (Berg et al., 2023). These systems learn to predict labels based on features by modeling the probability of a label (Foster, 2022). Increasingly, however, *generative AI* (GenAI) with embedded large language models (LLMs) with examples such as GPT-4 or GitHub Copilot has raised organizations’ interest in leveraging the potential of LLMs to enhance work performance. In contrast to discriminative AI, GenAI models aim to generate new data samples (e.g., creating text, images, code) by modeling the probability distribution of the underlying data (Foster, 2022). Given their broad applicability for various domains, GenAI is expected to become a widespread component of many IS (Cusumano, 2023).

GenAI shares common characteristics with discriminative AI, such as self-learning capacity, their probabilistic nature resulting in inscrutability, and the ability to act autonomously (Berente et al., 2021). In our view, two aspects distinguish GenAI systems from discriminative AI systems. First, while discriminative AI is trained on domain-specific data for specific tasks, GenAI foundation models are pre-trained, general-purpose models that offer various application scenarios across the organization. Second, while discriminative AI outputs are the result of inferences

from pre-defined data sources, GenAI depends on user or system inputs conveying user intents to create outputs (Brown et al., 2020).

These characteristics pose unique challenges for organizations that embark on embedding GenAI into their IS. Since GenAI foundation models are not trained on the unique business context and data, by default, they do not have a nuanced understanding of the business logic and routines to which they are applied (Benbya et al., 2024). Moreover, users with their own facets of business understanding influence the outputs of GenAI through inputs in the user-AI interaction. This adds to the possibility of deviating from the intended purpose that the GenAI system is supposed to support. Recovering from unintended negative outcomes can be costly for organizations and may even result in adverse effects on a company's reputation. For example, in 2024, a major airline was forced to pay damages to a customer after its GenAI-powered assistant provided information to a customer inconsistent with airline policy (Garcia, 2024). Facing the challenges GenAI poses, a recent study reveals that, despite the widespread adoption of GenAI, only 11 percent of companies apply GenAI systems at scale for real-world business scenarios (Baig et al., 2024).

Therefore, we argue it is crucial to understand how to overcome the unique challenges in GenAI development to effectively support organizational work. Since GenAI is designed to generate outputs based on inputs reflecting user intent, we suggest that managing the *user-AI interaction* becomes the focal point for developing GenAI systems.

The existing body of research on AI predominantly investigates human-AI collaboration and its impact on humans during use of AI-based systems including studies on agency shifting from humans to AI (e.g., Baird & Maruping, 2021; Fügener et al., 2022), on humans collaborating with AI in decision making (e.g., Gronlund & Aanestad, 2020; Lebovitz et al., 2021; van den Broek et al., 2021), or on the impact of user interaction on user perception and behavior while applying AI-enabled systems (e.g., Rzepka & Berger, 2018). Our study aims to extend this body of knowledge by providing an understanding of how to design and develop GenAI systems with the user-AI interaction in focus.

The research question we ask is: *“How are GenAI-based systems developed in the interaction of developers, users, and the GenAI artifact to support tasks in an organizational context?”*

We address this question in an exploratory study of three development projects for GenAI systems in the context of a large corporation as a case site. Applying a socio-technical perspective, we explore how these GenAI systems are developed and how the challenges described are addressed during design and development (Lyytinen & Newman, 2008).

The remainder of the paper is structured as follows. First, we conceptualize GenAI systems in organizations from a socio-technical perspective, reflect on GenAI systems development, and outline the relevance of the user-AI interaction and user intent. Second, we elaborate on our research design and data analysis approach. Third, we present the empirical findings of our

study. Finally, we discuss the findings, contributions, and limitations of our study, and give an outlook on further research.

2 Related Work

2.1 GenAI systems in organizations

We conceptualize GenAI systems in organizations from a socio-technical perspective comprising the technical LLM capabilities while being embedded in the social and organizational context to support users in addressing organizational tasks (Baird & Maruping, 2021).

From a technical perspective, as an agentic IS artifact, we focus on LLMs as a subset of GenAI. LLMs are foundation models for text and are pre-trained on large corps of web-scale text data. LLMs are designed to react to inputs initiated by users or systems (Brown et al., 2020). Recent LLMs are based on the transformer architecture, a deep learning approach that learns patterns in data and generates new outputs (Vaswani et al., 2017). Transformer-based LLMs are the basis of conversational agents, with ChatGPT being the most prominent one.

From a social perspective, we focus on the developer and user perspectives and the actions taken by developers towards the user-AI interaction. As developers, we refer to all experts who deliver GenAI systems including experts making use of software engineering and machine learning techniques to implement GenAI systems and those experts who plan and coordinate within the GenAI projects to ensure end-to-end delivery. With users we refer to end users who interact with the GenAI system to solve a specific task within the organization as well as to power users who are domain experts and knowledgeable about the business domain and processes and are usually familiar with systems analysis techniques. As such, power users are usually involved in the design and testing of the GenAI systems.

Such agentic IS cannot be fully understood without the *context* in which they operate (Rahwan et al., 2019). Context has been defined as the organizational setting in which phenomena occur that may serve as a tension system comprising opportunities and constraints (Johns, 2006). Drawing on this, we propose that context can either enhance or limit the effect of GenAI systems on task performance. Context can be highly varied ranging from human-related (e.g., expert knowledge, individual preferences), organizational-related (e.g., established rules and practices), to domain-related (e.g., embedded business logic, architecture standards). The usefulness of GenAI-generated outputs largely depends on how well the GenAI system considers the organizational context.

2.2 GenAI systems development

Traditional IT systems are rule-based systems with explicitly encoded rules and procedures. Developers encode knowledge from expert users and represent this explicitly through rules and other system representations. IT systems apply those rules to support users in their tasks, forming an information system as a socio-technical system (e.g., Bostrom et al., 2009). Accordingly, traditional information systems development (ISD) focuses on the dyadic relationship between developers and users (e.g., Chen et al., 2011) and on managing the user and the user requirements (e.g., Keil et al., 2013).

With the advent of AI, the established focus in ISD had to be revisited. In particular, the AI-based agentic IS artifact emerges as an additional, intelligent actor with agency (Baird & Maruping, 2021). The AI-based system forms an algorithmic representation of the problem-solving space using the AI model, self-learns, and acts in a non-deterministic way (Berente et al., 2021). For the development of AI-based systems, the focus shifts from the traditional dyadic view to the triadic relationship between developers, users, and the AI artifact, requiring developers to manage AI in addition to user needs by supporting the mutual alignment between the AI system and the user mental representation (Wu-Gehbauer & Rosenkranz, 2024).

Likewise, developing GenAI systems with foundation models, unfolds in the triadic relationship between developers, users and the GenAI artifact. However, GenAI systems are designed to take in input conveying user intents to generate new content. This implies that developing GenAI systems is at its core a matter of designing the user-AI interaction.

2.3 User-AI interaction in GenAI

In our study we focus on the user-AI interaction, which we define as an emergent process of reciprocal interactions and continual adjustments between users and GenAI that can result in improved task performance (Benbya & McKelvey, 2006). The user-AI interaction consists of mainly two parts: (1) understanding the user intent, as conveyed through user or system inputs; and (2) generating responses based on the user intent that are aligned to the user and organizational context.

User intent is defined as purpose for action (Kasthuriarachchy et al., 2020). In an organization, this typically refers to an action to perform a task that entails business logic, such as a business process, workflow, or organizational routine. For example, when designing a GenAI-based chatbot, answers must follow certain business rules; GenAI-assisted coding involves adhering to certain coding standards or specific API definitions; or internal chatbots generating responses to organizational processes must follow standardized workflows. User intents can be provided to GenAI through user or system input. In case of system input, the design of this input is influenced by user intents, which the GenAI system is supposed to fulfil. Both when user input

and system input is provided, the goal of the GenAI system is to fulfil a user intent that aligns with task objectives.

LLMs have demonstrated enhanced capabilities for text understanding. Understanding the user intent expressed in text and putting it into the user context ensures relevance and adequacy of the outputs (Vaswani et al., 2017). However, identifying user intents and adequately addressing them while adhering to required business logic can be challenging. Intents expressed by users can be ambiguous, relying on implicit domain-specific knowledge, or intuitive assumptions that the model was not trained on (He et al., 2024). Furthermore, generating responses aligned to organizational context depends on contextual information from the involved business process or business logic. This means that user-AI interaction needs to be managed to consider both the way users interact with GenAI and the required business logic. If done insufficiently, GenAI that is prone to hallucinations, might provide responses deviating from the user intent and business logic (Ji et al., 2023).

We differentiate user intents in an organizational context between narrow and broad user intent (Fernández-Martínez et al., 2022). *Broad intent* encompasses GenAI to address a broad variety of user goals. On the one hand, this requires users to express their intent for GenAI to understand and identify the user intent. On the other hand, GenAI needs a comprehensive understanding of the organizational environment and business logic across the broad range of different intents. Contrary to this, *narrow intent* is when users have a specific, well-defined organizational task that they want to accomplish. Here, GenAI can address the specified task without the need to interpret the user intent. For example, in software development, GenAI assists with various tasks in broad user intent, such as code generation, bug fixing or testing. Supporting a narrow intent is a GenAI system targeting to fix security vulnerabilities in the code. The differentiation between broad and narrow intent is important because it determines the business scenario to be supported by GenAI as well as the form of human-AI interaction in GenAI systems.

3 Research Method

3.1 Research Design and Setting

To address the research question, we conduct an exploratory case study and follow an inductive approach (Sarker et al., 2018). Case studies allow for developing an in-depth understanding of information systems in a socio-technical context (Yin, 2009). We study the phenomenon of interest across three GenAI project cases serving different types of user intents and explore how the GenAI systems are developed in the interaction between developers, users and the GenAI artifact. Specifically, we aim to generate understanding on how to overcome the challenges of leveraging GenAI as general-purpose technology and how to manage the dynamics in the user-AI interaction for the resulting GenAI systems to be effective in supporting tasks in an

organizational context.

We selected a leading multi-national telecommunications enterprise as our case site. We consider the selected case site to be revelatory. According to analysts' assessments, the case company is one of the most advanced telco operators when it comes to AI adoption. Beyond having built up extensive data and AI platforms and skills, AI projects across business areas are developing AI-based systems to turn the potential of AI including GenAI into business value. Regarding GenAI, pre-trained LLMs form the basis for the GenAI systems development. For the study, we had access to the early stage GenAI systems projects, among them a pioneering project that started beginning of 2022.

3.2 Data collection and analysis

Our main methods for data collection are interviews and project documentation as well as video and audio material. We selected the projects based on two criteria: (1) The selected projects include projects both for broad and narrow user intent; and (2) the GenAI systems developed in the projects are productively launched or shortly before launch following a phase of pilot testing.

To gain a thorough understanding of how GenAI systems are developed to support organizational objectives, we interviewed twelve informants from three GenAI systems development projects in spring 2024. As informants we focused on two groups of human actors per project: the developer and the user. To address our research question, it is essential to include both perspectives. Consequently, we sampled interviewee pairs of developers and users for every GenAI project.

Each interview lasted between 45 and 65 minutes and took place either onsite at the case company or via online video conferencing. We followed a semi-structured approach to conduct the interviews, using an interview protocol and at the same time leaving space for probing based on the informants' responses. In the interviews, we aimed to understand actions taken to develop the GenAI systems from the perspectives of developers and users.

Table 1: Data sources

<i>Intent</i>		<i>Broad</i>	<i>Narrow</i>		
Project	Type	A	B	C	Total
Interviews Developer	#	4	2	1	12 (655 min)
	time	210 min	125 min	55 min	
Interviews User	#	2	2	1	5 (110 min)
	time	110 min	110 min	45 min	
Audio & Video	#	2	3	-	5
Documents	#	6	3	3	12

Complementing the interviews, we accessed and studied additional data sources in form of project documentation, and video and audio materials describing the development of the GenAI systems and the resulting systems. These additional data sources enabled a validation of our understanding of the development process of the GenAI systems as well as the systems features, allowing us to triangulate our findings. Table 1 summarizes our data sources. Table 2 describes the GenAI systems projects.

Table 2: GenAI systems development projects

Project Description and User Intent	
A	IT service desk chatbot <i>Artifact:</i> GenAI-enabled chatbot to fulfill user requests to the IT service desk targeted to improve user experience and workload reduction of service desk agents <i>Intent:</i> Supporting a broad range of different user intents, for example IT issues with IT hardware, IT software, certificates, collaboration and communication or mobile services (broad intent)
B	Legacy code migration <i>Artifact:</i> GenAI-based legacy code migration solution to support software developers in migrating legacy code into state-of-the-art technologies targeted to accelerate legacy application retirement and IT modernization <i>Intent:</i> Supporting software developers in code translation of legacy programming languages (narrow intent)
C	Customer feedback insights <i>Artifact:</i> GenAI-based solution to identify the main topics in customer feedback aimed to improve customer experience reflected in the company's net promoter score (NPS) <i>Intent:</i> Supporting customer experience managers who work on improving NPS with identifying and summarizing the main customer feedback (narrow intent)

Our data analysis approach was inductive. Specifically, we draw on the grounded theory method of open, axial, and selective coding (Strauss & Corbin, 1998) to develop theoretical understanding on the phenomenon grounded in empirical data (Eisenhardt, 1989b). We make sure to link our analysis and coding to our research question as follows: First, as our core unit of analysis, we focus on actions taken in the interaction between developers, users, and the GenAI artifact, as they are key to developing GenAI systems. Second, our coding centers around actions taken during developing the GenAI systems to address the core challenge of managing the user-AI interaction. The first author conducted the coding. The resulting codes were critically discussed among all authors. Our coding process was highly iterative. In the initial coding round, preliminary concepts were identified, which were further validated and enriched by evidence emerging in later interviews (Saldaña, 2009). In addition, we used the documentation on the GenAI systems and systems projects to validate the concepts and relationships developed from the interview data.

We coded the interview data in three stages. In the first stage, we conducted open coding that

involved line-by-line coding. In the second stage, we further analysed and refined the constructs that emerged during open coding to form higher-level categories. For example, the first order concepts of *user input conversion* and *request of user inputs in user-AI interaction* both relate to managing the user interaction when interacting with GenAI. From these concepts, we developed the category of controlling user interaction. In the third stage, we applied selective coding with the goal to develop theoretical propositions to build understanding on how to develop effective GenAI systems to support organizational work. In this stage, we developed the core categories of *enablement* and *control*, which are coherently integrating all other identified categories. We found that, taken together, these two categories explain how to design and manage user-AI interaction that is at the core of GenAI systems development.

4 Findings

The GenAI systems developed in all projects support the performance of specific tasks with inherent business logic. For the GenAI systems to be able to provide support, they have to take the existing business context into account. In addition, user input in the user-AI interaction affects the output of a GenAI system, as it is used by GenAI to generate the responses. Consequently, in the user-AI interaction, GenAI systems need to consider (1) the business context, and (2) the way users interact with GenAI.

Our analysis of the three GenAI systems development projects identified an interplay of the two core mechanisms of *enablement* and *control* through which developers reconcile the two key challenges in the user-AI interaction. On the one hand, we found that enriching the capabilities of both the GenAI artifact and the users enabled an enhanced user-AI interaction. We refer to this mechanism as *enablement* of the user-AI interaction. On the other hand, we found that at the same time the user-AI interaction required guidance and steering of the GenAI artifact and of the user to generate useful outputs. We refer to this mechanism as *control* of the user-AI interaction.

While the interplay of the mechanisms of enablement and control was identified across all three GenAI systems development projects, we found differentiating instances of enablement and control depending on the GenAI system supporting a broad user intent or a narrow user intent.

4.1 Enablement and control for GenAI systems supporting a broad user intent

The IT service desk chatbot developed in project A of the case company is a GenAI system that supports broad user intent by helping users in resolving a broad range of different IT issues. The user-AI interaction materializes in form of conversations through which the GenAI system creates understanding of the user intent and generates corresponding answers. Accordingly, the

user-AI conversations that form the foundation of the user-AI interaction in case of broad user intent are subject to enablement and control when developing the GenAI system.

4.1.1 Enablement

Enablement of the user-AI interaction is provided towards the GenAI artifact and towards the users. Enablement of GenAI aims at providing the GenAI artifact the necessary business context to support the performance of a specific task. This includes *preparing context data* and *contextualizing GenAI*. Preparing context data refers to identifying the relevant context data sources and assuring data quality by data preprocessing such as data cleaning, data categorization or data augmentation. With contextualizing GenAI the business context data is provided so that the GenAI artifact can access it when generating outputs:

“A content workstream works on improving the document structure by adding headings, so that GenAI can capture the content better. The document content is optimized by splitting large documents and focusing them on the essential content, or by enriching documents containing sparse information with more comprehensive information.” (Kenji, User, Project A)

Enablement of the user-AI interaction towards the users provides users with enhanced abilities to interact with GenAI for fulfilling user requests. This is achieved by *enabling AI systems functioning understanding*, *enabling benefit understanding* as well as by *enabling interaction with AI*. We found that due to the unique characteristics of AI systems with their self-learning capacity and probabilistic nature resulting in inscrutability, users require support to grasp the non-deterministic nature of AI. As the GenAI systems are dedicated to specific tasks within the organization, enabling users with benefit understanding is necessary to equip users with the understanding on how their task completion can be improved by applying the GenAI systems. In addition, users are supported for a more effective interaction with the GenAI systems to address their requests, for example with communication and tutorials on the best practices for user-AI interaction:

“The users are provided with tips on how to formulate user requests to the chatbot most effectively within the chatbot interface and there is a broad communication on best practices for interacting with the chatbot. Users will need a learning phase how to properly prompt the chatbot.” (Kenji, User, Project A)

4.1.2 Control

In parallel to the enablement of the user-AI interaction, we found that control of the user-AI interaction towards GenAI and towards the users was the second instrumental mechanism to steer

and guide the user-AI interaction to be effective in addressing a specific task in the case organization. Since user-AI interaction for broad user intent is realized through user-AI conversations, *controlling the conversational flow* is being focused on. Control towards the GenAI artifact is carried out by *controlling the conversational flow with business rules*, *controlling the conversational flow with input-answer pairs*, and *validating the conversational flow*. Controlling the conversational flow with business rules ensures that the GenAI artifact follows specific business logic in the user-AI interaction and is realized by implementing a specific set of business rules to be considered by GenAI in the conversation with users. With controlling the conversational flow with input-answer pairs, GenAI should use a pre-defined set of answers to reply to specific user inputs. Validating the conversational flow is done to ensure the quality of the conversational flow in terms of correctly applying the business rules in the conversation. This is achieved via different means involving the users of the GenAI systems such as user testing before launch, regular quality checks with power users, or ongoing collection and analysis of end user feedback. The following comments reflect these control mechanisms:

“There is a huge difference between a business-ready chatbot and a chatbot like ChatGPT. ChatGPT never comes with the guarantee of information accuracy. In our case, we simply cannot afford any hallucinations. That’s why we need to restrict the chatbot with a framework of rules so that we harness LLM and at the same time can be sure that it does not hallucinate.” (Nikolai, Developer, Project A)

“We collect user feedback during interaction. In the conversation, users can say whether the user experience is positive or not. If not, users can provide open feedback. We can map this feedback to the corresponding conversation and analyse it in detail. This allows us to understand whether, for example, an inadequate or ambiguous description of the business rules has resulted in an inappropriate response and adjust the business rule description accordingly.” (Vijay, Developer, Project A)

With the mechanism control of the user-AI interaction oriented towards users, the focus is on controlling the user interaction so that the users are guided for a goal-oriented conversation with AI to solve a specific task with the help of the GenAI system. This is accomplished by *converting user inputs* and *requesting user inputs in conversation*. By converting user inputs, user prompts are transformed into a form that GenAI can process them in an enhanced way. Through the request of user inputs in conversation, required information can be provided to GenAI during the user-AI conversation for GenAI to properly handle the user request, for example by requesting users to rephrase their request more clearly or by requesting missing information that GenAI needs to be able to provide a helpful answer:

“We don’t send the user prompts one-to-one to the LLM. We are modifying the user prompts by putting them into context for the LLM. For example, if the intent of getting advice on how to apply certain IT services is identified, the chatbot is not just retrieving the existing manuals and direct the users to them, but we enhance the prompt and ask GenAI to provide in addition a crisp summary of the relevant paragraphs.” (Alexander, Developer, Project A)

“Some IT issues differ only in subtle aspects that it is difficult for the LLM to identify the correct business process that resolves the issues based on only a few inputs from the users. When users tend to be lazy and just provide a few key words, we stimulate the conversation by asking them to rephrase or we prompt the users to provide more information. We do this by specifying the question together with what type of information has to be collected from the user. This is handled by the LLM.” (Nikolai, Developer, Project A)

4.2 Enablement and control for GenAI systems supporting a narrow user intent

The GenAI systems for legacy code migration and customer feedback insights support a narrow user intent, respectively. The legacy code migration system is supporting software developers with translating a legacy programming language into a state-of-the-art programming language. The customer feedback insights system is supporting customer experience managers by identifying the central topics in the customer feedback. As both systems are dedicated to a narrow user intent, we found that the design of the user-AI interaction was narrowed down accordingly.

In contrast to broad user intent where conversations in the user-AI interaction is essential to identify the user intent, the intent is set for GenAI systems for narrow intent. Therefore, focus for developing GenAI systems for narrow intent is on the user-AI interaction following the clarification of the user intent. Put differently, for narrow user intent, user-AI interaction is not designed as user-AI conversation but focus on the design of inputs required from the users and the design of outputs required from GenAI to support a designated intent. For example, for the GenAI legacy code migration system in project B, input in form of code in the source legacy programming language is required from the users, and output in form of an adequate translation into the target programming language is needed. Accordingly, the *user input and the AI output* that form the key focus areas of the user-AI interaction in case of narrow user intent, are subject to enablement and control when developing the GenAI system. The differences in managing the user-AI interaction when developing GenAI systems for narrow user intent become evident in the differences of how developers realize enablement and control.

4.2.1 Enablement

For narrow user intent, enablement of user-AI interaction towards GenAI is realized through the same mechanisms as for broad user intent to provide GenAI with access to relevant business context. For the legacy code migration system in project B, the documentation of the legacy programming language as well as example translations from the source legacy language to the target programming language are provided as context to GenAI. However, for enablement towards users, instead of focusing on the enablement of the interaction of users with AI in case of broad user intent, the identified mechanism at play is *enabling input and output understanding*. Enabling input and output understanding specifies towards the users the input required from the users as well as the output of the GenAI system that users can expect to tackle the intended task with the support of the GenAI system:

“We need to foster an understanding of users that you can only get out of the system from what you put in. The data represents the voice of the customer, not company processes. If customers write that their internet connection is not working, the algorithm cannot know that it is due to an issue of the router. The output of the GenAI system depends on how the customer surveys are designed and how the customer responses turn out.” (Luis, Developer, Project C)

4.2.2 Control

Control of the user-AI interaction towards GenAI for narrow intent centers on *controlling the AI output* instead of controlling the conversational flow in case of broad intent. This includes *controlling outputs with business rules* and *validating the outputs*. Controlling outputs with business rules can entail defining the relevant business logic, breaking down the intended task with the required business logic into sub-components to be considered by GenAI or modelling a workflow with business rules to be followed by GenAI to generate adequate outputs. Validating the outputs again involves users in the validation, but focuses on the AI outputs rather than the conversational flow:

“In the system, three levels of categories for customer feedback are displayed. A huge discussion point with business departments and users is: What is a category? How should a topic be aggregated? Or when is a topic a topic, how much customer feedback needs to come together so that we can say, for example, that internet speed seems to be an important topic for customers.” (John, User, Project C)

“We have implemented workflows steering GenAI to carry out the code translation in steps. For example, we take the source code and translate it into natural language. We then approach the code generation into target language. In case users

want to use certain packages or frameworks, we import them. We can make the workflows as we like without losing any information. This allows us to work on more complex problems. We noticed, once we give GenAI a lot of information, not all the information is considered. If we give several instructions, sometimes some are left out. This is not the case with workflows because they cannot be skipped.” (Isabel, Developer, Project B)

Instead of the interaction of users being the focus of control in case of broad intent, controlling the user input is the primary concern when developing GenAI system for narrow intent. The mechanism we found in effect here is *delimiting user input*. With *delimiting user input* only specified user inputs are accepted by the GenAI system or inputs from users during use are not made possible at all:

“The prompts are pre-configured. In this tool, the users no longer write the prompt themselves.” (John, User, Project C)

“Based on experience, the quality of the output suffers significantly with prompts that are completely freely created by users. The acceptance and use of the tool depend on whether the output quality is satisfactory. If users do not achieve their desired outcomes – and this would be often the case with self-written prompts – the tool will not be used. Experts are needed to develop high-quality prompts. For our tool, the data scientists write the prompt themselves.” (Luis, Developer, Project C)

Table 3 provides a summary of the identified GenAI systems development mechanisms.

Table 3: Summary of mechanisms

	Broad Intent	Narrow Intent
Enablement of User-AI Interaction	Enabling GenAI with business context through preparing context data and contextualizing GenAI	
	Enabling users for user-AI interaction through enabling AI systems functioning understanding, enabling benefit understanding, and enabling interaction with AI	Enabling users for input-output understanding through enabling AI understanding of system functioning, system benefits, and system input and outputs
Control of User-AI Interaction	Controlling the conversational flow of GenAI with business rules, input-answer pairs, and validation	Controlling GenAI output through controlling outputs with business rules and validating the outputs
	Controlling user interaction through converting user inputs, and by requesting user inputs in conversation	Controlling user input through delimiting user input

5 Discussion and Implications

In our study, we identified that developing GenAI based systems to support organizational tasks with inherent business logic centers on designing the user-AI interaction. This is achieved through the core mechanisms of *enablement* and *control* of the user-AI interaction. Depending on whether the GenAI system supports broad or narrow user intent, developers instantiate the core mechanisms differently. In both cases, we observe an interplay between *enablement* and *control* to enhance the user-AI capabilities on the one hand and to converge them towards specific business logic on the other hand. Building on our findings, we explain how GenAI systems are developed to support organizational tasks with specific business context and how this is accomplished in the triadic relationship between developers, users and GenAI. Based on this, we propose a framework for developing GenAI-based systems supporting tasks in the organizational context illustrated in Figure 1.

With the *enablement* mechanism, the divergent nature of the GenAI system is strengthened by providing rich context data to GenAI to enhance its ability to generate a broader range of outputs, as well as by enabling users for an enhanced interaction with GenAI for example through learning extended prompting techniques resulting in their ability to produce varied outputs. In contrast, the mechanism *control* aims at converging both the GenAI interaction and the user interaction towards adhering to the specific business logic required to fulfill tasks in the organizational context. Therefore, in the development of GenAI systems, we find the developers to reflect on the need for divergence and convergence, respectively, when enacting *enablement* and *control* to harness the divergent nature of GenAI and at the same time to ensure that the GenAI system converges to the necessary business logic and produces useful and reliable outputs in the organizational context.

The mechanisms *enablement* and *control* are instantiated differently for broad user intent supporting a variety of user intents, and narrow intent aimed at accomplishing a specific, well-defined organizational task. In the case of narrow intent, where user intent is pre-set and user interaction to express user intent towards GenAI is not necessary, our findings show that user-AI interaction is narrowed down. This manifests in user inputs being delimited or rendered completely obsolete. With our framework we posit that through continuous *enablement* and *control* of the user-AI interaction, developers design GenAI systems that can serve organizational objectives.

With our study, we contribute to the literature on human-AI interaction. This field of literature has so far focused on studying user interaction with AI outputs to solve organizational problems with mainly two focus areas: (a) a focus on the AI artifact and the feasibility of AI outputs to solve problems in specific domains with humans verifying the AI outputs, altering the AI model and applying the AI outputs to specific tasks (e.g., Asatiani et al., 2021; Gronsund & Aanestad, 2020; Lebovitz et al., 2021; Raisch & Fomina, 2025; van den Broek et al., 2021); and (b) a

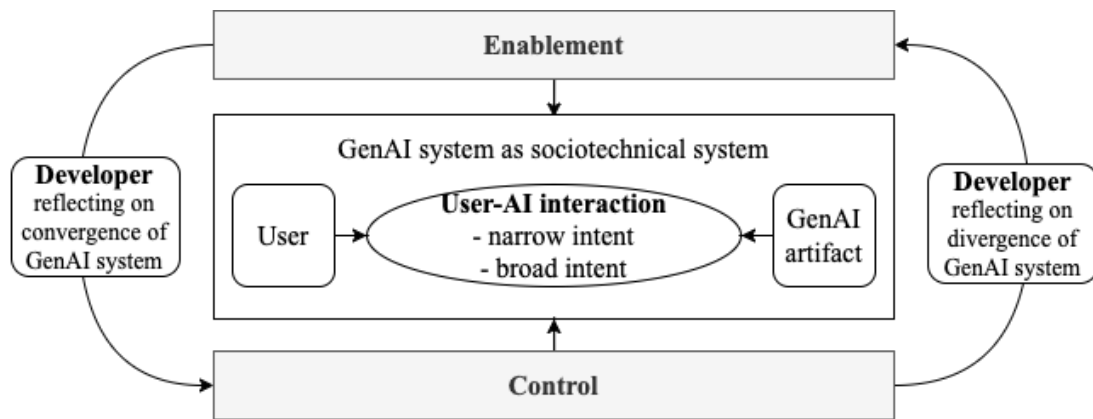


Figure 1: Framework for developing GenAI-based systems

focus on humans and the impact of AI outputs on humans (e.g., Fügener et al., 2021; Glikson & Woolley, 2020; Strich et al., 2021).

While both research streams have recognized the importance of the dyadic relationship between users and AI in leveraging AI outputs, our study adds to it by creating an understanding on the importance of human-AI interaction for generative technologies during development. In particular, our findings shed light on how to design the human-AI interaction both towards the AI agent and the human agent for GenAI systems to produce useful outputs. We found that the divergent characteristics of GenAI need to be considered through the enablement of human-AI interaction, while in parallel the control of human-AI interaction is essential for the GenAI system to converge on organizational goals. This offers a novel view on human-AI interaction for generative technologies.

For practitioners, our study reveals that developing a GenAI system is not merely about developing the technical system. Rather, it is critical to understand the GenAI system and its development as a socio-technical system. Specifically, for organizations aiming to leverage GenAI, our findings highlight the importance of focusing their development efforts on developing the user-GenAI interaction. The key mechanisms identified with enablement and control through which developers shape the user-GenAI interaction can serve as an orientation for leaders and experts involved in developing GenAI systems. In addition, we provide practitioners with an account of two types of GenAI systems supporting broad versus narrow user intent. Depending on the form of intent and related tasks, user-AI interaction is designed differently through distinct instantiations of enablement and control mechanisms. This can help them to deliberately design the GenAI system depending on the task and intents the system should support.

Limitations of our study stem from the boundary conditions of our work and provide avenues for further research. Our findings are informed by studying GenAI development projects focusing on GenAI systems supporting organizational tasks that require specific task context and business logic. This provides a revelatory case to study how to develop GenAI systems supporting specific organizational objectives. Going forward, studying the development of GenAI systems

supporting tasks that can be completed without the need to access additional context information and how the need for enablement and control mechanisms shifts could provide interesting complementary insights.

6 Conclusion

Organizations pursuing the development of GenAI-based systems that support organizational goals need to consider GenAI as a socio-technical system, going beyond the mere focus of the technical system. Managing the human-AI interaction becomes the focus of the development efforts involving both enablement to leverage the divergent nature of GenAI systems as well as control for the system to converge to business logic required for organizational work. Understanding these core development practices is key to realize the value potential of GenAI. Our findings and framework provide a first step to uncover the factors and mechanisms necessary to successfully design and develop human-GenAI interactions.

References

- Asatiani, A., Malo, P., Nagbol, P. R., Penttinen, E., Rinta-Kahila, T., & Salovaara, A. (2021). Sociotechnical Envelopment of Artificial Intelligence: An Approach to Organizational Deployment of Inscrutable Artificial Intelligence Systems. *Journal of the Association for Information Systems*, 22(2), 325–352.
- Baig, A., Merrill, D., & Sinha, M. (2024). Moving past gen AI's honeymoon phase: Seven hard truths for CIOs to get from pilot to scale. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/moving-past-gen-ais-honeymoon-phase-seven-hard-truths-for-cios-to-get-from-pilot-to-scale>
- Baird, A., & Maruping, L. M. (2021). The Next Generation of Research on IS Use: A Theoretical Framework of Delegation to and from Agentic IS Artifacts. *MIS Quarterly*, 45(1), 315–341.
- Benbya, H., & McKelvey, B. (2006). Using coevolutionary and complexity theories to improve IS alignment: a multi-level approach. *Journal of Information Technology*, 21, 284–298.
- Benbya, H., Strich, F., & Tamm, T. (2024). Navigating Generative Artificial Intelligence Promises and Perils for Knowledge and Creative Work. *Journal of the Association for Information Systems*, 25(1), 23–36.
- Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing Artificial Intelligence. *MIS Quarterly*, 45(3), 1433–1450.
- Berg, J. M., Raj, M., & Seamans, R. (2023). Capturing value from artificial intelligence. *Academy of Management Discoveries*, 9(4), 424–428.
- Bostrom, R. P., Gupta, S., & Thomas, D. (2009). A meta-theory for understanding information systems within sociotechnical systems. *Journal of Management Information Systems*, 26(1), 17–48.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., & Askell, A. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
- Chen, C. C., Liu, J. Y.-C., & Chen, H.-G. (2011). Discriminative effect of user influence and user responsibility on information system development processes and project management. *Information and Software Technology*, 53(2), 149–158.
- Cusumano, M. A. (2023). Generative AI as a New Innovation Platform. *Communications of the ACM*, 66(10), 18–21.
- Eisenhardt, K. M. (1989b). Building theories from case study research. *Academy of Management Review*, 14(4), 532–550.
- Fernández-Martínez, F., Luna-Jiménez, C., Kleinlein, R., Griol, D., Callejas, Z., & Montero, J. M. (2022). Fine-tuning BERT models for intent recognition using a frequency cut-off strategy for domain-specific vocabulary extension. *Applied Sciences*, 12(3), 1–15.
- Foster, D. (2022). *Generative Deep Learning*. O'Reilly Media, Inc.

- Fügener, A., Grahl, J., Gupta, A., & Ketter, W. (2021). Will Humans-in-the-Loop Become Borks? Merits and Pitfalls of Working with AI. *MIS Quarterly*, 45(3), 1527–1556.
- Fügener, A., Grahl, J., Gupta, A., & Ketter, W. (2022). Cognitive challenges in human–artificial intelligence collaboration: Investigating the path toward productive delegation. *Information Systems Research*, 33(2), 678–696.
- Garcia, M. (2024). What Air Canada Lost In ‘Remarkable’ Lying AI Chatbot Case.
- Glikson, E., & Woolley, A. W. (2020). Human Trust in Artificial Intelligence: Review of Empirical Research. *Academy of Management Annals*, 14(2), 627–660.
- Gronsund, T., & Aanestad, M. (2020). Augmenting the algorithm: Emerging human-in-the-loop work configurations. *Journal of Strategic Information Systems*, 29(2), 1–16.
- He, Z., Li, S., Song, Y., & Cai, Z. (2024). Towards Building Condition-Based Cross-Modality Intention-Aware Human-AI Cooperation under VR Environment. *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 1–13.
- Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., & Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12), 1–38.
- Johns, G. (2006). The essential impact of context on organizational behavior. *Academy of Management Review*, 31(2), 386–408.
- Kasthuriarachchy, B., Chetty, M., Karmakar, G., & Walls, D. (2020). Pre-trained language models with limited data for intent classification. *2020 International Joint Conference on Neural Networks*, 1–9.
- Keil, M., Rai, A., & Liu, S. (2013). How user risk and requirements risk moderate the effects of formal and informal control on the process performance of IT projects. *European Journal of Information Systems*, 22, 650–672.
- Lebovitz, S., Levina, N., & Lifshitz-Assaf, H. (2021). Is AI Ground Truth Really ‘True’? The Dangers of Training and Evaluating AI Tools Based on Experts’ Know-What. *MIS Quarterly*, 45(3), 1501–1525.
- Lyytinen, K., & Newman, M. (2008). Explaining information systems change: a punctuated socio-technical change model. *European Journal of Information Systems*, 17, 589–613.
- Rahwan, I., Cebrian, M., Obradovich, N., Bongard, J., Bonnefon, J.-F., Breazeal, C., Crandall, J. W., Christakis, N. A., Couzin, I. D., & Jackson, M. O. (2019). Machine behaviour. *Nature*, 568(7753), 477–486.
- Raisch, S., & Fomina, K. (2025). Combining human and artificial intelligence: Hybrid problem-solving in organizations. *Academy of Management Review*, 50(2), 441–464.
- Russell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach*. Pearson.
- Rzepka, C., & Berger, B. (2018). User interaction with AI-enabled systems: A systematic review of IS research. *International Conference on Information Systems (ICIS) Proceedings*.
- Saldaña, J. (2009). *The Qualitative Coding Manual*. Thousand Oaks.

- Sarker, S., Xiao, X., Beaulieu, T., & Lee, A. S. (2018). Learning from first-generation qualitative approaches in the IS discipline: An evolutionary view and some implications for authors and evaluators (PART 1/2). *Journal of the Association for Information Systems*, 19(8), 752–774.
- Strauss, A., & Corbin, J. (1998). *Basics of qualitative research techniques*. Sage Publications.
- Strich, F., Mayer, A.-S., & Fiedler, M. (2021). What do I do in a world of artificial intelligence? Investigating the impact of substitutive decision-making AI systems on employees' professional role identity. *Journal of the Association for Information Systems*, 22(2), 9.
- van den Broek, E., Sergeeva, A., & Huysman, M. (2021). When The Machine Meets The Expert: An Ethnography of Developing AI for Hiring. *MIS Quarterly*, 45(3), 1557–1580.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
- Wu-Gehbauer, M., & Rosenkranz, C. (2024). Towards a Framework for Developing Artificial Intelligence Systems for Human-Artificial Intelligence Augmentation. *European Conference on Information Systems (ECIS) Proceedings*.
- Yin, R. K. (2009). *Case study research: Design and methods* (Vol. 5). Sage Publications.

Paper IV

Delegation in Developing Artificial Intelligence-based Systems – How Developers Construct Conditions for User-Artificial Intelligence Delegation

Mei Wu-Gehbauer
University of Cologne

Christoph Rosenkranz
University of Cologne

Completed Working Paper, presented at the Paper Development Workshop AIS SIG DITE in 2025

Delegation in Developing Artificial Intelligence-based Systems – How Developers Construct Conditions for User-Artificial Intelligence Delegation

Abstract

Artificial intelligence (AI)-based systems increasingly act as goal-directed agents, raising the question of how responsibility for tasks is delegated and governed. Existing IS research treats delegation mainly as a dyadic user-AI relationship at use time. This study examines how developers work with AI-based systems whose behavior they cannot fully specify or inspect, and how their development practices condition user-AI delegation. Based on a multiple-case study, we introduce the concept of delegation-in-development: Developers hand over parts of their traditional responsibility for designing the system's decision logic to learning and partly opaque systems through appraisal-, distribution-, and coordination-in-development practices. These practices create two interlinked delegation cycles, between developers and the AI-based system and between users and the AI-based system. Additionally, we find that these delegation cycles are shaped by developer's temporal orientations of iterative, projective, and practical-evaluative orientations, which give rise to distinct ways of allocating responsibility to the AI-based system and of enabling user-AI delegation. The study advances delegation theory to a triadic perspective and reconceptualizes information systems development (ISD) for AI-based systems as the design and governance of responsibility transfers.

Keywords: Agentic systems, artificial intelligence, information systems development, human-AI delegation, developer agency

1 Introduction

Humans collaborating with artificial intelligence (AI)-based systems to perform work is increasingly emerging as a new form of organizing in contemporary workplaces (Mayer et al., 2025b). Such *human-AI collaboration* is characterized by humans and machines combining their complementary strengths to multiply their capabilities: humans contribute contextual knowledge and judgment, while AI-based systems provide computational power and pattern recognition capabilities (Raisch & Krakowski, 2021). This development is particularly evident in knowledge-intensive domains such as healthcare, customer service, or software development, where human-AI collaboration becomes central to accomplishing complex expert tasks (Raisch & Fomina, 2025). While organizations are investing heavily in AI to augment human capabilities, many AI initiatives struggle to move beyond the pilot phase and fail twice as often as traditional IT projects (Baig et al., 2024; Bojinov, 2023).

These challenges point to a transformation in the assumptions underlying information systems development (ISD). For decades, ISD has been structured around a relationship between users

and developers (e.g., Bostrom & Heinen, 1977b). This relationship was conceptualized as one in which developers elicit knowledge from users and then encode it into deterministic systems that reliably execute predefined logic (Forsythe, 1993). In this framing, developers inscribe knowledge and assumptions into rule-based systems (e.g., Hanseth & Monteiro, 1997). The system itself was not an actor – it was an artifact fully determined by human specification. However, these assumptions become untenable in the context of AI-based systems. AI-based systems, particularly those relying on machine learning, are no longer designed by specifying explicit rules *ex ante*. Instead, they self-learn, infer causal relationships from data on their own, and operate in ways that are not fully transparent (Berente et al., 2021).

A defining feature of this new generation of AI-based systems is their *agentic nature*, that is, their capacity to pursue goal-oriented behavior, to act autonomously, and to operate without continuous human intervention (Baird & Maruping, 2021; Russell & Norvig, 2016). To account for this change, Baird and Maruping (2021) propose a *delegation*-based perspective on IS use. In their delegation framework, the relationship between humans and agentic IS artifacts is conceptualized as a *dyadic* configuration, where humans transfer decision authority and coordinate tasks with the AI-based system. This framework marks a significant step forward in moving beyond traditional conceptions of IS use toward theorizing AI-based systems as actors in their own right. As the framework centers on delegation during use, it leaves open the question of how delegation relationships are constructed during systems development and what role developers play in shaping them. This is problematic because the agentic attributes of AI-based systems that are central to the delegation processes are designed by developers (Baird & Maruping, 2021). Developers embed the agentic attributes, such as goal-orientation, learning capacity, and autonomy, into AI-based systems that define their behavior. In doing so, developers design how users will later delegate to the AI-based systems. They make design choices that determine what responsibilities can be assigned, and what forms of coordination between the user and AI-based systems will be possible (Baird & Maruping, 2021). In this sense, user-AI delegation is not only enacted by users during use but also constructed by developers during development. Understanding delegation, therefore, requires attention not only to practices during use but to how developers configure the very conditions under which delegation is possible.

To understand how the development of AI-based systems is different from traditional ISD, information systems (IS) research has begun to investigate the development practices through which these systems are built. Several streams of research have emerged, each emphasizing different aspects of developing AI-based systems and the resulting challenges in human-AI interaction: (1) human-centered AI design research (e.g., Auernhammer, 2020), (2) machine learning workflows from a technical perspective (e.g., Amershi et al., 2019a), and (3) developing AI-based systems that enable user accountability and that enable users to make sense of and work together with the AI system (e.g., Asatiani et al., 2021; Kane et al., 2021). Across these research streams, the developer's role is largely instrumental, either as technical implementers of system

functionality or as facilitators who support users in articulating needs and addressing challenges in the user-AI interaction. Yet, developers themselves face a fundamental shift in how they relate to the AI-based systems they build. In traditional ISD, developers fully determine system behavior by encoding rule-based logic. But with AI-based systems, this control is partial at best. Developers now delegate their own traditional responsibilities, such as rule definition and decision logic design, to the AI-based system. They must work with systems that self-learn, behave opaquely, and generate outputs in ways they cannot fully anticipate. Specifically, AI-based systems learn a probabilistic decision logic that developers cannot fully specify *ex ante* or inspect *ex post* (Berente et al., 2016; Faraj et al., 2018; Zhang et al., 2021). As a result, developers are engaged in a dual process of delegation: they construct the user-AI delegation relationship, and simultaneously, they offload parts of their own traditional design responsibility of constructing the decision model to the AI-based system. These dual delegation aspects of the developers' role render the dyadic view of delegation insufficient. Instead, understanding human-AI delegation requires a *triadic* perspective that incorporates developers as central actors in configuring the delegation between users and AI-based systems. With this study, we foreground developers as actors as well as their challenges in the relationship with agentic AI when configuring human-AI delegation. Specifically, we ask: *How do developers enact development practices when developing AI-based systems whose behavior they cannot fully specify or inspect, and how do they thereby construct conditions for the user-AI delegation?*

This study addresses this question by adopting a delegation lens. To answer our research question, we conducted an exploratory, multiple-case study of eight AI-based systems development projects at a leading international telecommunications company. The selected cases span both discriminative and generative AI-based systems. This sampling strategy allows us to examine how developers construct human-AI delegation differently depending on how they can configure the agentic properties of the underlying AI-based systems.

We draw on *delegation* (Baird & Maruping, 2021) and *agency* (Emirbayer & Mische, 1998) as complementary sensitizing lenses to theorize the construction of delegation within the triadic relationship between developers, users, and AI-based systems. The delegation lens helps us analyze how decision authority and task responsibility are distributed, both from developers to the AI-based system, and how developers anticipate and structure user delegation. The agency lens enables us to surface the temporally embedded assumptions, intentions, and actions of developers as they shape the conditions under which users will engage in delegation. Together, these lenses enable us to theorize how human-AI delegation is constructed in the triad during the development of AI-based systems.

2 Theoretical Background

2.1 Shifting Nature in Information Systems Development of AI-based Systems

Developing AI-based systems is unlike information systems development (ISD) in the past (Rosenkranz et al., 2025). Today, machine learning algorithms are at the core of contemporary AI-based systems that are trained on data and can act autonomously (Berente et al., 2016). From a development perspective, such systems involve two main components: (1) the *goal* that reflects the desired outcome to be achieved (*what* is to be done), and (2) the *decision model* that defines the problem-solving strategy and procedure to achieve the goal (*how* it should be done) (Baird & Maruping, 2021; Russell & Norvig, 2016).

For decades, the role of developers in IS development has been to translate user requirements, with both the goal and the way to achieve it, into structured formulas or rule sets encoded into source code (e.g., Bostrom et al., 2009; Hayes-Roth & Jacobstein, 1994). Once compiled, these systems executed tasks in a predetermined manner, reflecting goals and the decision logic fully defined by developers (Forsythe, 1993). As a result, system behavior based on the decision model remained stable and transparent during use (Gregor & Benbasat, 1999; Hayes-Roth, 1997). Consequently, ISD literature has traditionally focused on the dyadic relationship between users and developers, and their tensions created by goal incongruencies and information and knowledge asymmetries (e.g., Gallivan & Keil, 2003). These were addressed through various ISD approaches, including user participation (e.g., He & King, 2008; Henfridsson & Lindgren, 2010; Hunton & Beeler, 1997), human-centered design (e.g., Kling, 1977; Kozar & Mahlum, 1987), prototyping (e.g., Naumann & Jenkins, 1982), and agile practices (e.g., Lee & Xia, 2010; Maruping & Matook, 2020). The common premise remained that the system's decision model was authored and controlled by developers and could be fully understood, tested, and adjusted by humans.

With agentic AI-based systems, the design relationship between developers and the AI-based system changes fundamentally. AI-based systems are agentic systems with the ability to self-learn decision models and act autonomously (Berente et al., 2021). They are agentic in that they can operate with a degree of autonomy, learn from data, and adapt their behavior without explicit reprogramming by humans (e.g., Baird & Maruping, 2021; Berente et al., 2021; Chollet, 2021). The role of developers changes in a way that goals are embedded into the agentic AI system by developers, but the decision model is self-learned by the AI-based system based on data (Baird & Maruping, 2021; Murray et al., 2021; Ozkaya, 2020). As such, developers delegate the development of decision rules to learning algorithms while they focus on setting the goals, identifying and cleaning the training data, selecting the learning algorithms, tuning the parameters, and deciding which algorithms to depend on in which context (e.g., Amershi

et al., 2019a). For example, Strich et al. (2021) reported on a case company that introduced an AI-based credit decision system that autonomously learns its decision model to make loan decisions, determine the terms and conditions of loans, and alter lending criteria based on customer behavior and current market changes.

The resulting decision model itself remains unavailable for user scrutiny, and it is even for developers not fully transparent how AI systems exactly reach their outcomes (Burrell, 2016; Faraj et al., 2018). During model use, small adjustments in the model or added data points can lead to model drift with unintended consequences that are also difficult for developers to predict (Amershi et al., 2019b; Sculley et al., 2015). Traditional ISD approaches are inadequate for tackling these challenges faced by both users and developers.

A nascent stream of research in IS has begun to explore the implications of AI-based systems entering the ISD process as an own actor. The research streams can be clustered in three main themes: (1) human-centered AI design (e.g., Auernhammer, 2020), (2) machine learning workflows from a technical perspective (e.g., Amershi et al., 2019a), and (3) developing AI-based systems that enable users to make sense of and work together with the AI-based system (e.g., Asatiani et al., 2021; Kane et al., 2021). The first stream builds on the tradition of human-centered design and translates established ISD principles that focus on the user-developer relationship to the domain of AI. This work emphasizes the importance of developers considering user needs and values during development (van den Broek et al., 2021), and of ensuring transparency and fairness in AI-based system behavior (e.g., Auernhammer, 2020). Developers in this perspective are seen primarily as value stewards, responsible for integrating ethical concerns and user needs into the system. The second stream focuses on the transformation of software engineering workflows to accommodate the particularities of machine learning development. This work studies how development practices are adapted to the data- and model-driven nature of AI, including changes to data management, engineering, testing, deployment, and monitoring (e.g., Amershi, Begel, et al., 2019). Developers are framed here as technical implementers, tasked with implementing the machine learning workflow. A third stream of research engages more directly with the distinctive characteristics of AI-based systems, such as their capacity for self-learning, their opacity, and their autonomous behavior, which pose challenges in the human-AI interaction. However, these challenges are primarily examined from the user's perspective. This stream investigates how such characteristics of AI-based systems challenge interpretability by users, the importance of domain knowledge, or user oversight. In doing so, it highlights the increasing importance of users, particularly domain experts, in shaping the development process, for instance by keeping users "in the loop" to maintain oversight of the behavior of the AI-based system (Gronsund & Aanestad, 2020) or designing explainable AI outputs to support user interpretation and accountability (Arrieta et al., 2020; Asatiani et al., 2021). Developers are considered enablers who support users in overcoming these challenges.

While the research streams center around different dyads of relationships during development

of AI-based systems, there is limited understanding of the role of developers in the triadic relationship among developers, users, and the AI-based system. However, developers face similar challenges in their interaction with AI-based systems as users. We do not have a good understanding of how developers deal with these challenges during development and how this affects the user-AI delegation during use. This is important since decisions taken by developers can condition the user-AI delegation (Baird & Maruping, 2021). We propose that leveraging the delegation lens in the triad between users, developers, and the AI-based system provides a fruitful avenue to generate more nuanced insights regarding the role of developers and their relationship with AI-based systems during development as well as how this shapes the user-AI delegation (Ågerfalk, 2020; Baird & Maruping, 2021).

2.2 Human-AI Delegation from a Triadic Perspective

Baird and Maruping (2021) introduce *delegation* as a central concept for understanding relationships between humans and agentic IS artifacts such as AI-based systems. They define delegation as the transfer of rights and responsibilities for task execution between agents. With this delegation framework, they shift attention away from traditional use-centric IS theories, which focus on how humans adopt and use technologies. Instead, they advance a delegation-centric perspective that asks what humans hand over to an IS artifact, under which conditions, and how this handover is governed over time. This shift is motivated by the observation that contemporary AI-based systems no longer operate as passive tools. AI-based systems perform tasks under uncertainty, initiate actions, and pursue goals in a partly autonomous and goal-directed manner. This repositions AI-based systems as agentic actors and makes delegation a key concept for theorizing human-AI relationships.

In their framework, Baird and Maruping (2021) characterize an agentic IS artifact through three *agentic attributes*: its preferences (goals and decision models), endowments, and role. Preferences comprise the goals the system is meant to pursue and the decision model through which it selects among options. *Goals* specify desired outcomes and thereby delimit what counts as successful action. The *decision model* represents causal beliefs about how inputs, actions, and outcomes are linked, allowing the system to rank alternatives and guide decisions. *Endowments* refer to the resources that enable the AI-based system to act, such as access to data and digital infrastructure for data processing. *Role* captures the rights and responsibilities that the AI-based system is allowed to assume in the work system.

In this delegation framework, delegation is discussed from the perspective of a user-AI dyad. Baird and Maruping (2021), however, already note that this *dyadic view* is incomplete and call for extending the framework to include developers, who play a central role in making user-AI delegation possible. Developers, together with users, decide on goals, design and train decision models, provide endowments, and specify the role of the AI-based system. These choices take

place prior to use and thereby set the conditions under which user-AI delegation can later occur. We take up this call and adopts a triadic perspective on delegation, involving developers, users, and the AI-based system. In this view, developers determine the agentic attributes of the AI-based system. This triadic perspective enables an understanding of how developers' design choices, together with users' input and the AI-based system's learned behavior, jointly form the conditions for delegation that users encounter in use.

This *triadic* perspective is particularly relevant because a key step of this study is to extend the delegation lens from the user-AI dyad to the developer-AI relationship itself. Traditional ISD perspectives often treat developers and the AI-based system as part of the same technical domain (Amershi et al., 2019a; Sculley et al., 2015; Studer et al., 2021). However, AI-based systems learn a probabilistic decision logic that developers cannot fully specify *ex ante* or inspect *ex post* (Berente et al., 2016; Faraj et al., 2018; Zhang et al., 2021). This means that the path from inputs to outputs and the resulting decisions of the AI-based systems are not entirely under developers' control. Against this backdrop, we examine the developer-AI relationship through the delegation lens. Developers hand over part of their traditional responsibility for determining the decision model to the AI-based system. Instead of explicitly programming all decision rules, they let the AI-based system learn its own decision model from data. Developers thus become principals who retain responsibility for outcomes but delegate parts of the procedure for reaching those outcomes to a learning system (Baird & Maruping, 2021; Eisenhardt, 1989a) whose inner workings may be partly opaque (Burrell, 2016; Faraj et al., 2018; Russell & Norvig, 2016). Treating the developer-AI relationship as delegation foregrounds that the development work for learning and partly opaque systems is a problem of handing over responsibility and managing that handover.

To analyze how this delegation between developers and the AI-based systems unfolds, we draw on the three core mechanisms of delegation identified by Baird and Maruping (2021): appraisal, distribution, and coordination. *Appraisal* refers to how human agents assess what is at stake for them in relation to the system and what can be done in response (Beaudry & Pinsonneault, 2005; Fadel & Brown, 2010). In a delegation context, appraisal focuses on identifying beneficial complementarities between human and AI capabilities (Baird & Maruping, 2021). This entails assessing the capabilities of the agents with respect to the task fit (Pinski et al., 2023), as well as the compatibility of their decision-making processes for a given task (Baird & Maruping, 2021). Appraisal may change as humans learn more about the capabilities of the AI-based system during interaction or when the system adapts and is updated (Baird & Maruping, 2021). In this study, appraisal is used to examine how developers assess user capabilities, AI capabilities, and their compatibility to decide how their own responsibility can be handed to the AI-based system.

Distribution addresses the actual transfer of rights and responsibilities between agents. Because delegation is tied to goals, this transfer must be governed so that the agentic IS artifact pursues the intended objectives (Baird & Maruping, 2021; Eisenhardt, 1989a). Distribution is

realized through contracting and regulating. *Contracting* concerns how agents agree, explicitly or implicitly, on goals and acceptable behavior, identify misalignments and define how conflicts should be resolved (Castelfranchi & Falcone, 1998; Eisenhardt, 1989a; Hadfield-Menell & Hadfield, 2019). *Regulating* refers to ex-ante control activities that set boundaries on the agent's behavior, for example, by limiting data access or constraining the actions the AI-based system may autonomously take (Baird & Maruping, 2021; Kirsch, 1996). In the context of AI-based systems, developers face the challenge that they cannot fully align the system's behavior with their goals because they can only set objectives and constraints that guide the system's learning (Hadfield-Menell & Hadfield, 2019). As various cases show, developers may design an AI-based system for specific goals, yet the deployed system can still produce harmful or unexpected results (Amodei et al., 2016). In this study, distribution is used to analyze how developers design goal constraints, learning constraints, ex-ante control mechanisms to govern the responsibility of the decision model construction they hand to the AI-based system during development.

Coordination refers to aligning actions and tasks under conditions of interdependence and uncertainty so that work advances and goals are achieved (Gulati et al., 2012; Malone & Crowston, 1994). For delegation, coordination is crucial because the delegator must monitor how delegated tasks are carried out and how this affects interdependent work. Baird and Maruping (2021) emphasize mechanisms such as monitoring and updating between humans and the agentic IS artifact. The delegator checks how the artifact has performed its work, the artifact reports outcomes through various representations, and both sides adjust their behavior accordingly (Okhuysen & Bechky, 2009). Delegation, therefore, entails not only transferring responsibilities but also coordinating this transfer to maintain accountability, predictability, and common understanding. In this study, coordination is applied to examine how developers, in their role as delegators, monitor the AI-based system during development and how they respond to what they observe.

We use the delegation lens to move beyond a dyadic user-AI view and to theorize how developers delegate to the AI-based system during development and thereby set the conditions for later user-AI delegation.

2.3 Agentic Attributes of Discriminative and Generative AI-based Systems

We study the emergence of delegation relationships in the triad for two types of AI-based systems that are widely adopted by organizations: discriminative AI and generative AI (Berg et al., 2023; Chui et al., 2023a; Foster, 2022). By learning the relationships between inputs and outputs or by learning the structures within the input data, discriminative AI classifies inputs, predicts labels given one or more inputs, or clusters inputs (James et al., 2021). Generative AI, in contrast, learns the underlying structure and patterns in the data to generate entirely new data that is similar to a given dataset (Foster, 2022). Rather than predicting a label or category, generative

AI generates new content, such as text or images (Foster, 2022). For generative AI, we focus specifically on large language models (LLMs), a subset of generative AI for text understanding and text generation, due to their growing relevance for organizations (Yang et al., 2024). LLMs are general-purpose foundation models that are pre-trained on vast and diverse text-based data (Touvron et al., 2023), making them applicable across a variety of domains (Chui et al., 2023a) – such as customer support (Brynjolfsson et al., 2023), internal knowledge management (Alavi et al., 2024), or software development (Peng et al., 2023).

Discriminative AI-based systems and generative AI-based systems differ in how developers can intervene regarding the systems' agentic attributes, including goals, decision models, endowments, and roles.

The agentic attributes of discriminative AI-based systems are typically developed in the triadic relationship of developers, users, and the AI-based system (Baird & Maruping, 2021; Glaser et al., 2021; Waardenburg & Huysman, 2022). Developers, in collaboration with users, embed the goals into the system and provide endowments, such as selecting learning algorithms, providing domain- and task-specific data, as well as infrastructure for data processing. The system's decision model then emerges through learning on its own via optimization procedures, which fit the model to the specified goals using the provided endowments (Goodfellow et al., 2016). The role of the discriminative AI-based system is established to support specific tasks where the AI-based system is perceived as being helpful to achieve these goals (Baird & Maruping, 2021).

In contrast, the agentic attributes of generative AI-based systems, specifically LLMs, emerge differently. LLMs come with predefined goals and pre-trained decision models, developed outside of the organization and without the involvement of developers and users who engage with the LLMs in the organizational setting (Radford et al., 2018; Touvron et al., 2023). The decision model of LLMs is shaped by learning the general patterns and structures of text and language through training on vast, heterogeneous text-based datasets that are not specific to an organization, domain, or task (Radford et al., 2018; Touvron et al., 2023). LLMs model the probability distribution of tokens, where a token refers to a word, subword, or character (Foster, 2022). Given a preceding input context, LLMs predict the likelihood of the next tokens and generate the next token by sampling from this distribution (Vaswani et al., 2017).

As such, LLMs are initially general-purpose models, neither optimized for a specific organizational context nor for task-specific goals (Brown et al., 2020; Radford et al., 2018). Rather than directly embedding goals and developing the decision model, LLMs can be adapted for specific tasks in the collaboration of developers and users in two ways: (a) by shaping the goals through providing inputs such as task instructions and examples in the interaction, often referred to as prompts (Brown et al., 2020; Liu et al., 2023) that can be either provided as predefined system prompts or dynamically provided as user prompts; and (b) by providing additional endowments in the form of additional, task-specific knowledge base from a domain or an organizational context that can be accessed by the LLM during output generation without additional training, re-

ferred to as retrieval-augmented generation (RAG) (Lewis et al., 2020). This does not change the underlying decision model. Instead, it influences how the model accesses external task-specific and context-specific knowledge and generates responses based on it. As a result, the role of the generative AI-based system is fluid and depends on how dynamically goals and endowments are specified and provided by developers and users. We note that LLMs can be further trained through fine-tuning, modifying their decision models (Radford et al., 2018; Roziere et al., 2023). However, we focus on the adaptations as described above without the need to fine-tune the model, as they are predominant ways through which organizations adapt LLMs for specific tasks. The differences in how developers can configure the discriminative and LLM-based generative AI systems are summarized along the agentic attributes in Table 1.

Table 1: Overview of discriminative and LLM-based generative AI systems along the agentic attributes (as specified by Baird and Maruping (2021))

Agentic Attributes	Discriminative AI-based System	Generative AI-based System (focused on LLMs)
Goal	Goals are embedded into the discriminative AI-based system by developers in collaboration with users (Baird & Maruping, 2021)	Goals are predefined outside of the organization as next-token prediction rather than an organization-specific task goal (Radford et al., 2018; Touvron et al., 2023). Task goals can be shaped in practice at inference time via prompting, through predefined system prompts or user prompts (Brown et al., 2020; Liu et al., 2023).
Decision model	The system's decision model emerges through learning, as optimization procedures fit the model to the developer-specified goals using endowments provided by developers and users (Baird & Maruping, 2021)	The decision model is pretrained outside of the organization as a foundation model (Radford et al., 2018; Touvron et al., 2023). Organizations typically apply these models unchanged (Saffarizadeh et al., 2024), though they can further adapt them through fine-tuning to embed factual knowledge in the model parameters (Radford et al., 2018; Roziere et al., 2023).
Endowments	Developers, in collaboration with users, curate and provide domain- and task-specific data and the infrastructure for data processing, thereby enabling the learning process (Amershi et al., 2019a)	LLM-based systems are initially trained on vast, heterogeneous text-based datasets that are not specific to a task or domain (Radford et al., 2018; Touvron et al., 2023). Their endowments can be dynamically expanded at inference time through retrieval-augmented generation (RAG), which provides access to domain- and task-specific documentation without retraining the model (Lewis et al., 2020).

2.4 Developer Agency

Developers' choices about how to configure the agentic attributes of discriminative and generative AI-based systems are contingent on their agency. The agency of human actors is their capacity to set goals and intentionally act to realize them (Bandura, 2006). Understanding developers' agency is therefore central to our inquiry: it surfaces developers' assumptions, intentions, and actions as they (a) configure the conditions under which users will delegate to the AI-based system and (b) delegate part of their own design work, specifically, the "how" of task execution, to the system itself.

Developers' agency is inherently temporal when developing AI-based systems. AI-based systems enable novel, unanticipated ways of working that developers need to imagine and design for and that users may not yet anticipate (Faraj et al., 2018). At the same time, in an organizational context, developers are faced with established routines and practices, as well as the user context that needs to be considered (Berente et al., 2016). Therefore, developing AI-based systems requires developers' attention to possible futures (what AI *could* do), to past patterns of actions (what *has* worked), as well as to present contingencies (what immediate constraints and trade-offs must be navigated). Because these considerations and design decisions introduce temporal orientations, we adopt the view of agency as a temporally embedded process (Emirbayer & Mische, 1998).

In this conception, agency comprises three temporal orientations that provide us with an analytical lens for understanding the actions of developers. First, a *projective* future-oriented mode encompasses the imaginative exploration of possible future trajectories of action, in which received structures of thought and action may be creatively reconfigured in light of actors' desires for the future. It enables us to examine how developers envision possible futures with the AI-based systems. Second, an *iterational* past-oriented mode involves the selective reactivation of prior patterns of thought and action, as incorporated in practical activity, thereby giving stability and continuity. It captures how developers draw upon past practices and routines in the development of the AI-based system. Third, a *practical-evaluative* present-oriented mode refers to the situated capacity of actors to make practical judgements among alternative possible trajectories of action in response to the emerging demands, dilemmas, and ambiguities of present situations. It emphasizes the situatedness of evaluations and decisions in response to the behavior of AI-based systems as well as in response to the needs of the actors, tasks, and the organization.

3 Research Design

3.1 Research Setting and Data Collection

We conducted an exploratory theory-building multiple-case study (Eisenhardt, 1989b), examining eight AI-based systems development projects in a large European telecommunications provider. The organization offered an ideal research setting due to its broad portfolio of AI initiatives spanning key business domains, including customer service, network operations, or enterprise IT. We had access to rich documentation of the AI-based system artifacts developed, as well as to both the development-side and user-side roles involved in the development. We began with discriminative AI-based projects (e.g., campaign personalization, incident prediction, network optimization) in comparable organizational conditions (e.g., developer team size, agile delivery framework, data privacy and governance framework) but different domains, and later included generative AI-based projects (e.g., legacy code migration, knowledge assistance).

Most projects investigated followed agile delivery methodologies that were adapted to AI-based projects (e.g., Amershi et al., 2019a; Laato et al., 2022). In each case, our primary informants were members of the developer team. In the agile setting, the developer team of AI projects is a multi-disciplinary team typically composed of data scientists, machine learning engineers, IT professionals, and individuals representing the business and user perspective, who are jointly responsible for the end-to-end delivery of the AI-based systems (Amershi et al., 2019a; Davenport, 2021; Laato et al., 2022; Patil, 2011). To complement the developer perspective, we also included the user perspective in all cases, either through interviews with end users, user leads, or individuals serving as user-side representatives. Consistent with agile development practices prevalent in the case organization, user representation is often assumed by the product owner (PO), who operates as a member of the agile developer team and functions as a proxy for business and user needs. The PO bridges the user and developer sides by prioritizing work in the developer team and articulating value from a user perspective. What unites all user and user representative roles is their domain expertise, which is essential for developing AI-based systems that support tasks that rely on specialized knowledge.

In total, we conducted 34 semi-structured interviews (see Table 2). While interviews were the primary source, to further strengthen the substantiation of our concepts, we triangulated this data with multiple other data sources per case, particularly with documentation of the projects and project artifacts, and internal communication materials (Yin, 2009).

We used theoretical sampling for cases and informants during data collection. Each of our cases served as an analytical unit. Our aim was not an exhaustive cross-case comparison but building cumulative insight through replication logic. Accordingly, we sought projects that allowed replication and extension of emerging explanations (Eisenhardt & Graebner, 2007). As the study progressed, the final composition of cases was shaped by insights from earlier coding.

Table 2: Overview of data

Project Case	AI-based System and its Task	# DEV	# USR	Total	Int. Min.	No of Docs & Video
A - Campaign Personalization	AI-based system as a decision engine for target audience selection for personalized customer campaigns (discrim. AI)	5	2	7	363	15
B - Mobile network maintenance	AI-based system for the prediction and for root cause analysis of incidents in the mobile network (discrim. AI)	2	4	6	307	4
C - Fixed-line network maintenance	AI-based system for the prediction of incidents in the fixed-line network (discrim. AI)	3	1	4	213	12
D - Home network optimization	AI-based system for the prediction of home network issues (discrim. AI)	2	1	3	170	14
E - Customer service ticket routing	AI-based system as a decision engine for ticket routing (discrim. AI)	2	2	4	200	2
F - Legacy code migration	AI-based coding system for legacy code migration (gen. AI)	3	1	4	229	8
G – IT incident resolution	AI-based system as knowledge assistant supporting the incident resolution in IT operations (gen. AI)	2	2	4	203	6
H - Customer feedback insights	AI-based system for the identification of main topics in customer feedback (gen. AI)	1	1	2	102	3
Total		20	14	34	1.787	64

We started with discriminative cases. These cases revealed a salient temporal orientation difference: in some cases, the appraisal and governance of the AI-based system were anchored in past practices and user decision logics (iterational), whereas in other cases, the orientation was toward an imagined future way of work with AI disrupting the current practices (projective). Building on this, we extended the case sample with additional discriminative AI-based projects to stabilize the emerging concepts. Moreover, because our developing explanation suggested the practical-evaluative orientation (judgments in the moment of user-AI interaction), we sought projects where such an orientation would plausibly manifest. Because this orientation was not dominant in the discriminative projects, this led us to include generative AI-based projects, where situated input-output assessment is central. We continued until additional cases and interviews yielded diminishing theoretical returns.

3.2 Data Analysis

Our data collection and data analysis proceeded iteratively. We employed a three-stage coding, applying open, axial, and selective coding (Strauss & Corbin, 1998). Figure 1 shows examples of our coding scheme. We used delegation as an initial sensitizing lens to examine how developers delegate in development while shaping conditions for delegation in use (Baird & Maruping, 2021). We attended to the temporal orientations of the actors, that is, how they drew on the past (iterational), imagined futures (projective), and made situated judgments in the present (practical-evaluative) (Emirbayer & Mische, 1998), as the AI-based systems took shape. Importantly, we introduced the agency lens during analysis. Only after open coding made temporal orientation salient did we adopt the agency framework to refine constructs and inform subsequent theoretical sampling.

We began with open coding to identify initial concepts. Early codes in some discriminative projects were around anchoring AI-based systems in established user decision logic and explainability (for example, prioritizing compatibility with user practices, using interpretable models, joint threshold setting, and encoding known patterns). In other discriminative projects, codes emphasized AI's learning capabilities and performance (for example, broadening inputs, improving ground truths, treating explainability as secondary to outcomes, A/B testing). Through iterative comparison, we noticed that the temporal orientation differentiated the discriminative cases and the corresponding codes. This recognition occurred during open coding, at which point we drew on the agency lens to refine our constructs along the three orientations (iterational, projective, practical-evaluative). This insight informed us to add generative AI-based cases to examine whether the practical-evaluative orientation indeed formed a distinct pattern for how developers delegate to the AI-based systems during development.

In axial coding, we then related categories to the first-order concepts to develop higher-level concepts explaining the phenomenon. For instance, for the aggregate dimension *regulating (in*

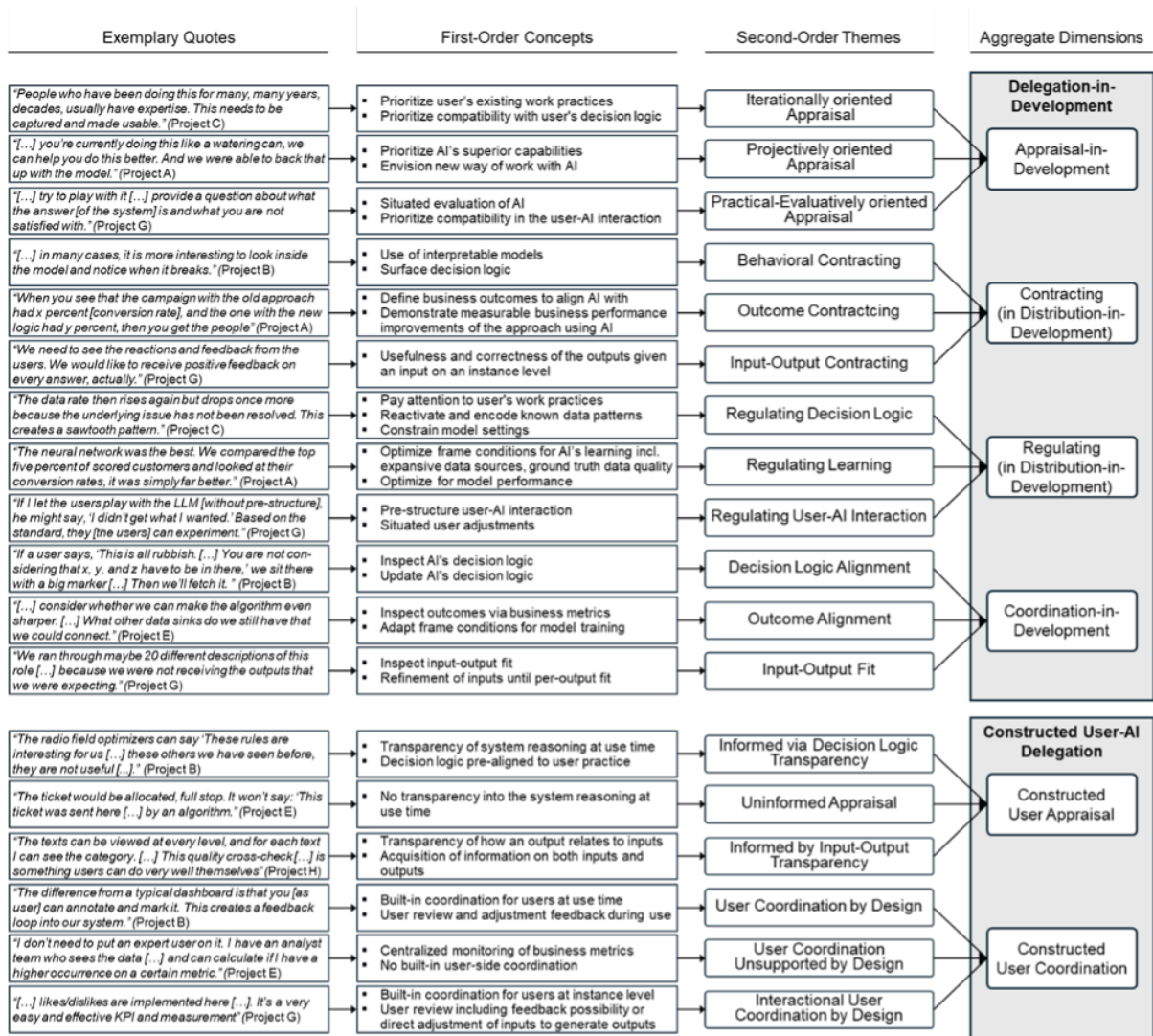


Figure 1: Data structure (analysis based on Strauss and Corbin (1998); data structure based on Gioia et al. (2013))

distribution-in-development), the way regulation unfolded differed by temporal orientation. In iterational patterns, conditions such as the salience of users' established decision practices led to actions that aligned the decision logic of the AI-based system with those practices (for example, through joint threshold setting, encoding known patterns when possible) with the consequence of decision logic transparency. In projective patterns, conditions such as perceived limits of the current practice and imagined AI-enabled futures led to actions that regulated learning conditions of the AI-based system (for example, through broadening inputs, improving ground truths, selecting the best performing model even when less explainable), with the consequence that appraisal relied less on internal reasoning and more on outcome alignment. In practical-evaluative patterns, the condition of situated usefulness led to actions that regulated user-AI interaction (for example, through pre-structured goal inputs, guided inputs, situated user adaption of inputs, per-instance checks), with the consequence of input-output transparency and interactional coordination. Accordingly, we specified three categories under the regulating dimension: *regulating decision logic*, *regulating learning*, and *regulating user-AI interaction*. Each category integrates the relevant open codes and specifies their typical conditions, actions, and consequences.

In the final stage (selective coding), we integrated the categories from axial coding into a theoretical scheme organized around two aggregate dimensions: *delegation-in-development* and *constructed user-AI delegation*. The purpose here is to relate the major aggregate dimensions to one another and build a coherent theoretical framework explaining the phenomenon. In our framework, delegation-in-development functions as the organizing core dimension, as it configures the forms of constructed user-AI delegation that become available in use. Within delegation-in-development, four sub-dimensions capture how developers enact delegation while building the AI-based system: *appraisal-in-development* (what the AI versus users should do), *contracting-in-development* (how goals and governance are specified), *regulating-in-development* (how the delegation to the AI-based system is regulated), and *coordination-in-development* (how alignment with desired goals is monitored and maintained). Constructed user-AI delegation comprises two sub-dimensions: *user appraisal* (how users can assess outputs) and *user coordination* (how users can monitor or correct the system during use). During selective coding, we linked these dimensions such that consequences produced in delegation-in-development became the conditions for appraisal and coordination in use. For instance: In an iterational configuration of delegation-in-development, design choices align the AI-based system with established user practices and its decision logic is made interpretable. The resulting decision logic transparency becomes the condition for user appraisal in use via visible model attributes. It also becomes the condition for user coordination. Coordination is built into the system (e.g., dashboards that can be annotated) so that users can examine and adjust those attributes. In effect, the configured forms user appraisal and coordination are themselves iterationally oriented according to the orientation established during development.

4 Findings

Developing AI-based systems entails a dual delegation process for developers: they must not only configure how users will delegate tasks to the AI-based system but also decide how to delegate their own responsibilities to the AI-based system. We find that these two aspects form a mutually constitutive process. How developers manage their own delegation to the AI-based system conditions how they design the user-AI delegation. Conversely, how developers design the user-AI delegation shapes how much responsibility they delegate to the system. The two delegation pathways co-evolve during system development, each conditioning the other.

Specifically, we find that developers adopt three dominant temporal tones when constructing user-AI delegation relationships. In the *iterational* mode of developing discriminative AI-based systems, developers anchor system design in established work practices, drawing on existing user work practices and task structures to shape the user-AI delegation. In the *projective* mode of developing discriminative AI-based systems, developers envision new ways of accomplishing tasks in the future that depart from existing practices. In contrast, in the *practical evaluative* mode, developers focus on the situational needs of the users when interacting with generative AI-based systems. These temporal orientations in constructing the user-AI delegation are tightly coupled with how developers delegate their own responsibilities to the AI-based system.

Together, the delegation processes during development reflect how developers simultaneously construct user-AI delegation and manage their own delegation to the AI-based system. This results in an interdependent delegation process of developers that is consequential for the design of the AI-based system and the design of the user-AI delegation. In the following, we present the findings in detail, starting with the findings for discriminative AI-based project cases, followed by findings for generative AI-based project cases.

4.1 Iterational Delegation-in-Development and Construction of User-AI Delegation in Discriminative AI-based Systems

Discriminative AI-based systems in our study were used to classify or predict well-defined outcomes, such as predicting incidents in the technical infrastructure or predicting customer behavior. Developers assess the AI-based system's capabilities in relation to users' capabilities in the context of specific tasks to appraise how the AI-based system and the users may contribute to accomplishing the tasks. This appraisal informs the temporal tone of developers' development practices. Depending on developers' appraisal, the dominant tone of the development practices for their own delegation to the AI-based system during development and for constructing the user-AI delegation is either *iteration* based on past patterns of users' actions or *projection* based on the possible future actions imagined with the AI-based system.

4.1.1 Iterational Delegation-in-Development

In the iterational mode of delegation-in-development, developers began with an *iterational appraisal* that centered on users' existing work practices. They judged domain experts as users as possessing a richer, causally grounded understanding of the problem space, while appraising the AI as efficient at optimizing predefined objectives yet limited in broader contextual understanding. As one developer put it when building a predictive maintenance system, "A machine learning algorithm is like a tunnel-visioned specialist. It only focuses on optimizing for the target variables and pushes this to the limit. The algorithm simply lacks a broader worldview that would allow it to say, 'That can't be right.' [...] We, as humans, operate on a very different knowledge base. We are anchored in the real world. We often see things that are about causality, not just correlation, much more clearly" (Paul, developer, project B).

This appraisal led developers to prioritize compatibility with users' decision-making process, anchoring the design of the AI-based systems in established user practices rather than displacing them, and to treat expert work practices as critical assets to be captured and reused. "Believing that we can automate everything with the latest, coolest AI methods is usually a misconception. People who have been doing this for many, many years, decades, usually have expertise. This needs to be captured and made usable" (Giorgi, developer, project C).

Based on this appraisal, developers established contracts as part of their *distribution-in-development* to establish desired goals for governing the transfer of their responsibilities to the AI-based system and to be able to identify misalignments. They applied *behavioral contracting*. This involved making sure that the AI's decision process was aligned with the user's decision process. To achieve this, the interpretability of the behavior of the AI-based system was essential. Developers together with users specified the guardrails for *how* the AI-based system could make decisions: "We develop the automation together with the users. They tell us: 'I always did this.' That's why we automate it" (Kristina, developer, project C). Where clear rules existed, developers encoded them explicitly rather than relying on self-learning by the AI-based system: "There is no point rolling out a machine learning algorithm when there is a clear rule" (Kristina, developer, project C). When developers did use learning methods, they emphasized explainability and intelligible decision logic. One developer described how the developers worked backward to surface the decision logic behind an AI model or switch to explainable algorithms to expose "combinations and preparations of attributes that are interesting to the users" (Paul, developer, project B). Developers emphasized that while powerful algorithms could marginally improve the prediction, detecting and understanding failure is often more valuable for governance: "[...] with a powerful technology I might squeeze out one, two, three percentage points [of improvement]. [...] But in many cases, it is more interesting to look inside the model and notice when it breaks. Because this tells you something about the data and about the processes" (Paul, developer, project B).

In their distribution-in-development processes, developers *regulated the decision logic* of the AI-based system to bound how it may act by aligning its behavior with user decision-making processes. They engaged closely with users to identify, preserve, and translate the decision rationales and logics underlying existing work practices into the AI-based systems. This approach, which was informed by users and guided by an iterational engagement with established and known practices, is well illustrated by Kristina (developer, project case C): “We sit down with them [the users] and ask: ‘What’s going on? How do you go about identifying the problem? We then try to reconstruct their approach and think about how we can technically recognize it, whether we can model it with rules, or whether a machine learning model would be more suitable.’” This quote exemplifies how developers sought to replicate users’ reasoning processes by orienting system design toward past user practices that proved to be effective in completing the work tasks.

The iterational regulation process unfolded through a series of interrelated practices. Developers began by investing time and effort to pay attention to the existing work practices of users. Through user workshops and feedback sessions, they aimed to identify expert knowledge that could inform the design of the AI-based systems. To promote users to share their expertise, developers engaged in fostering mutual respect. They signaled the value of domain expertise and actively made exaggerated expectations around AI transparent. As Giorgi (developer, project case C) put it, “I genuinely try to take people seriously and not talk nonsense. There’s so much being thrown at experts during the AI hype. Some come in and say: ‘I’ve got the latest AI tool, just throw in your data. Your expertise is basically useless.’ [...] I try to explain how certain machine learning concepts actually work. Even if people say, ‘...only 80 percent accuracy, that’s useless.’ [...] This honesty pays off, people remember you didn’t try to sugarcoat things.”

Based on the understanding of how users approach solving certain tasks, developers worked on surfacing, selectively reactivating, and encoding known data sources and patterns from user practices into the AI-based systems. The resulting design work translated expert rationales into rules and features that could be applied at scale. For example, in the fixed-line maintenance project, developers identified a characteristic “sawtooth” degradation pattern in incidents: “For certain individual [fixed network] lines and connections that have a problem, the data rate drops further, falls below a certain threshold, and as a result, the line reinitializes, it performs what’s called a full init and reconnects. The data rate then rises again but drops once more because the underlying issue has not been resolved. This creates a sawtooth pattern.” Developers first implemented detection of this pattern from available data, later improving it with a machine-learning model “to get even closer” (Kristina, project C). In the mobile network case when applying clustering, alternative algorithms and parameterizations were compared. The solution that best matched the expected error pattern known to users was selected: “We looked at how large the clusters were under each algorithm. Then we made a domain decision. Not mathematically justified, but the domain users said the error pattern best matched what we would expect” (Tomas,

developer, project B).

In other cases, developers constrained the model settings of the AI-based system to reflect user experience and judgment, ensuring that the system's logic was consistent with established user decision logic. In mobile network maintenance, the users co-determined the thresholds and the size of clusters in the model: "we brought the users in from the start. [...] they decided on the algorithm, determined the thresholds, and cluster size." (Tomas, developer, project B). In the fixed line maintenance case, developers provided a decision threshold slider and ran what-if analyses so users could select the cut-off that best matched operational trade-offs: "Here is the model, you pick the cut-off point [...] what would that mean if we apply it? 100 tickets come in, the model takes effect, how often did we get it wrong, how often right?" (Giorgi, developer, project C). These examples show that the AI-based systems were regulated by binding their key model parameters based on user experience and input. While developers treated expert practices as critical, they also emphasized the need to translate these into generalizable rules that can be reliably applied at scale. This required moderation between local users' anecdotal insights and data-driven validation. Through the validation of the generalizability of expert patterns and rules, developers ensured that patterns generalized beyond their local setting and did not induce false positives or systemic errors.

As part of the *coordination-in-development* processes and to monitor and coordinate the AI-based system behavior during development, developers centered on the *coordination of its decision logic alignment*. They enabled their own and user inspection of AI's decision logic by prioritizing interpretable algorithms and models that made it possible to "look under the hood," review data preprocessing and distributions, and scrutinize learned rules for alignment with domain logic. The technical scrutiny extended beyond aggregate performance metrics to whether decisions made sense in the domain. Developers explicitly favored simpler, transparent approaches over more complex but opaque models "to look under the hood much more easily," which improved their ability to trace the system's logic, how features were used, uncover issues, and exercise monitoring and control over the AI's decision logic.

In parallel, developers designed for structured user reviews so that expert user judgment and decision heuristics could be embedded as the system evolved. Developers implemented explainability mechanisms not merely for transparency but to enable users to validate and refine learned logic. They exposed intermediate artifacts, such as rule tables or feature-importance rankings, and conducted structured review sessions with users to detect mismatches with practice. In mobile network maintenance, for example, developers exposed attribute importances so that users could judge whether they reflected meaningful domain variables or were merely statistical artifacts. Attributes that were irrelevant or misleading were identified by users and subsequently excluded from the model. Developers welcomed user skepticism because it often exposed weaknesses in the AI models. The reviews could also reveal missing data sources that were critical: „If a user says, ‘This is all rubbish. [...] You are not even considering that x, y,

and z have to be in there,’ we sit there with a big marker and ask, ‘Which database contains that?’ Then we’ll fetch it. That’s incredibly valuable“ (Paul, developer, project B). Domain user corrections, therefore, drove targeted updates to rules, thresholds, and features, and underlying data, safeguarding reliability and keeping the system’s reasoning aligned with established practice.

4.1.2 Iterational Construction of User-AI Delegation

Developers *constructed user appraisal* in the iterational mode by designing for interpretability and transparency of the AI-based systems from development through use. Specifically, developers encoded user expert patterns and rules into the systems, co-specified and aligned features and thresholds to user domain decision logic and selected interpretable algorithms and models that were capable of exposing their decision rationale. In use, the system provided these pre-aligned explanation artifacts, such as attributes and thresholds driving the system behavior, as information to users. In this way, for the appraisal at use time, users were *informed by the transparency of the decision logic* of the AI-based system that is iterationally configured by developers during development together with users.

A second implication of the iterational mode of development for user-AI delegation is that aligning the AI-based system’s behavior with domain knowledge and user decision rationales did not end with development but extended into use. Developers *constructed user coordination* by designing the AI-based systems to provide users with explicit built-in means for coordinating the system behavior during use. In project case B, the ongoing alignment was operationalized through a dashboard interface that allowed users not only to observe system outputs but also to annotate and flag specific rules generated by the AI-based system. As Paul (developer, project B) described: “The difference from a typical dashboard is that you can annotate and mark it. This creates a feedback loop into our system. The radio field optimizers can say ‘These rules are interesting for us, we want to continue monitoring them. These others we have seen before, they are not useful, please don’t show them again.’” The system was designed to integrate this feedback into its next iteration of learning, allowing the AI-based system to learn from users’ monitoring and coordination. From the user side, the monitoring and coordination activities were formalized as a review and quality control process. Benjamin (user, project case B) described: “We have gotten into the habit of regularly reviewing the results every twelve weeks. Random results are drawn, and each team member works through 10 to 20 cases to check whether the system’s output actually makes sense.” This continuous coordination was viewed as critical due to the dynamic nature of the system’s environment, where input conditions regularly changed, for instance in project B with an AI-based system for predictive mobile network maintenance, due to factors like “new frequencies, technologies, antennas, or end devices”. Benjamin explained: “Many decision-makers think that once an AI automation is running, everything is fine. But if we don’t do this [review] regularly, the quality declines over time.” Through the design of

continuous user coordination, developers shaped how users could continue to guide and adjust the behavior of the AI-based system in use, thereby structuring the users' coordination as part of their delegation to the AI-based system during use.

4.2 Projective Delegation-in-Development and Construction of User-AI Delegation in Discriminative AI-based Systems

4.2.1 Projective Delegation-in-Development

In contrast to the iterational development mode, which emphasized aligning the logic and behavior of the AI-based system with existing domain user practices, the *projective delegation-in-development* was characterized by a forward-looking stance. Developers began with the assessment that existing work practices were inadequate for the complexity and scale of the task at hand. As one product owner, Isabel, in the developer team of the campaign personalization case explained, “We started from the basic idea: ‘Hey, you’re currently doing this like a watering can, we can help you do this better. And we were able to back that up with the model.’” This *projective appraisal* of the superior capability of AI-based systems comprised expectations for performance improvements as well as enhancements for customer experience. Alex, developer from the same project, noted, “Customers being able to get personalized service say they trust our company more as a company that cares. This can only be achieved via automation [with AI]. We will never have the resources to deal with customer interactions in a timely manner otherwise.”

This appraisal of AI's capabilities was shared by the leaders of domain user teams who similarly emphasized that existing work practices had reached their limits. In the campaign personalization project case, Michael, the leader of the marketing domain unit, described how the goal of identifying value-creating customer interactions is increasingly difficult to optimize using traditional approaches. He noted: “You can capture that much better with AI and logic in the background that draws on millions of data records than with the typical approach of human experts relying on the business knowledge they’ve built up over the past 20 years. That’s just how it is.” This appraisal of limitations in existing work approaches and the perceived necessity of AI was prevalent in the customer service routing case as well, where an AI-based system for customer ticket routing was developed. Anand, the leader of the domain users, reflected on issues with ticket misallocation in customer service. Existing rule-based systems had known limitations that could not be resolved through incremental adjustments. “We already know there is a high grey area of wrongly allocated tickets. We know that because we have metrics showing that the current [rule-based] algorithm does not work correctly. [...] We need new approaches, and this [AI-based system] would be such a new approach.” The AI-based systems were framed as means to rethink and improve business processes.

As part of the *distribution-in-development*, building on the appraisal that existing user work

practices had reached their limits and that AI enabled new and superior approaches to completing the work, developers adopted an *outcome-based contracting* approach in the projective trajectory, governing the outcome of applying the AI-based system rather than governing how it should reason. Developers, together with users, sought to demonstrate measurable business performance improvements. Michael, leader of domain users in campaign personalization, emphasized this approach: “When you see that the campaign with the old approach had x percent [conversion rate], and the one with the new machine and new logic had y percent, then you get the people”. Isabel, product owner in campaign personalization, reflected the same governing approach: “We ran a lot of calculations up and down and showed why this [AI-based approach] is better.”

To achieve desired outcomes, developers *regulated the learning* of the AI-based system by setting guardrails for the frame conditions of the system’s learning. Developers, together with domain leaders, began by projecting new ways of accomplishing work that could be enabled by the AI-based systems. In campaign personalization, Michael described how AI could fundamentally change campaign personalization by enabling far more granular targeting: “What it [referring to the AI-based system] could significantly improve is that we can now be more flexible in how we address customers because we can create smaller segments.” These emerging capabilities went beyond automating existing practices, pointing instead to new operational possibilities. For example, Michael explained that AI could support entirely new communication channels and reach customer segments that would otherwise remain under addressed: “You are actually reaching customers who don’t want to talk on the phone, who say, ‘I’ll look at it and decide.’ You are offering them a channel they didn’t know before. We are now bringing this into the app, which will generate even more interaction than on our online customer center. You will reach customer segments you never served before.” The future orientation and re-imagination of work processes were also evident in how developers and domain leaders envisioned combining previously disconnected data sources as input to the AI-based system to enable new business actions in campaign personalization. Rather than treating customer touchpoints such as online shopping carts, mobile apps, and customer service calls as isolated events, they projected a system in which AI could process data across these interactions to support more integrated and personalized customer interaction. A leader of the developer team (Pavel, project A) illustrated this by reflecting a current pain point: customers who begin a mobile contract renewal online in combination with a new end device but abandon the process at the last step. In today’s system, he explained, “no one mentions the renewal or the selected end device” in subsequent customer interactions, whether in a service call or follow-up email. As he put it, “We are leaving a lot on the table, especially in the interplay between online and offline.” The AI-based system was envisioned as an enabler of reconfiguring the entire customer communication logic to be more integrated, timely, and attuned to individual customer behavior and needs. These examples show how developers and domain leaders jointly imagined novel pathways for accomplishing work that could be realized through AI-based systems and how they provided and integrated new data

sources previously underused as system endowments for learning by the AI-based systems.

According to their appraisal that the patterns learned by the AI-based system could outperform existing user work approaches, developers in the projective mode prioritized the system's autonomous learning capability. Optimization for model performance was a key design criterion. Developers usually tested multiple machine learning methods, ultimately selecting the model with the optimal level of performance. This often resulted in the selection of complex, high-performing models, even when these were less interpretable. Carlos, developer in campaign personalization, explained, "The highest quality came from the neural network. We also tried five or six other methods, including ensemble approaches. But the neural network beats everything." Isabel, product owner in the same project confirmed: "The neural network was the best. We compared the top five percent of scored customers and looked at their conversion rates, it was simply far better."

Another frame condition the developers regulated was improving the ground truth data the system learned from, given that the AI-based system's decision logic would emerge directly from the data. This included sensitizing users to the importance of accurate data entry and involving them in systematic data quality checks. As one developer, Nicolas, from the project case customer service routing recalled, service agents often used incorrect problem codes without realizing the downstream effects on model performance: "They weren't aware of the consequences when using a wrong code, and that messes up all models based on that data." Data quality issues were compounded by fragmented systems and inconsistent documentation. Developers worked with users to improve data recording practices, introduced standard quality checks, and made responsibilities for data quality more explicit.

For the *coordination-in-development*, developers followed an approach oriented towards the *coordination of the outcome alignment*. In projects A and E, which both followed a projective development approach, experiments measuring the business impact were conducted to ensure that the AI-based approach was in fact superior in achieving business goals. In campaign personalization, where the focus was on personalizing customer communication and campaigns, the key business metric was the conversion rate. As developer Alex described: "We will have AB testing to see if customers prefer the personalized approach over the classic one." In customer service routing, where the system predicted whether to allocate customer requests to the technical customer support team or the field service, the key business value lay in reducing misallocations and avoiding additional routing of customer requests and unnecessary field visits, each of which incurred both costs and additional issue resolution time. This was also tested through AB experiments and tracked via established business KPIs.

The coordination using business metrics was complemented by user validation of outputs. Domain users were involved only after the AI-based system reached a certain level of performance. As John (PO, project E) explained, "It would be a waste of effort if we didn't first produce okay results. Before jumping into telling them [referring to domain users] how it should look, it's

better to start with something that has some value and then you can share it.” Ralf, developer in the same project, acknowledged that the developer team “actually spoke very little with the actual users [...] instead more with the service innovation department, several levels away from the service agents”. When domain users were engaged, their involvement centered on checking whether predictions were correct in practice rather than retracing the system’s decision logic. Nicholas described: “The model is running, the predictions are there. Now they are in the phase where field service employees take predictions and test them on site”. In this mode, user validation focused on verifying the correctness of the outputs of the AI-based system. John from project case customer service routing described systematically comparing the model’s predictions with human decisions to identify cases where “the model was making the correct decision and the human agent was wrong,” in order to assess “the potential impact that the model can have”. From the user side, the leader of the domain user team, Anand, in customer service routing, described this as a “reality check” involving several service domain specialists who regularly reviewed a number of ticket classifications from the AI-based system each week. While some tickets could be assessed in minutes, others required a full day of technical investigation, sometimes involving additional data sources that were not part of the model’s initial input.

When outcomes deviated from expectations, adjustments to the AI-based system were made in form of adapting the model training conditions. In customer service routing, users reviewed the system’s recommendations of whether a customer issue should be routed to the field service or resolved remotely to identify misclassifications. In case of misclassifications, data labels were corrected and fed back into the model for retraining, or additional data sources were integrated. Nicolas, developer in project case customer service routing noted: “This feedback loop is very important for us. If our model predicts for example ‘cat’ and the human says ‘dog’, we check why the model decided so, why the human decided differently, and then we correct the label for retraining.” Anand, leader of the domain users in customer service routing, pointed out that depending on the results achieved, they would consider “whether we can make the algorithm even sharper” by investing in additional data sources to improve the learning of the AI-based system.

In addition, as part of the coordination activities during development, developers sought to strengthen their confidence in the AI-based system’s decision logic through explainability methods. In campaign personalization, the product owner, Isabel, described how the team reviewed model explanations together, comparing them with their own hypotheses: “Sometimes it was clear, sometimes questionable, but it sparked discussion, helped us get into the topic, [...] and built trust in our own work. That’s very important.” Interpretability toward users, however, was not a priority. Due to the goal of achieving measurable outcome improvements with the AI-based systems, developers focused on demonstrating measurable performance improvements toward users. Accordingly, Carlos (project A) noted that “understanding [model reasoning] is a nice feature, but it’s not essential”. Explainability features, when introduced to the user domain,

were typically aimed at convincing senior stakeholders rather than supporting everyday user understanding. As a developer in the customer service routing case (Nicholas) explained, “The stakeholders didn’t directly define it [...] but the more transparency and clarity we can create, the easier it is to convince people. [...] Through discussions we came to the decision to actually provide explainability, for persuasion, for transparency, but it was not a requirement.”

4.2.2 Projective Construction of User-AI Delegation

The projective mode of development had distinct implications for how delegation between users and AI-based systems was constructed. Since governance was constructed around demonstrating outcome improvements, interpretability by users was not a design priority. Users were only involved in validating outputs using own means and data to verify whether they would have arrived at the same predicted result as the AI-based system. No further transparency into the reasoning process of the AI-based system was provided to the users. Additionally, throughout the development, the AI’s decision logic was not modified to fit with users’ decision logic. Consequently, the systems provided outputs with no additional information explaining the outputs to the users during use time.

The coordination of the actions of the AI-based system during use drew on monitoring and coordination based on outcome. *User coordination was unsupported by design* so that no user-side levers for user coordination during use are designed into the system. Instead, outcome monitoring focused on centralized reporting of business metrics. In the projective development mode, the system’s decision logic emerged without close alignment to domain users’ reasoning. As a consequence, ongoing monitoring of the AI-based system was an analytical task handled centrally by developers rather than by domain users. In customer service routing, Anand, leader of the domain users, described that the AI-based system could be checked automatically without involving any domain users: “I don’t need to put an expert on it. I have an analyst team who sees the data [...] and can calculate if I have a higher occurrence on a certain metric”. He added that in operational use, “the ticket would just be allocated” without exposing the underlying reasoning to domain users.

4.3 Practical-Evaluative Delegation-in-Development and Construction of User-AI Delegation in Generative AI-based Systems

The project cases of generative AI-based systems examined in our study employed large language models (LLMs) to support domain users’ work. Specifically, the cases include translating legacy code into modern programming languages to assist software engineers with code migration, generating summaries of incident-related information to support IT specialists in incident resolution, and identifying and summarizing main topics from open-text customer feedback to help product managers improve customer experience. Across these generative AI cases, the

system development was following a *practical-evaluative* present-oriented stance: developers continuously judged what works here and now, and adjusted the systems accordingly. LLM decision models came pre-specified, and in all generative AI-based system cases, developers accepted the opacity of the decision models as a design condition.

4.3.1 Practical-Evaluative Delegation-in-Development

In the generative AI project cases, developers first engaged in the *appraisal-in-development* and appraised LLMs as offering novel capabilities that could support users and accelerate them in their existing tasks by reducing manual effort and improving the quality of work. In the code migration case, this was evident in developers seeking to automate the tedious, time-consuming translation of legacy code so developers could focus on higher-value work: “we automate this entire process [...] so developers can complete the task faster and return to the tasks they enjoy” (Sarah, developer, project F).

In the incident resolution project case, developers emphasized how LLMs could automate analytical steps that previously required extensive manual inspection. As one developer explained (Nuri, project G): “In the past, you would click through collections and manually search for problems across applications. [...] But an LLM can do that by itself if instructed.” Similarly, developers in the code migration case judged that LLMs could accelerate developers’ work of legacy code migration: “[...] how to solve this with LLMs, not completely automated, but simply to speed up the translation process for the developers” (Sarah, developer, project F). In the customer insights case, developers contrasted earlier Natural Language Processing (NLP) methods, such as taxonomy- and keyword-based categorizations, lexicons curated by human specialists, and supervised classifiers trained on labeled examples, with an LLM’s capacity to surface themes and summarize without predefined categories: “Clustering, topic detection, and summarizing now work with LLMs [...] we have a ‘game changer’” (Sarah, developer, project H). As the product owner put it, the new capabilities of LLMs would improve the quality of analysis: “With LLMs it became clear: we can do much more. We can analyze much better. That’s why we started the project” (Eve, PO, project H). Together, the developers’ appraisal in these generative AI cases showed how developers judged the LLM capabilities as enablers to improve efficiency and quality of the users’ existing work tasks.

Within their appraisal, developers judged that the compatibility of the generative AI-based systems with the human task completion and decision processes must be demonstrated and established in the situated user-AI interaction. Accordingly, they called for environments, in which users could interact with the system: “We need to test that over time. We proposed a solution testing environment, so that they can play with it” and to develop judgment in the interaction: “try to play with it [...] provide a question about what the answer [of the system] is and what you are not satisfied with” (Xiao, developer, project G). The product owner stressed the im-

portance of the users interacting with the LLM-based systems as well: “[...] people are now touching this LLM so they can really see what LLM means” (Amir, PO, project G). Consistent with this, developers in the code migration project enabled users with hands-on interactions with the LLM-based system via a PoC: “[...] we started a PoC. After about one and a half months, we saw it worked” (Sarah, developer, project F).

During the *distribution-in-development*, developers in this practical-evaluative mode of development engaged in the *input-output contracting* of the generative AI-based systems at the instance level through governing the input-output fit. Each interaction with the system should yield an output that users evaluated as correct or useful enough to advance the user task. In the code migration case, developers aimed at a system that produced outputs users would find helpful: “[...] when the users say, ‘It’s not error-free, what the model produces.’ But the code was generated so quickly that it was already helpful, despite the need to adjust two mismatched lines. Because if it then worked, we still saved half an hour. That alone would be progress” (Sarah, developer, project F). In the incident resolution case, where incorrect system outputs would prolong incident resolution time, the product owner targeted an even higher level of correctness in the outputs: “We need to see the reactions and feedback from the users. We would like to receive positive feedback on every answer, actually. It’s a hundred percent of correctness that we would like to have. We need to provide exact information to the end users” (Amir, PO, project G).

To regulate how the generative AI-based system produced outputs that would meet user expectations, developers centered on *regulating the user-AI interaction* and applied several levers. First, the developers regulated the quality standards for the user-AI interaction to ensure goal clarity by pre-structuring goal inputs to the system through pre-defined fixed system prompts, role definitions, and the design of workflows. Across all three generative AI-based project cases, the formulation of the pre-structured goals was not left to the individual users. In the code migration project, developers encoded default engineering workflow into the generative AI-based system to anchor the interaction with the system: “We first tried to understand the legacy code and described it in natural language [using the LLM], then searched in our frameworks of the target programming language for functions that cover the described behavior and based on that generated the target code. [...] With the default workflows, we give them [the users] the feeling that this works. If we didn’t provide this to them and they started defining workflows themselves, I can imagine they would come back after a week and complain” (Sarah, developer, project F). In the incident resolution project case, developers designed fixed system prompts to guide the generative AI-based system to comply with its expected role: “We don’t leave prompt engineering to the users because this is a defined use case. The system should act as a lead incident manager. [...] We are writing roles for this use case. [The system] knows how it should behave. It knows that when it’s using the provided data sources, it is using that from the investigation of the incident perspective. [...] we can exactly set the system to behave like this.” (Amir, PO,

project G). To address variability in user skill in interacting with the generative AI-based system and ensure a baseline standard of output quality, developers argued for standardization of goal input before user adaptations: “You need to have a standard [...] The solution developed by us brings some standardization. When the users interact with our solution, they will get a standard quality in the outputs. If I let the users play with the LLM [without pre-structured goal inputs], he might say, ‘I didn’t get what I wanted.’ Based on the standard, they [the users] can experiment” (Xiao, developer, project G). In the customer insights case, developers designed fixed non-editable system prompts as well to stabilize goal input and intent, and safeguarding the resulting output quality: “Based on my experience, with completely open prompts the quality of the output suffers. [...] If [the users] don’t get where they want to, and that’s what happens with their own prompts, the tool won’t be used. [...] In the background is our system prompt. We won’t make it adjustable, it will stay as it is” (Luis, developer, project H).

Second, developers grounded user-AI interaction in domain- and task-relevant context by supplying the generative AI-based system with task-specific endowments drawn from organizational knowledge bases. In the code migration project, developers provided the LLM with the legacy language documentation to enrich the translation context, and provided it with the custom framework in the target language so the system could reuse existing functions and classes rather than generating generic code. In the incident resolution project, developers integrated incident-related application knowledge documentation as the LLM’s context: “Once we have the application and time range [of the incident], we collect related business objects [...] this is the context we feed to the LLM, which then analyzes incidents, changes and other data for that timeframe” (Mohammed, developer, project G). Developers characterized this work as contextualization, ensuring that each model response considers appropriate organizational knowledge.

Developers pre-structured both goal inputs and endowments in close alignment with users by considering the users’ workflow logic and the endowments users rely on in the design of the interaction with the generative AI-based system. In legacy code migration, developers mirrored engineers’ practice into the interaction workflow with the system – first understand the code in the source language and describe it in natural language, then search the target language frameworks for functions that implement the described behavior, and in the last step, generate code in the target language: “We were able to translate the code [using LLM] by simply asking: ‘How would we proceed as humans?’” In the incident resolution case, developers held direct and regular discussions with lead incident managers to capture “what they look at first, second, and from which data sources,” and transferred this process into the goal and endowment refinement so the generative AI-based system could “mimic the thinking of the incident manager [...] reflect it in the solution.” Concretely, rather than a simple prompt, the developer developed a chain-of-thought to align with the steps the incident managers took in investigating an incident: “how the incident manager operates [...] First, he checks the incident in the problem ticket. Then he checks the root cause analysis. I’ll create this chain. It’s quite specific. I’m tailoring the solution

[...]” (Xiao, developer, project G).

Building on the pre-structured goal inputs and task-specific endowments, developers enabled users to tailor the interaction with the generative AI-based system themselves to meet their situated needs. Users could both refine intent by clarifying what the system should do next, and they could supply local endowments by adding their own additional context information. In the code migration case, software developers as users of the generative AI-based code migration system could add software packages and frameworks that ensured alignment with their own project standards in the output generation. In the incident resolution case, the system was primed for a lead incident manager role, yet users could still regulate the queries as needs emerged: “It doesn’t mean they [the users] cannot obtain different information. The users can place different kinds of questions, and the system will be able to recognize that it should provide different kinds of answers” (Amir, PO, project G). In the customer insights case, on top of the standard fixed goal inputs pre-defined by developers, they designed the goal input to explicitly require further dynamic user input in the interaction to shape output generation. The system “asks the user: please describe your interview [data with end customers] [...] provide one or several example texts [...] What is your expected topic for this text? That’s the few-shot prompting, and we provide [users] help for it” (Luis, developer, project H).

The *coordination-in-development* of the system behavior was oriented toward *coordinating the input-output fit* on the instance level, the system’s internal model decision logic was not a coordination object. Instead, the opacity of the LLM’s decision logic was accepted as a condition in the development. A developer stated, “You just need to understand its inputs and outputs. [...] You cannot know each and every detail, each and every component of the system” (Victor, developer, project F). Others echoed the stance that internal workings were secondary to results: “How it works under the hood is not really important” (Kenji, user, project F). “We are treating it as a black box despite us adding tools to it” (Jacob, user, project F). Instead, the focus was on whether outputs reliably met task needs: “My interest in delving deeper [into the LLM] is limited. As long as the actual result is good, why should I scrutinize it any further?” (Leo, PO, project H).

In line with this instance-level input and output focus, developers first implemented output validation via technical gates. In the code migration project, developers treated successful source code execution as a first gate: “The requirement was the quality of the generated code. [...] first see that it’s at least valid code that runs in the target programming language environment” (Victor, developer, project F). Reflection loops were used that re-ran generated code and returned errors for self-correction by the LLM: “We can run it again. [...] If, for example, we have a syntax error, with the reflection part, you can reduce the number of times the model writes invalid code [in the target language]. [...] We can let the model run its own code, recognize the errors and fix them” (Kenji, user, project F). In the incident resolution case, developers similarly sent outputs back to the LLM for a consistency check, “ask if this answer makes sense” (Mohammed,

developer, project G).

The technical output evaluation was complemented by user evaluation of outputs in all three generative AI-based project cases. This user evaluation was highly iterative and interwoven with the regulative development practices. Developers iterated with users the inputs and the resulting outputs, adapted the regulation of the generative AI-based system, until the users found the outputs to be adequate and fitting to their needs. In the project cases, user evaluation took the form of manual inspection of generated code by software engineers in the code migration case, onboarding domain user evaluators for evaluating the system outputs in the incident resolution case, and user review of customer feedback categorization in the customer insights case. Developers emphasized that user evaluation on output level was always necessary to assess whether the outputs were useful and accurate in the domain context: “It is always necessary that a human looks at it again and says: does the quality fit as it is?” (Leo, PO, project H).

When outputs failed to meet user needs in the interaction, developers iteratively adapted the inputs, refining the goal input including role descriptions, until per-output fit was achieved based on user evaluation. In the incident resolution case, the developers adapted the role definition as part of the goal input together with lead incident managers, testing whether the LLM met the adequate role and revising it iteratively: “We ran through maybe 20 different descriptions of this role [...] because we were not receiving the outputs that we were expecting” (Amir, PO, project G). Another need of the incident managers as users was for the output answers to be both relevant and concise due to time pressure: “there’s a lot of information [in the system outputs], although we need less but important info [...] we work under stress and don’t have enough time” (Patrick, user, project G). In response, developers implemented changes to the generative AI-based system and incorporated brevity and keywords into the goal input instantly during the sessions with users: “I named the keywords important to me. They [the developers] programmed it live so that in the next version the information not needed disappeared” (Patrick, user, project G). Users’ real-time evaluations thus directly re-shaped goal inputs, aligning outputs with users’ situated decision needs. Beyond in-session updates, developers logged the user interactions, such as queries or goal refinements, and conducted post-hoc analysis for improvements of the goal input to the system: “[...] the lead incident manager can ask questions regarding the output displayed. They are collected. [...] We are doing the logging and we see what the lead incident manager asks about. Over time, we can do some analytics of those questions, and we can integrate those questions into the prompts. Again, to improve the general outputs for the user” (Xiao, developer, project G). In the customer insights case, developers and users iterated the topic schema and its linkage to organizational terminology, iterating together with users “What is a category? [...] when is a topic a topic?” and how far emergent themes from the generative AI-based system should be aligned to company-internal terms (Leo, PO, project H). These cycles “jumped between fully automatically generated topics and predefined topics. [...] The end users signed off that the [generated] categories were acceptable for them and workable. We

iterated through these cycles for several months” (Luis, developer, project H). Ultimately, after these iterative evaluations and adaptations, the solution converged on a hybrid approach: a persistent category map to stabilize outputs while leaving room for new themes (Luis, developer, project H).

4.3.2 Practical-Evaluative Construction of User-AI Delegation

In the practical-evaluative mode of development, *user appraisal during use was constructed by providing instance-level input-output transparency* in use. Users could obtain additional information on the inputs and outputs in each interaction. Concretely, incident managers could interrogate the outputs shown with targeted questions: “the lead incident manager can ask questions [...] about the info displayed” (Xiao, developer, project G). In the customer insights case, user appraisal was supported through a guided chat that was “not completely free but works through follow-up prompts and question-answer interactions” (Luis, developer, project H). In addition, the outputs in the customer insights case could be inspected by the users on the instance-level via hierarchical drill-downs: “[as user] I get the texts displayed in the tool in their hierarchical clusters, that are category, topics, and verbatims. The texts can be viewed at every level, and for each individual text I can see which category it falls into. [...] This quality cross-check ‘does what happened here make sense?’ [...] is something users can do very well themselves” (Luis, developer, project H). Through these means, users were informed in their appraisal in situ in their interaction with the generative AI-based system.

User coordination in the practical-evaluative mode was *designed into the interaction* with the system at the instance level during use. Each interaction supported per-output coordination of users by providing them with the ability to evaluate and provide feedback on the output and to adjust goals and endowments during the interaction.

In the incident resolution case, this meant users could signal satisfaction per answer during the interaction: “[...] likes/dislikes are implemented here [...]. It is a very easy and very effective KPI and measurement, which can be implemented. [...] you can have direct feedback from the user whether this information was the right one” (Amir, PO, project G). This continuous user monitoring was used to monitor and further improve the effectiveness of the generative AI-based system for the users.

In the code migration case, users could not only engage in monitoring but also adapt the output generation and re-generate outputs by adjusting goals and endowments as input to the AI-based system. Starting from default prompts and a default translation flow, “the users can copy this workflow and then append further agents or tasks [...] for example, after code generation, add a task to write unit and integration tests, or to write the documentation [...] or assemble a complete workflow themselves” (Sarah, developer, project F). Through these per-output feedback and user adjustments in the interaction, users were enabled for coordination in use that kept outputs

aligned with users' situated standards and needs.

5 Discussion

Our findings reveal two coupled delegation processes. Beyond the user-side delegation at use time theorized by Baird and Maruping (2021) and framed as a dyadic user-AI relationship, we show that developers delegate during development as well. We theorize this process as *delegation in development*: Developers transfer a part of their traditional responsibility for constructing the system's decision logic to the AI-based system, based on their *appraisal-in-development*, which involves an appraisal of the user capabilities, AI capabilities, and their compatibility. Developers then govern their responsibility transfer through *distribution-in-development*, regulating the delegation ex ante, and through *coordination-in-development*, controlling and monitoring the AI-based system ex post. These mechanisms ensure that outputs and actions of the AI-based system align with desired goals. During delegation in development, developers configure how users will be able to appraise and coordinate the AI-based system at use time.

With our study, we extend the dyad at use time to a triad of developers, AI, and users across development and use, with delegation-in-development shaping delegation-in-use. Interestingly, the temporal orientation (iterational, projective, or practical-evaluative) (Emirbayer Mische, 1998) that developers adopt during delegation-in development is mirrored at use time during user-AI delegation: iterational delegation-in-development yields iterational conditions for user-AI appraisal and coordination, projective yields projective, and practical-evaluative yields practical-evaluative. Figure 2 synthesizes the interplay between the processes of delegation-in-development and delegation-in-use.

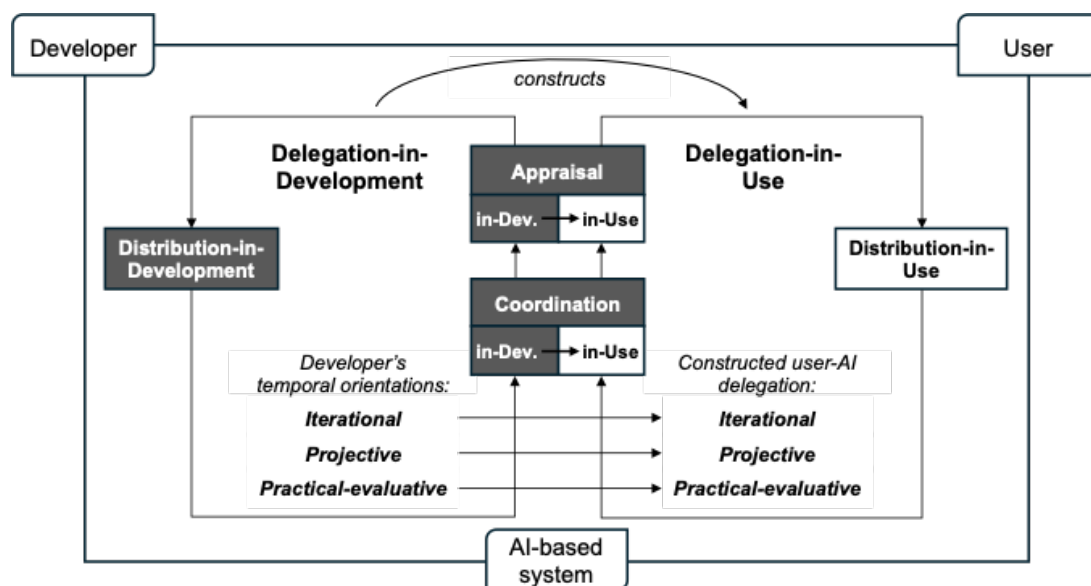


Figure 2: Model of two delegation cycles in the development of AI-based systems: Delegation-in-development & delegation-in-use

Thus, we demonstrate that the developers' temporal orientation (iterational, projective, practical-evaluative) is the key differentiator in delegation-in-development. Furthermore, the delegation-in-development in a specific orientation constructs the user-AI delegation through the corresponding orientation. This orientation determines what and how much is delegated, how it is governed, and where and to what extent user involvement is placed across development and use. Table 3 summarizes how each orientation determines appraisal-, distribution-, and coordination-in-development, while Table 4 shows how these choices materialize for users and shape the subsequent user-AI delegation in use. In an iterational orientation, developers delegate narrow parts of the decision logic construction and maintain tight control loops with ongoing transparency and interpretability. This constructs a corresponding user appraisal and coordination that is iterationally oriented to align the AI's decision logic with established user work practices proven to be effective in the past. In a projective orientation, developers delegate more expansively and steer primarily by outcomes. They endow the AI-based system with broader learning autonomy and judge the AI-based system against business metrics. This constructs a projective appraisal and coordination at use-time oriented toward evaluating the performance of the projected, new way of work enabled by the AI-based system. In a practical-evaluative orientation, developers delegate flexibly around instance-level fit. They emphasize input-output traceability, and design for situated adjustments in the moment of interaction. This constructs practical-evaluative appraisal and coordination for users at the instance-level of the user-AI interaction.

Across the three orientations, what varies is how much responsibility is transferred to the AI-based system, where governance is anchored (check of decision logic versus outcome measurement versus interactional fit) and how this constructs user appraisal and coordination in the corresponding orientation for the user-AI delegation at use time. In sum, developers' delegation-in-development sets the conditions under which use-time delegation unfolds, and the orientation of the constructed user-AI delegation corresponds to the temporal orientation during delegation-in-development.

5.1 Theoretical Implications

This research contributes to the literature on *IS delegation*, *algorithmic learning*, and *information systems development (ISD)* for AI-based systems.

First, we extend IS delegation theory (Baird & Maruping, 2021) by conceptualizing delegation as a triadic process that links developers, users, and AI-based systems across development and use. Prior research predominantly treats delegation as a dyadic, use-time phenomenon between users and AI (e.g., Baird & Maruping, 2021; Jussupow et al., 2021; Lebovitz et al., 2022; Zhang et al., 2021). Our findings reveal that delegation begins earlier, during development, as developers themselves enter a delegation relationship with the AI-based system. Through this developer-AI delegation during development, developers construct the AI system's capa-

Table 3: Delegation-in-development per developers' temporal orientation: summary of concepts and definitions

Concept	Iterational	Projective	Pactical-Evaluative
Appraisal-in-Development	<i>Developers' appraisal from a temporal stance from which they appraise user and AI capabilities for task completion and the compatibility between users' and AI's decision-making processes.</i>		
	Iterationally oriented Appraisal	Projectively oriented Appraisal	Practical-Evaluatively oriented Appraisal
	Iterationally oriented appraisal is oriented toward prioritizing users' work practices and the alignment of AI's compatibility with them.	Projectively oriented appraisal is oriented toward prioritizing AI's superior capabilities to envision new ways of work.	Practical-evaluatively oriented appraisal is oriented toward situated evaluation of the AI-based system and compatibility in the user-AI interaction.
Distribution-in-Development: Contracting	<i>For the transfer of developers' responsibilities of the decision model construction to AI, developers establish desired goals and how to achieve them ex ante to govern this transfer and be able to identify misalignments.</i>		
	Behavioral Contracting	Outcome Contracting	Input-Output Contracting
	Governance of how AI makes decisions to align AI's decision process with users' decision process.	Governance of what must be achieved by AI to align with desired business outcomes.	Governance of instance-level input-output fit.
Distribution-in-Development: Regulating	<i>For the transfer of developers' responsibilities of the decision model construction to AI, developers regulate the possible actions of the AI-based system by setting boundaries.</i>		
	Regulating Decision Logic	Regulating Frame Conditions for Learning	Regulating User-AI Interaction
	Regulating by encoding known patterns and by constraining model settings based on user practices.	Regulating by optimizing input and model performance for autonomous learning.	Regulating by pre-structuring user-AI interaction with room for situated user adjustments.
Coordination-in-Development	<i>Developers' coordination activities during development to monitor and update the AI-based systems for alignment with the desired goal defined by developers for the responsibility transfer.</i>		
	Coordination of Decision Logic Alignment	Coordination of Outcome Alignment	Coordination of Input-Output Fit
	Inspection and update of decision logic enabled by interpretable models.	Inspection of outcomes of task completion using business metrics.	Evaluation of input-output fit at the instance level of the user-AI interaction.

Table 4: Construction of user-AI delegation per developers' temporal orientation: summary of concepts and definitions

Concept	Iterational	Projective	Practical-Evaluative
Development-Constructed User Appraisal	<i>Constructed user appraisal during user-AI delegation by delegation-in-development.</i>		
	Informed by Transparency of Decision Logic	Uninformed	Informed by Input-Output Transparency
	Transparency on system reasoning at use time with decision logic pre-aligned to user practice in development.	No transparency into the reasoning of the AI-based system, and no alignment of the AI's decision logic to user practice in development.	Transparency on input and output per user-AI interaction.
Development-Constructed User Coordination	<i>Constructed user coordination during user-AI delegation by delegation-in-development.</i>		
	User Coordination by Design	User Coordination Unsupported by Design	Interactional User Coordination by Design
	Built-in coordination for users at use time to review and initiate adjustments of the AI-based system.	No user-side coordination built into the system at use time.	Built-in, instance-level coordination for users at use time to review and adjust per output during the interaction.

bilities, outputs, and coordination mechanisms, which in turn shape the user-AI delegation. We identify three developer orientations that configure the user-AI delegation in different ways. In an iterational delegation approach, developers delegate parts of the decision logic construction to the AI-based system, while simultaneously scrutinizing and controlling the emerging decision logic of the AI-based system to align it with users' established decision logic. By design, this iterational delegation approach embeds transparency, interpretability, and review means that support users' later appraisal and coordination during user-AI delegation. In a projective delegation approach, developers refrain from controlling how the AI-based system arrives at its outputs. Instead, they steer the AI-based system by business outcomes and metrics rather than by internal reasoning. Because coordination is organized around ex post performance, the AI-based system provides limited built-in support for user appraisal and coordination during user-AI delegation. In a practical-evaluative delegation approach, developers do not construct the decision model (they adopt an existing model) but retain control over goals in the user-AI interaction. They govern input-output fit through ongoing, instance-level user evaluation and feedback. This spans development and use, and users are provided with instance-level traceability of inputs and outputs for appraisal and coordination during development and use. Thus, delegation in AI-based systems unfolds as two cycles of appraisal, distribution, and coordination that are linked, spanning development and use, rather than a process at use time alone. This triadic framing advances the delegation perspective in IS by explaining how the conditions for user-AI delegation are constructed through developer-AI delegation.

Second, we contribute to the literature on algorithmic learning in AI-based systems by demon-

strating that the learning capacity of these systems is not merely technologically given, but is shaped by how developers transfer responsibility to the AI-based system and regulate this transfer. Prior work emphasizes that machine learning systems construct their own decision logic from data, resulting in partial opacity and unpredictability in organizational settings (e.g., Burrell, 2016; Faraj et al., 2018; Lebovitz et al., 2022; Seidel et al., 2019; Zhang et al., 2021). We show that how much responsibility developers are willing to hand over to the AI-based system, and thus how broadly and autonomously it is allowed to learn, depends on the developers' temporal orientation (Emirbayer & Mische, 1998). In an *iterational* orientation, developers delegate their responsibility only narrowly for decision logic construction, keep the system's learning constrained, rely on interpretable models, and close user review to maintain control. In a *projective* orientation, they delegate more expansive responsibility for optimizing outcomes of the AI-based system, broaden its learning scope, and govern it primarily through outcome metrics and performance monitoring. In a *practical-evaluative* orientation, particularly with generative models, developers accept that they cannot meaningfully shape the internal decision model and instead involve users in the situated evaluation and correction of individual outputs during the user-AI interaction, while retaining responsibility for designing interaction patterns and retrieval mechanisms. In this way, we demonstrate that the self-learning characteristic of AI-based systems is not an inherent property of the technology but an outcome of developers' delegation to these systems. The extent of autonomous learning capacity of AI-based systems is co-constructed through developers' delegation and regulation practices.

Third, we advance research on ISD for AI-based systems by specifying how development work changes when system behavior cannot be fully specified and inspected. Traditional ISD emphasizes requirements engineering (Browne & Rogich, 2001; Hickey & Davis, 2004) and the specification of rules or decision logic (Herbst, 1996; Ram & Khatri, 2005). In AI-based systems, by contrast, developers hand over parts of the decision logic construction to the AI-based system and then coordinate this delegation rather than solely encoding decision rules. ISD thus shifts from specifying system logic to designing human-AI delegation. We further extend work on human-in-the-loop design in AI-based systems development, which highlights the role of domain expertise (Gronsund & Aanestad, 2020; van den Broek et al., 2021). While existing literature generally recommends involving domain experts, Paper IV refines the notion of human-in-the-loop by showing that its extent and locus differ with developers' temporal orientations in delegation-in-development. In an *iterational* orientation, human-in-the-loop is intensive and continuous across development and use, aimed at aligning the AI's internal decision logic with the users' decision logic. In a *projective* orientation, developers embed greater autonomy in the AI-based system and steer by performance objectives, so that human-in-the-loop becomes limited, focused on providing quality ground truths rather than reviewing AI's internal reasoning. In a *practical-evaluative* orientation, developers do not shape the model's internal decision logic but focus on maintaining input-output fit at the instance level. Here, human-in-the-loop is intensive but localized at the instance level for situated input-output evaluation and adjustments. We

show that the extent of human-in-the-loop ranges from high (iterational and practical-evaluative) to low (projective), while its focus shifts from internal logic alignment (iterational) to outcome monitoring (projective) to instance-level input-output evaluation (practical-evaluative). Accordingly, ISD for AI-based systems is best understood as a developer-AI delegation and coordination, in which the degree and locus of human-in-the-loop are resulting from design choices determined by the developer's temporal orientation.

Together, these insights reshape ISD for AI-based systems from a process of specifying and implementing decision logic into a practice of designing and governing the delegation of responsibility across the triad of developers, users, and the AI-based system.

5.2 Practical Implications

Our theorization of delegation-in-development foregrounds the development phase as critical for organizing the introduction and use of AI-based systems. Unlike traditional information systems, where development centers on implementing specified rules, we show that the core task here is to decide how to delegate to the AI-based system during development. How developers allocate responsibilities to the system and govern them – specifically, whether they adopt an iterational, projective, or practical-evaluative orientation – directly shapes how user-AI delegation can subsequently unfold in use. These design-time choices determine what the AI-based system is authorized to decide, what users can appraise and contest, and which coordination levers are available during use. Accordingly, such choices should not remain within the purview of technical teams alone but be made jointly with key business stakeholders, including users and leaders, so that the organization can intentionally determine the extent and form of delegation to the AI-based system it is willing to operationalize and govern. Making the development orientation explicit, knowing how it configures the conditions of use, enables both *ex ante* alignment of the delegation design and, *ex post*, targeted adjustments when use reveals mismatches between the intended way of delegation and the actual user-AI interaction.

Practice often assumes that the involvement of domain experts is always necessary for building successful AI solutions. Our findings qualify this view by specifying when and where expertise has the greatest effect. Expert input is most valuable when aligning system behavior with established practices or when interpreting ambiguous outputs that depend on professional judgment. It is less central when outcomes can be clearly specified and measured. And it moves toward the interaction layer when the outputs of the AI-based system can be assessed and adjusted at the moment of user-AI interaction. Rather than involving experts by default, our findings suggest that organizations should be aware of how to involve experts effectively.

Finally, our theorization of delegation-in-development shows that managing the uncertainty and probabilistic behavior of the AI-based system is a key concern during development. The regulation and coordination mechanisms established during development carry forward into use

and shape the forms of coordination available in use. For example, if coordination in development hinges on making model reasoning reviewable, users will later coordinate by verifying and adjusting model attributes. By contrast, if coordination is organized around measurable outcome metrics during development, use-time coordination will be managed through performance monitoring rather than inspection of the decision model of the AI-based system. Therefore, organizations should anticipate different coordination needs and specify coordination mechanisms in line with their choices in delegation-in-development.

Taken together, our findings help organizations recognize the importance of the development phase and suggest treating the development of AI-based systems as designing delegation. These delegation decisions during development should be made explicitly, as they both construct the role of the AI-based system and condition what users can appraise, contest, and coordinate in use.

6 Future Research and Conclusion

This study has limitations that open up opportunities for future research. Our analysis foregrounds developers and their practices in constructing delegation with AI-based systems. This responds to the state of existing IS research, which has predominantly examined users' adoption and use of AI-based systems, while developers often remain in the background as technical implementers. The paper extends this literature by detailing how developers confront emerging challenges in development due to the agentic characteristics of AI-based systems, how they construct user-AI delegation, and how they enact development practices in the triad of developers, users, and the AI-based system. Users' perspectives on development-time decisions and conditions are incorporated in less depth. The findings thus offer a particularly detailed account of how developers enact development practices in the triad. Future research could therefore complement these findings with studies of how users experience the delegation conditions constructed during development. Additionally, longitudinal studies that examine AI-based systems from development through use and subsequent redesigns could examine how users' experiences in everyday use feed back into developers' appraisal-, distribution-, and coordination-in-development practices and how these practices are reinforced, adapted or reworked over time. Such research could deepen our understanding of how the triadic delegation practices in AI-based systems evolve across development and use.

This study theorizes delegation in AI-based systems development from a novel triadic perspective that links developers, users, and the AI-based system. Addressing the research question of how developers enact development practices for AI-based systems whose behavior they cannot fully specify or inspect, and how these practices condition how users engage with these systems in use, the study introduces the concept of delegation-in-development. It shows that developers themselves delegate to the AI-based system by handing over parts of their traditional responsi-

bility for decision logic construction to learning and partly opaque systems through appraisal-, distribution-, and coordination-in-development practices. At the same time these practices determine how users can later delegate tasks, appraise outputs, and coordinate with the AI-based system in use. Delegation is thereby conceptualized as two interlinked cycles of delegation, one between developers and the AI-based system during development, and one between users and the AI-based system during use.

A further central contribution of the study is to identify that this dual delegation is shaped by developers' temporal orientations. Iterational, projective, and practical-evaluative orientations give rise to distinct ways of allocating responsibility to the AI-based system and of enabling user-AI delegation. The study demonstrates that developers' delegation choices concerning the AI-based system condition the user-AI delegation in use. In answering the research question, the study conceptualizes the development of AI-based systems as the design and governance of responsibility transfers across the triad of developers, users, and the AI-based systems.

References

- Ågerfalk, P. J. (2020). Artificial intelligence as digital agency. *European Journal of Information Systems*, 29(1), 1–8.
- Alavi, M., Leidner, D. E., & Mousavi, R. (2024). A Knowledge Management Perspective of Generative Artificial Intelligence. *Journal of the Association for Information Systems*, 25(1), 1–12.
- Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmermann, T. (2019a). Software engineering for machine learning: A case study. *IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 291–300.
- Amershi, S., Weld, D., Vorvoreanu, M., Fournery, A., Nushi, B., Collisson, P., Suh, J., Iqbal, S., Bennett, P. N., & Inkpen, K. (2019b). Guidelines for human-AI interaction. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–13.
- Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*.
- Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Benetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, 82–115.
- Asatiani, A., Malo, P., Nagbol, P. R., Penttinen, E., Rinta-Kahila, T., & Salovaara, A. (2021). Sociotechnical Envelopment of Artificial Intelligence: An Approach to Organizational Deployment of Inscrutable Artificial Intelligence Systems. *Journal of the Association for Information Systems*, 22(2), 325–352.
- Auernhammer, J. (2020). Human-centered AI: The role of Human-centered Design Research in the development of AI. *Synergy - DRS International Conference 2020*, 1315–1333.
- Baig, A., Merrill, D., & Sinha, M. (2024). Moving past gen AI's honeymoon phase: Seven hard truths for CIOs to get from pilot to scale. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/moving-past-gen-ais-honeymoon-phase-seven-hard-truths-for-cios-to-get-from-pilot-to-scale>
- Baird, A., & Maruping, L. M. (2021). The Next Generation of Research on IS Use: A Theoretical Framework of Delegation to and from Agentic IS Artifacts. *MIS Quarterly*, 45(1), 315–341.
- Bandura, A. (2006). Toward a psychology of human agency. *Perspectives on Psychological Science*, 1(2), 164–180.
- Beaudry, A., & Pinsonneault, A. (2005). Understanding user responses to information technology: A coping model of user adaptation. *MIS Quarterly*, 29(3), 493–524.
- Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing Artificial Intelligence. *MIS Quarterly*, 45(3), 1433–1450.

- Berente, N., Lyytinen, K., Yoo, Y., & King, J. L. (2016). Routines as shock absorbers during organizational transformation: Integration, control, and NASA's enterprise information system. *Organization Science*, 27(3), 551–572.
- Berg, J. M., Raj, M., & Seamans, R. (2023). Capturing value from artificial intelligence. *Academy of Management Discoveries*, 9(4), 424–428.
- Bojinov, I. (2023). Keep Your AI Projects on Track. <https://hbr.org/2023/11/keep-your-ai-projects-on-track>
- Bostrom, R. P., Gupta, S., & Thomas, D. (2009). A meta-theory for understanding information systems within sociotechnical systems. *Journal of Management Information Systems*, 26(1), 17–48.
- Bostrom, R. P., & Heinen, J. S. (1977b). MIS problems and failures: A socio-technical perspective. Part I: The causes. *MIS Quarterly*, 1(3), 17–32.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., & Askell, A. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
- Browne, G. J., & Rogich, M. B. (2001). An empirical investigation of user requirements elicitation: Comparing the effectiveness of prompting techniques. *Journal of Management Information Systems*, 17(4), 223–249.
- Brynjolfsson, E., Li, D., & Raymond, L. R. (2023). Generative AI at work.
- Burrell, J. (2016). How the Machine ‘Thinks’: Understanding Opacity in Machine Learning Algorithms. *Big Data & Society*, 3(1), 1–12.
- Castelfranchi, C., & Falcone, R. (1998). Towards a theory of delegation for agent-based systems. *Robotics and Autonomous Systems*, 24(3-4), 141–157.
- Chollet, F. (2021). *Deep learning with Python*. Simon; Schuster.
- Chui, M., Hazan, E., Roberts, R., Singla, A., Smaje, K., Sukharevsky, A., Yee, L., & Zimmel, R. (2023a). The economic potential of generative AI: The next productivity frontier. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier>
- Davenport, T. H. (2021). The analytics team. In *Informatics body of knowledge* (pp. 49–76). Wiley.
- Eisenhardt, K. M. (1989a). Agency theory: An assessment and review. *Academy of Management Review*, 14(1), 57–74.
- Eisenhardt, K. M. (1989b). Building theories from case study research. *Academy of Management Review*, 14(4), 532–550.
- Eisenhardt, K. M., & Graebner, M. E. (2007). Theory building from cases: Opportunities and challenges. *Academy of Management Journal*, 50(1), 25–32.
- Emirbayer, M., & Mische, A. (1998). What is agency? *American Journal of Sociology*, 103(4), 962–1023.

- Fadel, K. J., & Brown, S. A. (2010). Information systems appraisal and coping: The role of user perceptions. *Communications of the AIS*, 26(1), 6.
- Faraj, S., Pachidi, S., & Sayegh, K. (2018). Working and organizing in the age of the learning algorithm. *Information and Organization*, 28(1), 62–70.
- Forsythe, D. E. (1993). Engineering knowledge: The construction of knowledge in artificial intelligence. *Social Studies of Science*, 23(3), 445–477.
- Foster, D. (2022). *Generative Deep Learning*. O'Reilly Media, Inc.
- Gallivan, M. J., & Keil, M. (2003). The user–developer communication process: a critical case study. *Information Systems Journal*, 13(1), 37–68.
- Gioia, D. A., Corley, K. G., & Hamilton, A. L. (2013). Seeking qualitative rigor in inductive research: Notes on the Gioia methodology. *Organizational Research Methods*, 16(1), 15–31.
- Glaser, V. L., Pollock, N., & D'Adderio, L. (2021). The biography of an algorithm: Performing algorithmic technologies in organizations. *Organization Theory*, 2(2), 1–27.
- Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1). MIT press Cambridge.
- Gregor, S., & Benbasat, I. (1999). Explanations from intelligent systems: Theoretical foundations and implications for practice. *MIS Quarterly*, 23(4), 497–530.
- Gronlund, T., & Aanestad, M. (2020). Augmenting the algorithm: Emerging human-in-the-loop work configurations. *Journal of Strategic Information Systems*, 29(2), 1–16.
- Gulati, R., Wohlgezogen, F., & Zhelyazkov, P. (2012). The two facets of collaboration: Cooperation and coordination in strategic alliances. *Academy of Management Annals*, 6(1), 531–583.
- Hadfield-Menell, D., & Hadfield, G. K. (2019). Incomplete contracting and AI alignment. *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 417–422.
- Hanseth, O., & Monteiro, E. (1997). Inscribing behaviour in information infrastructure standards. *Accounting, Management and Information Technologies*, 7(4), 183–211.
- Hayes-Roth, F. (1997). Artificial intelligence: what works and what doesn't? *AI Magazine*, 18(2), 99–113.
- Hayes-Roth, F., & Jacobstein, N. (1994). The state of knowledge-based systems. *Communications of the ACM*, 37(3), 26–39.
- He, J., & King, W. R. (2008). The role of user participation in information systems development: Implications from a meta-analysis. *Journal of Management Information Systems*, 25(1), 301–331.
- Henfridsson, O., & Lindgren, R. (2010). User involvement in developing mobile and temporarily interconnected systems. *Information Systems Journal*, 20(2), 119–135.
- Herbst, H. (1996). Business rules in systems analysis: a meta-model and repository system. *Information Systems*, 21(2), 147–166.

- Hickey, A. M., & Davis, A. M. (2004). A unified model of requirements elicitation. *Journal of Management Information Systems*, 20(4), 65–84.
- Hunton, J. E., & Beeler, J. D. (1997). Effects of user participation in systems development: a longitudinal field experiment. *MIS Quarterly*, 21(4), 359–388.
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An introduction to statistical learning - second edition*. Springer.
- Jussupow, E., Spohrer, K., Heinzl, A., & Gawlitza, J. (2021). Augmenting medical diagnosis decisions? An investigation into physicians' decision-making process with artificial intelligence. *Information Systems Research*, 32(3), 713–735.
- Kane, G. C., Young, A. G., Majchrzak, A., & Ransbotham, S. (2021). Avoiding an oppressive future of machine learning: A design theory for emancipatory assistants. *MIS Quarterly*, 45(1), 371–396.
- Kirsch, L. J. (1996). The management of complex tasks in organizations: Controlling the systems development process. *Organization Science*, 7(1), 1–21.
- Kling, R. (1977). The organizational context of user-centered software designs. *MIS Quarterly*, 1(4), 41–52.
- Kozar, K. A., & Mahlum, J. M. (1987). A user generated information system: an innovative development approach. *MIS Quarterly*, 11(2), 163–174.
- Laato, S., Mäntymäki, M., Minkinen, M., Birkstedt, T., Islam, A. K. M., & Dennehy, D. (2022). Integrating machine learning with software development lifecycles: Insights from experts. *European Conference on Information Systems (ECIS) Proceedings*.
- Lebovitz, S., Lifshitz-Assaf, H., & Levina, N. (2022). To engage or not to engage with AI for critical judgments: How professionals deal with opacity when using AI for medical diagnosis. *Organization Science*, 33(1), 126–148.
- Lee, G., & Xia, W. (2010). Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. *MIS Quarterly*, 34(1), 87–114.
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., & Rocktäschel, T. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459–9474.
- Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9), 1–35.
- Malone, T. W., & Crowston, K. (1994). The interdisciplinary study of coordination. *ACM Computing Surveys (CSUR)*, 26(1), 87–119.
- Maruping, L. M., & Matook, S. (2020). The Multiplex Nature of the Customer Representative Role in Agile Information Systems Development. *MIS Quarterly*, 44(3), 1411–1437.
- Mayer, H., Yee, L., Chui, M., & Roberts, R. (2025b). Superagency in the Workplace - Empowering people to unlock AI's full potential. <https://www.mckinsey.com/capabilities/>

- mckinsey-digital/our-insights/superagency-in-the-workplace-empowering-people-to-unlock-ais-full-potential-at-work#/
 Murray, A., Rhymer, J., & Sirmon, D. G. (2021). Humans and technology: Forms of conjoined agency in organizations. *Academy of Management Review*, 46(3), 552–571.
- Naumann, J. D., & Jenkins, A. M. (1982). Prototyping: the new paradigm for systems development. *MIS Quarterly*, 6(3), 29–44.
- Okhuysen, G. A., & Bechky, B. A. (2009). 10 coordination in organizations: An integrative perspective. *Academy of Management Annals*, 3(1), 463–502.
- Ozkaya, I. (2020). What is really different in engineering AI-enabled systems? *IEEE Software*, 37(4), 3–6.
- Patil, D. J. (2011). *Building Data Science Teams*. O'Reilly Media, Inc.
- Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of AI on developer productivity: Evidence from GitHub Copilot. *arXiv preprint arXiv:2302.06590*.
- Pinski, M., Adam, M., & Benlian, A. (2023). AI knowledge: Improving AI delegation through human enablement. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 1–17.
- Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*. San Francisco, CA, USA.
- Raisch, S., & Krakowski, S. (2021). Artificial Intelligence and Management: The Automation-Augmentation Paradox. *Academy of Management Review*, 46(1), 192–210.
- Raisch, S., & Fomina, K. (2025). Combining human and artificial intelligence: Hybrid problem-solving in organizations. *Academy of Management Review*, 50(2), 441–464.
- Ram, S., & Khatri, V. (2005). A comprehensive framework for modeling set-based business rules during conceptual database design. *Information Systems*, 30(2), 89–118.
- Rosenkranz, C., Stray, V., & Wiesche, M. (2025). Navigating the new frontier of information systems engineering. *Business & Information Systems Engineering*, 67(1), 1–5.
- Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Sauvestre, R., & Remez, T. (2023). Code Llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.
- Russell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach*. Pearson.
- Saffarizadeh, K., Keil, M., & Maruping, L. (2024). Relationship Between Trust in the Artificial Intelligence Creator and Trust in Artificial Intelligence Systems: The Crucial Role of Artificial Intelligence Alignment and Steerability. *Journal of Management Information Systems*, 41(3), 645–681.
- Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in machine learning systems. *Advances in Neural Information Processing Systems*, 28.

- Seidel, S., Berente, N., Lindberg, A., Lyytinen, K., & Nickerson, J. V. (2019). Autonomous tools and design: a triple-loop approach to human-machine learning. *Communications of the ACM*, 62(1), 50–57.
- Strauss, A., & Corbin, J. (1998). *Basics of qualitative research techniques*. Sage Publications.
- Strich, F., Mayer, A.-S., & Fiedler, M. (2021). What do I do in a world of artificial intelligence? Investigating the impact of substitutive decision-making AI systems on employees' professional role identity. *Journal of the Association for Information Systems*, 22(2), 9.
- Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., & Müller, K.-R. (2021). Towards CRISP-ML (Q): a machine learning process model with quality assurance methodology. *Machine Learning and Knowledge Extraction*, 3(2), 392–413.
- Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., & Bhosale, S. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- van den Broek, E., Sergeeva, A., & Huysman, M. (2021). When The Machine Meets The Expert: An Ethnography of Developing AI for Hiring. *MIS Quarterly*, 45(3), 1557–1580.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
- Waardenburg, L., & Huysman, M. (2022). From coexistence to co-creation: Blurring boundaries in the age of AI. *Information and Organization*, 32(4), 1–11.
- Yang, J., Jin, H., Tang, R., Han, X., Feng, Q., Jiang, H., Zhong, S., Yin, B., & Hu, X. (2024). Harnessing the power of LLMs in practice: A survey on ChatGPT and beyond. *ACM Transactions on Knowledge Discovery from Data*, 18(6), 1–32.
- Yin, R. K. (2009). *Case study research: Design and methods* (Vol. 5). Sage Publications.
- Zhang, Z., Yoo, Y., Lyytinen, K., & Lindberg, A. (2021). The unknowability of autonomous tools and the liminal experience of their use. *Information Systems Research*, 32(4), 1192–1213.

Paper V

Unlocking the Potential of Generative Artificial Intelligence – A Case Study in Software Development

Mei Wu-Gehbauer
University of Cologne

Christoph Rosenkranz
University of Cologne

In: Proceedings of the International Conference on Information Systems (ICIS), 2024

Unlocking the Potential of Generative Artificial Intelligence – A Case Study in Software Development

Abstract

To unlock the potential of GenAI, software developers are facing a key tension. On the one hand, GenAI, as a general-purpose technology, offers vast possibilities for software development. On the other hand, developing software in enterprises requires a deep and nuanced understanding of the existing application landscape. This results in the need to align between GenAI's algorithmic representation and the developer context. Through an exploratory case study and drawing on the lens of functional affordances, we identify an interplay of two core mechanisms that software developers employ to realize action possibilities of GenAI-assisted software development in the organizational environment: (1) enacting immediate action possibilities, and (2) expanding potential action possibilities. With our study, we contribute to the understanding of GenAI systems as a space of action possibilities, the role of humans to shape these possibilities in recurrent use and design, and how humans collaborate with GenAI.

Keywords: Generative artificial intelligence, human-AI collaboration, software development, case study, functional affordances, action possibilities

1 Introduction

Generative artificial intelligence (GenAI) has considerably impacted business and industry, excelling in its ability to understand natural language and to generate text, images, or computer code (Denning, 2023). With its economic potential to increase productivity and its disruptive impact on how people work, it is at the center of strategic management attention for many organizations (Brynjolfsson et al., 2023; Chui et al., 2023b). In contrast to the application of discriminative artificial intelligence (AI), which infers its outputs from pre-defined data sources to predict and classify labels best suited for solving specific, close-ended tasks (Foster, 2022), the current generation of general-purpose GenAI models is trained on vast amounts of data representing a variety of real-world problem domains in order to generate new data samples, thereby broadening the scope of uses to more open-ended application scenarios (Eloundou et al., 2023). Given the virtually limitless and open-ended application scenarios, the key to unlock the potential from GenAI is the ability of humans to identify how to work well together with GenAI for the intended user context (Berg et al., 2023).

However, it is challenging for humans to identify how to work together with GenAI to create solutions that align to their needs and context. GenAI operates on an algorithmic representation of reality learned in training, which is not necessarily a faithful representation of the real-world context. This may lead to GenAI generating random or hallucinated solutions that can result

in misalignments with the context to which it is applied (Benbya et al., 2024). In order to complete specific tasks in an organization, context plays a critical role. For GenAI to be able to support tasks in an organization, it needs to have an understanding of the unique situational factors and context required for task completion (Rahwan et al., 2019). This implies that users need to actively assess and shape when and how GenAI can be applied and how context can be considered. It remains unclear how users handle this situation with GenAI's vast possibilities on the one hand and potential misalignments between the algorithmic representation and the specific organizational context on the other hand.

So far, existing literature only provides partial answers. The current body of research on human-AI interaction predominantly focuses on the use phase rather than the development and design of AI systems (e.g., Baird & Maruping, 2021; Fügener et al., 2022; Lebovitz et al., 2021). Moreover, the design of GenAI systems is largely considered from a technical perspective focusing on developing domain-specific GenAI models (e.g., Roziere et al., 2023) or enhancing the GenAI models with other techniques such as retrieval-augmented generation (RAG) or knowledge graph (e.g., Lewis et al., 2020; Pan et al., 2024; Peng et al., 2023; Ram et al., 2023). We find this problematic, since use and design of GenAI systems are interdependent and humans play a vital role to continually using as well as shaping the GenAI systems and enabling fit with their context for the systems to effectively support organizational tasks (Waardenburg & Huysman, 2022). In an organizational context, this is especially important given the unique situational factors and context GenAI needs to be aware of, such as established routines and processes, technical constraints, or operational context (Benbya et al., 2024). We aim to contribute to the body of research on human-AI interaction for GenAI systems by unpacking the sociotechnical mechanisms across the design and use of GenAI systems to unlock their potential in the organizational context.

We explore this phenomenon in the domain of *software development*, one of the supposedly most promising application scenarios of GenAI with a potential, estimated productivity improvement by 20 to 45 percent (Chui et al., 2023a). Software developers apply GenAI across a broad scope of different software development tasks along the stages of software development (Nguyen-Duc et al., 2023). Thus, GenAI has the potential to fundamentally change and transform the way how software is developed (Burtch et al., 2024; Scott, 2023). We deem software development to be a suitable domain because developing software can be a cognitively expensive task and software development in organizations requires a deep and nuanced understanding of the context comprising the existing application landscape and challenges posed by legacy code or technical debt (Fürstenau et al., 2019; Holland et al., 1999; Rinta-Kahila et al., 2023). We aim to develop a theoretical understanding of how software developers realize the potential offered by GenAI. We employ the concept of action possibilities (Markus & Silver, 2008) as a theoretical lens in this study and argue it is crucial to explore how software developers realize the action possibilities of GenAI-assisted software development in order to understand how the value of GenAI can be

unlocked in an organizational context. Therefore, the research question of our study is: “*How do software developers realize action possibilities of generative AI-assisted software development in organizations?*”

To address the research question, we employ a grounded, inductive, single case study approach. We investigate in-depth a large multi-national telecommunications enterprise that broadly rolled out GenAI-based solutions to its software developers starting in 2023. Based on our case analysis, we explore and theorize how software developers realize the action possibilities of GenAI for software development.

The remainder of the paper is structured as follows. First, in related work and theoretical background, we conceptualize GenAI-assisted software development from a representational perspective, highlight the importance of context, and introduce functional affordances as our theoretical lens. Second, we elaborate on our research design and method. Third, we present the empirical findings of our study. Fourth, we discuss the findings, our contributions, and limitations of our study. Finally, we close with a conclusion.

2 Related Work and Theoretical Background

2.1 Representational Perspective on GenAI-assisted Software Development

Designing and writing software code to solve a problem can be expensive (Allamanis et al., 2018; Thomas & Hunt, 2019; Zmud, 1980). Thus, software developers and managers traditionally have looked for ways to make software development easier, for example, by actively trying to reuse code (Banker & Kauffman, 1991), move common functionality into libraries to facilitate reuse (Von Krogh et al., 2005), or turn to online knowledge communities to get answers for a problem (Faraj et al., 2016). Since OpenAI’s ChatGPT has been made easily accessible for software developers via a simple user interface, via APIs as well as via developer tools such as GitHub Copilot (Ernst & Bavota, 2022), GenAI has become increasingly prevalent in software development (Burtch et al., 2024).

GenAI can be of enormous benefit for software developers because coding shares many features with human language and text (Allamanis et al., 2018). Code is a highly structured form of text and GenAI excels in learning syntactic and semantic patterns from the vast amount of text-based data they are trained on. At the same time, GenAI is not limited to specific tasks or predefined goals, as its general-purpose models are applicable to a broad range of developer scenarios (Nguyen-Duc et al., 2023). For example, GenAI systems can elicit requirements (Zhang et al., 2023), translate code from one language to another (Weisz et al., 2022), suggest and generate code (Barke et al., 2023; Bird et al., 2023), generate unit tests (Yuan et al., 2023), or generate documentation for code (Ahmed & Devanbu, 2022). In this study, our focus of GenAI is on large language models (LLMs) due to its relevance to software development tasks.

LLMs are a subset of GenAI models dealing with text understanding and generation (Dwivedi et al., 2023). As general-purpose models with broad applicability, LLMs are trained on large amounts of available web-scale language data (e.g., public Common Crawl, GitHub, Wikipedia) by applying a combination of ML algorithms (Touvron et al., 2023; Vaswani et al., 2017).

We know from previous studies that the resulting model is not a faithful representation of the real environment or context to which the model is then applied. It may only capture explicit knowledge in the form of language and text and not the rich know-how practices of experts in the field (Lebovitz et al., 2021), it may not be trained on incumbent work practices or legacy technology (Asatiani et al., 2021), or it may be trained on outdated data (Poldrack et al., 2023). We term the resulting GenAI model together with the data it is trained on, as an *algorithmic representation* of reality that may or may not be a faithful representation of the reality (Alaimo & Kallinikos, 2022; Østerlie & Monteiro, 2020).

Representations are a central concept in IS theory (Burton-Jones et al., 2017). The concept refers to the idea that the purpose of information systems (IS) is to provide representations of real-world phenomena and IS are more useful if they provide faithful (i.e., accurate and complete) representations. Representations have been applied to effective use of IS (Burton-Jones & Grange, 2013), IT alignment (Sia & Soh, 2007), and enterprise system fit (Strong & Volkoff, 2010). Building on this stream of research, our study aims to use representations as a theoretical lens to explain how GenAI represents reality in the context of software development. In representation theory, an IS comprises three structures: deep structure, surface structure, and physical structure (Wand & Weber, 1995). Deep structure specifies the domain which the IS represents. Surface structure refers to the way users access and interact with the IS. Physical structure refers to the infrastructure that supports the other two. Since deep structure is the key to representation (Burton-Jones & Grange, 2013), we focus on this aspect. Deep structure in our context refers to the GenAI model together with the data it is trained on that constitute the algorithmic representation of reality.

The challenge for this algorithmic representation is to generate results that developers can use and that align to their work context. For this, GenAI systems take inputs from developers to infer what needs to be done. During inference, the models rely on their learned algorithmic representation to create probable solutions and provide these results back to the software developers (Russell & Norvig, 2016). The algorithmic representation with the inference of the task solution is opaque and highly complex (Berente et al., 2021). GenAI's algorithmic representation is a *probabilistic model*, that is, the same inputs will produce different outputs on different inference executions, and outputs can be very sensitive to minor changes in the input parameters (Miceli-Barone et al., 2023). In addition, so-called 'hallucinations' are a common concern with GenAI-generated code as it can be imaginary (Agrawal et al., 2023). There is a big difference between inferred code from the algorithmic representation that seems reasonable and compiles without errors, versus correct code that addresses the developer task considering

its context (Burtch et al., 2024). Consequently, GenAI models applied for coding are rather fragile and may only possess a shallow understanding of code (Wu et al., 2023). The challenge of GenAI remains to create task solutions that align with the developer context and address the developer task at hand.

This challenge has been highlighted in GenAI research as the ‘alignment problem’, which broadly refers to the challenge of ensuring that AI systems are useful to their human users and pursue goals that match human values and interests (Ji et al., 2023; Millière, 2023; Ngo et al., 2022; Wolf et al., 2023). The alignment problem so far has been addressed from a technical perspective by introducing reward policies and human reinforcement feedback (Arumugam et al., 2019), prompting strategies (e.g., Liu et al., 2023; White et al., 2023), and retrieval augmented generation architectures (e.g., Lewis et al., 2020; Peng et al., 2023; Ram et al., 2023). In this study, we draw on the sociotechnical definition of alignment from IS research referring to an emergent process of dynamic interactions and continual adjustments between humans and IS that collectively may result in greater performance (Benbya & McKelvey, 2006; Benbya et al., 2019). Our focus in this study is on alignment between the GenAI system and the developer context in order for the GenAI system to create task solutions that address the developer task at hand considering the developer context. Figure 1 illustrates our conceptualization of GenAI-assisted software development from a representational perspective.

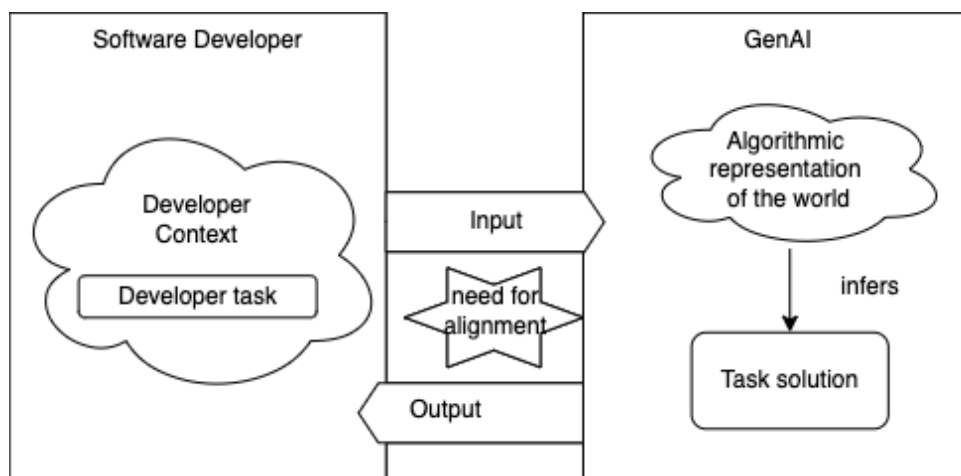


Figure 1: Representational perspective on GenAI-assisted software development

2.2 Context in GenAI-assisted Software Development

Context in software development refers to all relevant information that developers need to complete a task (Dybå et al., 2012). Context can be highly varied ranging from human-related (e.g., expert developer knowledge, individual development preferences), organizational-related (e.g., rules, procedures and standards, established development practices), to domain-related (e.g., technical development conventions, architecture requirements) (Johns, 2006).

The usefulness of GenAI-created outputs for developers is largely dependent on how well the GenAI system infers a task solution that is aligned with the developer's context. For example, GitHub Copilot works well when the information about the currently active file and code region provides sufficient context to infer a task solution (Bird et al., 2023). However, when the information provided about the context is not sufficient (e.g., because functionalities or APIs are defined elsewhere or an external library is required not seen during training), GenAI might start hallucinating, for example, by making up libraries that do not exist or using APIs incorrectly (Agrawal et al., 2023). In this regard, context acts as a sensitizing device that makes us more aware of the potential situational and temporal boundary conditions of the observed phenomena (Bamberger, 2008). It allows us to better assess the applicability of GenAI for software development. Building on this, we differentiate between *context-light* tasks which refer to tasks where the developer can be expected to complete the task with limited access to the specific context, and *context-heavy* tasks that cannot reasonably be expected to be completed without access to the specific developer context. For example, context-light tasks can be code suggestions that can be inferred from existing code fragments that the GenAI model was trained on or generating standard methods and classes where the relevant context is accessible in the GenAI system. Context-heavy tasks involve, for example, translating legacy code that the GenAI was not trained on (Barke et al., 2023), refactoring and reviewing code considering specific coding standards and practices (Shypula et al., 2023), or adding code to large repositories of legacy code with embedded business logic that emerged over time. This differentiation is important because it determines how developers align with GenAI systems and provides the basis for data sampling.

Often, a substantial share of context-heavy tasks is prevalent in an organizational setting, as software developers are faced with legacy application landscapes (Fürstenau et al., 2019; Holland et al., 1999), technical debt resulting from monolithic architectures, poor documentation, “spaghetti code” (Rinta-Kahila et al., 2023), established routines and development practices (Berente et al., 2016), or formal processes and procedures to ensure quality and security of the code base. The challenge of identifying and including the relevant context as input for GenAI systems intensifies in these situations, as the GenAI system must understand various and diverse factors such as the developers' needs, future path dependencies regarding target architecture and engineering practices, or the impact of decisions made in the past embedded in the business logic of the legacy application landscape. Consequently, there is a high risk that GenAI may not adequately incorporate these contextual factors in a real-world scenario. This leads to misalignment between the outputs produced by the GenAI system and the developers' context (Benbya et al., 2024). Recent literature highlights the need for empirical studies to examine how GenAI-assisted software development works in and creates benefits for real-world scenarios, with software developers and actual programming tasks in an organizational context (Xu et al., 2022).

2.3 Action Possibilities for GenAI-assisted Software Development

Considering the frequent occurrence of context-heavy tasks for software developers in organizations and the difficulties in specifying this context, it is of utmost importance to understand and theorize how software developers can effectively leverage the capabilities offered by GenAI system while simultaneously managing the challenges posed by their specific contexts and alignment. To understand how GenAI is used in organizations and what the potential consequences for software development are, we draw on the concept of *functional affordances*, which provides a theoretical lens to examine how GenAI relates to developers' work and tasks (Leonardi, 2011). In our study, functional affordances are "action possibilities for goal-oriented actions afforded to specified user groups by technical objects" (Markus & Silver, 2008, p. 622).

First, for the phenomenon of GenAI systems, functional affordances provide action possibilities allowed by the *material properties* of the technical object itself. In the case of GenAI, we conceptualize three material properties: (1) GenAI systems have *self-learning capabilities*; they are built on ML algorithms that learn from large and varied amounts of data and adapt to new data over time without explicit programming (Berente et al., 2021); (2) GenAI systems have *generative capabilities*; they are able to generate new data based on inferences made from their GenAI model (i.e., algorithmic representation) (OpenAI, 2023); (3) GenAI systems have *interaction capabilities*; they can interact with developers, take inputs, and provide outputs based on the users' inputs (Brown et al., 2020). Given these dynamic and adaptable material properties, functional affordances provided by GenAI systems are not deterministic and will change over time with changed models, changed inputs, and changed outputs. Second, affordances emerge when the material properties of information systems are interpreted as affording *action possibilities within the context of their use*. For our phenomenon of GenAI, the context of use can be varied as described in the previous section. GenAI affords different action possibilities for context-light tasks than for context-heavy tasks. Consequently, the actual affordances of GenAI systems for software development are highly linked to the required level of context for the task to be performed. Third, *the affordances need to be perceived as such* before they can be actualized. The need for perceiving affordances implies a distinction between the designer, who creates affordances, and the users, who perceive to actualize affordances (Leonardi, 2011). These lines, however, are blurred for GenAI systems. Instead, software developers are co-creators in the creation and modification of the GenAI outputs, thereby recurrently engaging in both design and use activities (Waardenburg & Huysman, 2022).

In sum, we argue that, depending on the context level of the task and the need for alignment, functional affordances of GenAI systems provide two dimensions of action possibilities: *immediate action possibilities* and *potential action possibilities* (Pentland et al., 2022), which need to be realized through interactions between the developer and the GenAI system. We consider the two dimensions as a continuum of action possibilities ranging from immediate to poten-

tial action possibilities depending on the level of context need. This provides the basis for our conceptualization of functional affordances for this study. We refer to immediate action possibilities when the developer applies GenAI to perform context-light tasks. For example, in case developers need to write standard code functions or reuse libraries they used before, they enact immediate actions because it can be assumed that the algorithmic representation closely fits with the developer context. In this case, GenAI can help to accelerate code completion and directly increase productivity (Barke et al., 2023). In case of potential action possibilities, developers aim to apply GenAI for context-heavy tasks. In this scenario, the algorithmic representation can be assumed to be divergent from the developer context and higher level of alignment is required. For example, if developers want to translate legacy code to modern programming languages and documentation on the legacy code is not or scarcely publicly available, GenAI needs to be provided with additional context information on the legacy code to be able to support the translation task. Figure 2 provides an illustration of the action possibilities for GenAI-assisted software development.

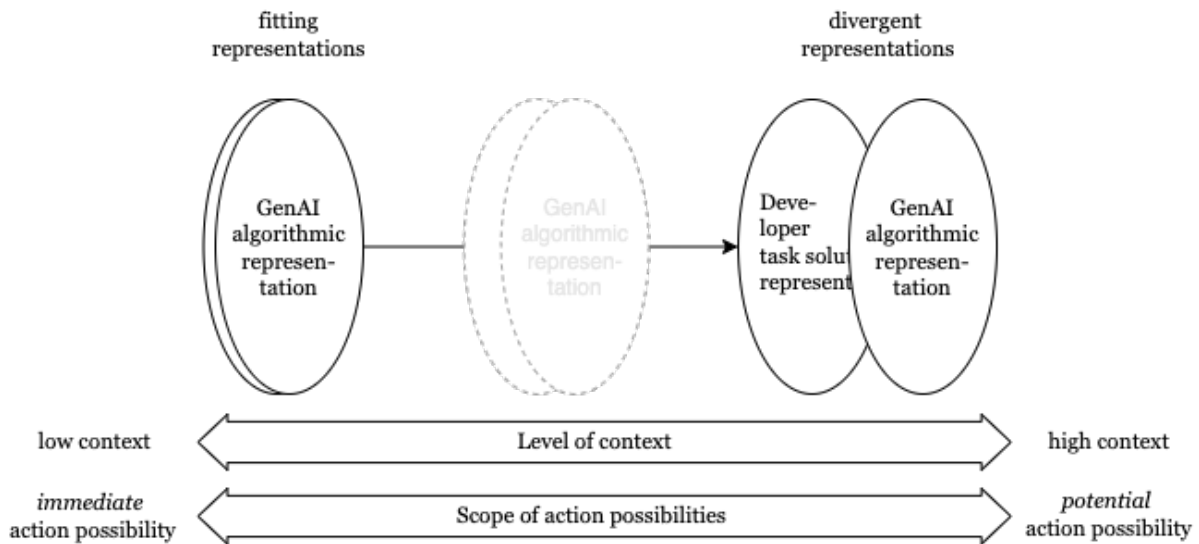


Figure 2: Action possibilities for GenAI-assisted software development

3 Research Design and Method

Since the broad application of GenAI systems supporting software development in organizations is a new phenomenon, we conducted an exploratory case study (Eisenhardt, 1989b; Yin, 2009) and studied one organization in depth to gain a detailed and rich understanding. Single case studies are well established in IS research to explore the interaction between organizational actors and technology in their contexts. We draw on the lens of functional affordances to examine how software developers engage with GenAI and realize its possibilities for software development.

As case organization, we selected the internal IT service provider of a multi-national telecommunications enterprise. We consider this case organization an ideal case site to investigate our

research question due to several reasons. First, the telecommunications provider is among the leading telecommunications companies worldwide. The board of management has set the application and adoption of AI as one of its top drivers for innovation and growth. All business units including the internal IT organization are contributing with dedicated AI-based solutions to improve customer experience, productivity and quality. Second, the internal IT service provider offers the complete IT service portfolio and thus makes an important contribution to the competitiveness and market success of the whole enterprise. It is responsible for the design, implementation, provision and maintenance of all IT systems and IT infrastructure services required to support the business of the enterprise across all its business segments. Third, the internal IT service provider has launched several initiatives since beginning of 2023 to actively leverage GenAI capabilities across the entire software development lifecycle.

The case organization introduced GenAI-based systems assisting software developers since the beginning of 2023, following the launch of ChatGPT end of 2022. Three types of GenAI systems are available to the developers: (1) GenAI-based code generation system offered as a Software as a Service (SaaS), (2) access to pre-trained, closed-source LLMs offered as a cloud-based service, and (3) access to pre-trained, open-source LLMs.

Our main method of data collection is interviews, supplemented with documentations and presentations of the GenAI systems being applied at the case organization. Informants were sampled along the criteria of lower versus higher need for providing additional context to the GenAI systems when applying GenAI to support their software development tasks. *Lower context need* is realized when prompt-based interactions are sufficient, and developers rely on the inherent knowledge and capabilities of the GenAI system without additional information. *Higher context need*, on the other hand, is realized when the GenAI system needs to be enhanced with additional, contextual information to be able to generate relevant outputs.

We focus on software developers in the case organization as key informants, as our study focuses on how the software developers actualize the action possibilities. As software developers, we refer to all professionals who use code to solve problems and apply or develop GenAI-assisted software development. This includes developers with different expertise areas, such as full stack development, ML engineering, or software architecture. Table 1 provides an overview of the interview data.

For data analysis, we followed the iterative process of open, axial, and selective coding, using a form of grounded theorizing (Strauss & Corbin, 1998). To address our research question, we focus on the software developer's actions regarding GenAI-assisted software development as our core unit of analysis, as these constitute the mechanisms that enable the developers to realize the action possibilities with GenAI.

Given the characteristic of coding as a “cyclical act” (Saldaña, 2009), our coding process involved a highly iterative approach. Initial concepts and relationships informed subsequent

Table 1: Interview data

Context	# of Interviews Software Developers	Expertise Area	Interview Time in Minutes
Lower context need	15	Full stack	842min
Higher context need	4	Full stack	228min
	2	Software Architecture	122min
	3	ML Engineering	170min
Total	24		1.362min

rounds of theoretical sampling, data analysis, and continuous conceptual development. The initial stage of analysis (open coding) focused on identifying preliminary concepts, while closely examining the actions of software developers in relation to GenAI in software development. We conducted a line-by-line coding of all interviews, generating open codes. These were subsequently grouped and integrated, leading to the first-order concepts. For example, *validating understanding of outputs* is a concept that originated from the concepts of asking GenAI for explanation and verifying own understanding of outputs.

During the axial coding phase, we further developed categories and systematically related these to the first-order concepts identified earlier in the open coding stage. This step of axial coding was informed by the theoretical lens of functional affordances to explore how software developers realize the action possibilities of GenAI in software development by aligning the objective to apply GenAI and by aligning between the developer's task context and GenAI's algorithmic representation. For instance, when developers realize potential action possibilities, we found them *aligning to company-level objectives* to more effectively tackle complex or resource-heavy tasks. When aligning the representation, the identified category *validate* was associated with validating understanding of outputs, validating using known coding standards, and validating using software development practices, revealing that developers build on available knowledge and experience as well as on established software development routines for validation of GenAI outputs.

In the selective coding phase, we integrated the identified categories into the core categories of *enacting* and *expanding* the action possibilities of GenAI in software development. These concepts unified the other categories into a coherent theoretical framework, relating them to the concepts of context and the respective dimensions of action possibilities. The core category of *enacting* synthesizes the mechanisms interact and validate, which taken together are required to enact the immediate action possibilities for context-light tasks. The second core category of *expanding* comprises the mechanisms design and control, which are instrumental to expand the space of action possibilities for GenAI-assisted software development by creating new immedi-

ate action possibilities to tackle context-heavy tasks. The identified mechanisms are described in detail in the next section.

We triangulated the coding with additional materials such as documentation and demo presentations from the GenAI systems to increase validity. Following these steps, and through iterative coding and revision, we ultimately converged to a thorough understanding and theoretical explanation of the observed phenomena.

4 Findings

We find that for GenAI-assisted software development in organizations, software developers employ alignment mechanisms to realize action possibilities for GenAI-assisted software development. Alignment mechanisms are both required for (1) *immediate action possibilities* for development tasks with lower context need, and for (2) *potential action possibilities* for development tasks with higher context need. While both areas require alignment along the same dimensions of aligning the objective and aligning the representation, different mechanisms apply for immediate action possibilities versus for potential action possibilities. Next, we first outline the identified mechanisms for immediate action possibilities, followed by those identified for potential action possibilities. Example quotes are provided to illustrate the mechanisms. All names of interviewees used are pseudonyms to ensure privacy of the participants.

4.1 Immediate Action Possibilities

Immediate action possibilities of GenAI-assisted software development unfold for context-light development tasks with lower context need. For these tasks, standard, prompt-based interactions that the GenAI systems provide as well as the inherent capabilities of the GenAI systems are sufficient to support developers with accomplishing the tasks. For example, we found the following tasks in this category: initial information gathering, generation of boilerplate code, code completion, or unit testing. The mechanisms that the developers employ to realize the immediate action possibilities for GenAI-assisted software development are described subsequently.

Aligning to individual-level objective has been identified as the mechanism for *aligning the objective* to apply GenAI for immediate action possibilities. We found that software developers engage with GenAI-assisted software development for the immediate action possibilities to enhance their individual work efficiency, quality, and overall satisfaction:

“The greatest added value of GenAI is definitely efficiency. By being more efficient, I have more time to review and improve the code quality. I conducted a small experiment. I took a standard task that usually takes me about three to four hours based on experience. I compared how quickly I could complete it with GenAI. I finished

in one to two hours and achieved a test coverage of 80 to 90 percent. Had I done it without GenAI, I certainly wouldn't have achieved such high test coverage. I had time to invest in quality. As a result, my overall quality during the development process was higher because I wrote the unit tests during the development itself.” (Jacob)

Interact and **validate** have been identified as mechanisms for aligning the *representation* of GenAI with developer context in case of immediate action possibilities.

Interact manifests itself in iterative interactions with GenAI by providing input to GenAI and iteratively refining the input. It includes *preparing the input* based on problem understanding and break-down into subtasks, as well as *iteratively refining the input* by providing examples and experimenting with different prompts:

“Once I understand what I need to build and I have a clear goal in mind, I break it down into smaller subtasks. I then use GenAI to implement these subtasks by asking GenAI to generate the boilerplate code for each.” (Gabor)

“If GenAI's response isn't quite right, I can offer guidance by saying, “You have made an error here, please correct it.” This helps steer the direction of the code suggestions. Such interaction is particularly crucial when you're just beginning to use GenAI. Once I have developed a logical solution plan for myself, I can follow it step by step. I can instruct GenAI: “Here is an example. I would like a regex that matches this string and creates capture groups from it”.” (Anand)

Validate refers to validating the outputs of GenAI and involves *validating understanding of the outputs* based on own experience, *validating using known coding standards*, and *validating using software development practices* such as compilation and execution of code, performing code reviews, and testing practices:

“There's this principle in clean code or refactoring that a good method should not exceed one page in length. The code that the AI generates is usually within this limit if it's well-crafted. If not, it suggests that I haven't adequately broken down my problem into subtasks. I make sure I thoroughly understand the code suggestions, I can then accept it and feel fully responsible for it. I have never used code that I didn't understand.” (Alexander)

“You need a solid grasp of the programming language you're using and keep yourself updated with the latest developments. When GenAI gives you code suggestions,

treat them as if you're reviewing another developer's work. It's important to critically assess whether the AI-generated code is not only correct but also optimized, or if it has the potential to be improved. Check for any issues just as you would with human-written code in the code review, because overlooking problems can lead to serious complications later.” (Matteo)

We summarize all mechanisms that realize the immediate action possibilities of GenAI-assisted software development as enacting mechanisms. **Enacting mechanisms** are employed by software developers to engage in GenAI-assisted software development for tasks with lower context need.

4.2 Potential Action Possibilities

Potential action possibilities of GenAI-assisted software development unfold for context-heavy development tasks with higher context need. We consider them as *potential* action possibilities, since the inherent knowledge and capabilities of the GenAI systems are not sufficient but need to be enhanced with additional context to be able to generate relevant outputs assisting software developers. For example, we found these tasks with higher need for context: explanation of custom code, translation of legacy code, or acceptance testing. As the potential action possibilities are not readily available for software developers through *enacting mechanisms*, we identified mechanisms employed by software developers, which turn the potential action possibilities into immediate action possibilities, making the potential action possibilities accessible to the software developers and making GenAI feasible to support tasks with higher need for context. The mechanisms are described in detail subsequently.

Aligning to company-level objective has been identified as the mechanism for aligning the objective to apply GenAI for *potential* action possibilities. We found that some software developers engage in expanding the action possibilities of GenAI-assisted software development when they identify time-consuming technical transformation activities or highly recurring tasks that require additional contexts to create company-level value in terms of faster delivery, cost reduction or higher quality and security:

“We have several thousand interfaces on this old platform that need to be updated to a more modern technology because we have to decommission the old system. This led us to the idea of accelerating the conversions with the help of GenAI.”
(Giorgi)

“There are already translation systems available for the common programming languages. However, within our company, we still use legacy programming languages. There is a particular need to support the translation of languages that are

already outdated, not well documented, and for which there are no active developer communities. The goal is not full automation, but to accelerate this transformation process using GenAI.” (Aya)

Design and control have been identified as mechanisms for aligning the representation of GenAI with developer context in case of potential action possibilities.

Design refers to designing additional contextualization enhancing the GenAI systems and resulting in new, reusable GenAI-based artifacts. It encompasses *selecting an LLM* based on understanding of GenAI functions and performance comparison, *providing task-specific contexts* using contextualization techniques, such as few- and multi-shot learning, retrieval augmented generation, or finetuning, and *developing new GenAI-based artifacts for reuse* by developers. The emerging GenAI-based artifacts offer new immediate action possibilities:

“We go for the GPT model as the default approach, because we have seen that it achieves superior performance across a range of tasks. But for very specific tasks, we opt for open source models that can be fine-tuned on our datasets and can in this case offer a better performance while also being more cost-efficient.” (Benjamin)

“We are migrating several applications from on premise to the cloud environment and have to migrate to new technologies that integrate more seamlessly into the cloud services. We have developed a code translation solution using GenAI. We published the solution on the DevOps platform to make it available for other developers. They can easily clone and reuse it.” (Chen)

Control relates to the new GenAI-based artifact emerging from the design mechanism and include the following aspects: *scoping* of the application area of the artifact, *initial validation* of the artifact and its outputs involving technical and user testing, *ongoing steering* of artifact behavior by specifying instructions or implementing pre-defined workflows as well as ongoing monitoring of the feasibility and quality of artifact behavior by establishing quality measures, ongoing validation with golden standards, or gathering and analyzing user feedback:

“Our GenAI tool for addressing code vulnerabilities was tested in different projects. In one case, it was not successful due to significant changes in the frontend resulting from integrating the tool and its code suggestions. We cannot guarantee the appearances to be unchanged post-fix. On the other hand, the GenAI tool performed well in backend services, leading us to decide to rollout the solution for backend services only.” (Giorgi)

“We work with workflows in our GenAI solution, which help us decompose the complex problem of translating code from a legacy programming language into a new

language into smaller problems and ensure that all problems are being addressed. We upload the documentation we have into a context database. Instead of sending a large prompt, for instance, we send three smaller prompts to the GenAI model. We noticed that when we give GenAI many instructions at once, 10 out of 20 are being ignored. This is not the case with workflows, as they cannot be skipped.” (Aya)

We summarize all mechanisms to realize the potential action possibilities of GenAI-assisted software development as expanding mechanisms. **Expanding mechanisms** are employed by software developers to expand the space of action possibilities for GenAI-assisted software development by turning the potential action possibilities into new immediate action possibilities via the means of new GenAI-based artifacts.

Table 2 provides an overview of the identified core mechanisms that software developers employ to realize immediate as well as potential action possibilities of GenAI-assisted software development.

Table 2: Overview of alignment mechanisms

	Enacting Mechanisms for Immediate Action Possibilities	Expanding Mechanisms for Potential Action Possibilities
Aligning objective	<i>Aligning to individual-level objective, e.g., individual efficiency of coding</i>	<i>Aligning to company-level objective, e.g., acceleration of complex, time-consuming, or recurring software development tasks</i>
Aligning representation	<i>Interact and Validate</i> - Interact: Iterative interactions with GenAI by providing input and iteratively refining input - Validate: Validation of GenAI outputs based on own experiences, known coding standards, and using software development practices	<i>Design and Control</i> - Design: Design of enhanced contextualization resulting in new, reusable GenAI-based artifacts - Control: Scoping and initial validation of new GenAI based artifacts, ongoing steering and monitoring of GenAI-based artifacts

4.3 Interaction between Enacting and Expanding Mechanisms

The two core mechanisms of *enacting* and *expanding* are key to realizing the action possibilities of GenAI-assisted software development. We find them to be intertwined: while enacting mechanisms serve as key mechanisms to realize existing immediate action possibilities, new action possibilities are created through expanding mechanisms. Accordingly, the space for immediate action possibilities is being expanded. Thereby, new GenAI-based artifacts emerge that provide the new immediate action possibilities, which in turn can be realized through enacting mechanisms.

We offer the following vignette from the case organization to illustrate the intricate interplay between *enacting* and *expanding*.

During service scans, code vulnerabilities are regularly identified, which can lead to a range of security issues and are producing a recurring need for software developers to address these vulnerabilities. A group of software developers decided to use GenAI to resolve the vulnerabilities to increase software quality and security and reduce risks for the company: *“We’ve been thinking about how we can improve the overall code quality and security level of the company. Patching security vulnerabilities is a task that requires recurring effort.”* After evaluating both closed-source and open-source GenAI systems, the closed-source GenAI system was selected due to superior performance. For the GenAI system to generate code suggestions for addressing the identified vulnerabilities, the source code security scan reports and the source code files are provided as contexts to the GenAI. Due to the recurring need for developing these kinds of code security corrections, a GenAI-based code security patch solution is developed and provided as a GenAI-based software solution that can be integrated as an additional step into the pipeline of every software project: *“We have added an additional pipeline step between the analysis through static application security testing and deployment that everyone can include. This step is using GenAI to generate code suggestions addressing the identified vulnerabilities from the scans.”* Before providing the solution to other software developers, the solution is being piloted and validated in pilot software projects. Due to undesirable changes in the frontend and the higher need to rework the code suggestions, the decision has been made to rollout the solution for backend applications only: *“We tested it with sample projects from the company. In many cases, it worked very well. However, in one project, it wasn’t as successful because the automated fixing of security vulnerabilities changed the appearance of the website. It altered input fields and similar elements. This means frontends are problematic. With backend services, it works exceptionally well.”* Users of the solution are provided with the possibility to rate each code suggestion. The ratio of accepted suggestions is monitored on an ongoing basis. The GenAI-based code security patch solution is supporting software developers to address the vulnerabilities more efficiently by providing suggestions for code modifications. The generated code suggestions are not automatically deployed into the codebase but need to be reviewed by the software developers using this solution in their software projects beforehand. Consequently, the code security patch solution is provided as merge requests to the software developers who are validating the code suggestions and can accept, change, or decline the code suggestions: *“We have handed over the GenAI-based solution to a team that will operate and continue to develop it including adapting it to changing models. We’ve specifically built in functionality that allows developers to comment on and edit the merge requests. For this, we’ve created a tool that analyzes in how far the suggestions are accepted or further processed. For the users, I see the review of the suggestions as their responsibility - checking what has been generated, determining if it matches what is needed, assessing if the quality is sufficient or identifying what’s missing.”*

This example shows how the *expanding mechanisms* involve reducing security risks stemming from code vulnerabilities for *aligning to company-level objectives*. The code security patch

solution results as new, reusable GenAI-based artifact from the design mechanism during expanding to create the new immediate action possibility of addressing vulnerabilities by applying the code suggestions from this GenAI-based artifact. Through the *control* mechanism as part of expanding, the GenAI-based artifact providing the emerging new immediate action possibility is scoped, initially validated as well as monitored concerning feasibility and quality. The new action possibilities of GenAI-assisted code vulnerability resolutions are realized through *enacting mechanisms*. They enable software developers to be more efficient, thereby *aligning to individual-level objective*. The developers interact with the GenAI-based code security patch solution and need to *validate* the outputs of the GenAI-based solution.

5 Discussion and Contribution

In our in-depth case study, we consider the sociotechnical aspects involving the GenAI system, humans using and designing GenAI, and the organizational and task context. We explored how software developers realize action possibilities of GenAI-assisted software development in the organizational environment of a large enterprise. Figure 3 proposes a framework that summarizes our findings and insights from our case study. With the framework we posit that realizing action possibilities of GenAI-assisted software development involves enacting and expanding mechanisms, which are intertwined and depend on the work and task context. Software developers draw on *enacting* mechanisms to realize the immediate action possibilities of GenAI-assisted software development. With increasing context need, expanding mechanisms as actions taken by developers are required to expand the space of action possibilities to include additional potential possibilities afforded by GenAI beyond the existing, immediate action possibilities. The resulting action possibilities emerging as new, immediate action possibilities are again realized by developers through enacting mechanisms.

When *enacting* immediate action possibilities, software developers apply GenAI to support their task completion. This is possible when the context provided to GenAI via the inherent capabilities of the GenAI system is sufficient to produce useful outputs enhancing the developer's task completion. This mechanism ensures that developers are supported to get their tasks done resulting in faster completion. While immediate action possibilities are readily available, they need to be enacted nevertheless by the software developers. Due to the probabilistic nature of the GenAI models, this enactment involves interaction with the model and validation of the model outputs to arrive at a useful output. The goal here is to maintain the developer flow and apply GenAI to get things done faster.

When developers engage in *expanding* the space of potential action possibilities, GenAI cannot be readily applied to the current work and task context of the software developers. With increasing level of specific context need, developers need to actively shape new action possibilities by engaging in design activities. This enables developers to apply GenAI for more task

scenarios, while enlarging the space of potential action possibilities for GenAI-assisted software development. At the same time, it might include changing old ways of how work is done. By designing enhanced contextualization to align with developer tasks with high context needs, enhanced GenAI-based system artifacts emerge that turn potential action possibilities into new immediate action possibilities. Additionally, the new GenAI-based artifacts are scoped, validated and monitored to provide the basis for enacting these new action possibilities. The focus here is to enlarge the space of action possibilities for GenAI and leverage additional potential for the organization.

Taken together, our framework shows how developers realize the action possibilities afforded by GenAI thereby navigating the potential misalignments of general-purpose GenAI models and the specific developer context. These mechanisms stepwise expand the action possibilities of GenAI for software development and provide the basis for transforming software development with GenAI.

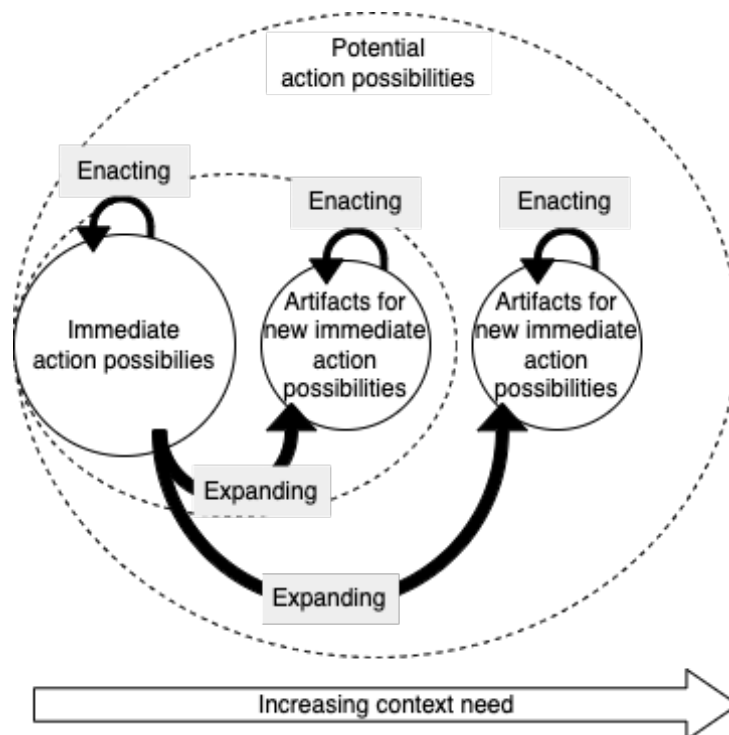


Figure 3: Realizing action possibilities of GenAI-assisted software development

Our study offers several contributions to and novel views on human-AI interaction for GenAI.

First, our study contributes to the literature on agentic IS artifacts by conceptualizing GenAI as a space of action possibilities. With that, we build on previous work that points out that knowledge workers need to navigate the rapidly advancing scope and sophistication (frontier) of GenAI for it to add value (e.g., Dell’Acqua et al., 2023; Susarla et al., 2023). We found that GenAI cannot be understood as a technology with features and capabilities that can be readily applied to a certain domain such as software development. Rather, GenAI should be regarded as affording a broad variety of both immediate and potential action possibilities for completing a task. Our

findings add to the existing literature on IS delegation and IS as agentic artifacts, to which tasks can be delegated to or vice versa (e.g., Baird & Maruping, 2021; Fügener et al., 2021). We detail the work on IS delegation when interacting with GenAI, exploring mechanisms when and how humans interact with GenAI artifacts to complete the work in organizations. While we recognize that GenAI systems as agentic artifacts are endowed with assets and capabilities, we found that depending on the desired objective and task context, GenAI systems offer action possibilities that need to be realized and can be expanded. This introduces a new perspective on IS delegation: the scope of tasks to be supported with GenAI is much more open-ended and differs in their context need; therefore, delegation is not a one-time decision but involves a constant re-assessment of delegation decisions and with the possibility to extend the delegation scope. Our findings demonstrate the importance of humans' ability to identify how to work well together with GenAI for the intended objective and task (Berg et al., 2023). To realize the potential of GenAI, we show the dynamics of how action possibilities can be realized and extended with the identified mechanisms of enacting and expanding and their interplay.

Second, we demonstrate the important role of humans in actively shaping action possibilities of generative technologies thereby contributing to the literature studying digital technologies through an affordance lens (e.g., Leonardi, 2011; Pentland et al., 2022; Seidel et al., 2013). Out-of-the box, GenAI can only create conditions for new ways of completing tasks, and it falls on the shoulder of human actors to actively shape these action possibilities to realize the potential of GenAI and create value through GenAI. Instead of perceiving affordances in their use following an intended design, for generative technologies use and design activities are mutually dependent on each other and unfold during the human-AI interaction. Action possibilities of GenAI are realized through actions of human actors combined with affordances made possible through GenAI's ability to consider user input and context information. Our findings show that, depending on context need, differing mechanisms are required to realize action possibilities with GenAI and how use and design activities are intertwined. This is broadening the affordance lens when investigating digital technologies so that social and material aspects of digital technologies are not only enacted in use for a specific user intent (Markus & Silver, 2008) but new affordances may be shaped in the interaction of humans with generative technologies that can lead to designing new action possibilities. This is also in line with previous literature on AI proposing that the lines between IS use and design are blurred in the context of AI (Waardenburg & Huysman, 2022). We identified two mechanisms, enacting and expanding mechanisms, and proposed a framework that reflects these recurrent use and design activities when interacting with generative technologies. Context plays a crucial role in this respect given the algorithmic representations of generative technologies. When the representation fits the user context, users can rely on enacting mechanisms to engage with GenAI to complete a task. When, on the other hand, the representation diverges, the expanding mechanisms come into play creating new GenAI-based artifacts, which integrate GenAI's algorithmic representation with the user context and make it possible for users to realize these new potential possibilities of GenAI. We provide

a concrete account in the domain of software development and of what software developers do to realize the action possibilities by engaging in recurrent use and design of GenAI systems. We thereby provide a deeper understanding of the underlying mechanisms of human-AI interaction for generative technologies.

In terms of practical recommendations, with the results of our study from a large enterprise, we provide valuable insights for software developers, technology leaders, and managers leveraging GenAI in an organizational context. We show that GenAI should not be merely considered as a tool to be used. It rather offers a large space of action possibilities for developers that need to be actively shaped. Contrasting recent practice reports proclaiming AI will marginalize the relevance of software developers (Bajarin, 2024), our findings underscore the pivotal role of developers in unlocking the potential of GenAI. In addition, organizations should be aware of both their context-light and context-heavy activities to adapt their approach to applying GenAI. Beyond context-light scenarios, where GenAI can be applied with low alignment efforts and generates immediate value, expanding the application of GenAI to costly, context-heavy scenarios can yield added value for organizations. The mechanisms identified for enacting and expanding GenAI-assisted software development provide orientation for practitioners to realize the possibilities of GenAI.

Our study has several limitations. First, due to our approach of a single case study, while it enables in-depth understanding, it typically provides less abstraction from the case organization. However, we chose to investigate our research question in the domain of software development, which is a well-established domain with similar needs and challenges across different organizations and industries. With this and by relating our findings to existing literature, we move towards abstraction and theoretical understanding. Second, although we draw on the domain of software development in this study, we propose that the findings are not exclusive to this domain, but rather add to the understanding of human-AI interaction with generative AI. In this regard, further studying other domains would be a fruitful way forward to make the suggested framework more robust and general in nature.

6 Conclusion

In organizations, completing a task rarely starts with a completely clean slate, as it usually builds on existing and often unique processes and systems. We illustrate this reality using the example of the software development domain. Software developers usually apply GenAI systems when working on large repositories of existing code, adding, or adapting functionalities, refactoring code, or debugging and reviewing the existing code. Each software developer operates within a unique context. To realize the potential of GenAI in such predominantly context-heavy environments, we need to understand how humans actively shape GenAI systems to continually enact and expand the possibilities of GenAI. The importance of humans playing an active role

in shaping GenAI will continue to be an important research stream when assessing the potential and impact of GenAI for individuals, organizations, and society.

References

- Agrawal, L. A., Kanade, A., Goyal, N., Lahiri, S. K., & Rajamani, S. K. (2023). Guiding language models of code with global context using monitors. *arXiv preprint arXiv:2306.10763*.
- Ahmed, T., & Devanbu, P. (2022). Few-shot training llms for project-specific code summarization. *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, 1–5.
- Alaimo, C., & Kallinikos, J. (2022). Organizations decentered: Data objects, technology and knowledge. *Organization Science*, 33(1), 19–37.
- Allamanis, M., Barr, E. T., Devanbu, P., & Sutton, C. (2018). A survey of machine learning for big code and naturalness. *ACM Computing Surveys (CSUR)*, 51(4), 1–37.
- Arumugam, D., Lee, J. K., Saskin, S., & Littman, M. L. (2019). Deep reinforcement learning from policy-dependent human feedback. *arXiv preprint arXiv:1902.04257*.
- Asatiani, A., Malo, P., Nagbol, P. R., Penttinen, E., Rinta-Kahila, T., & Salovaara, A. (2021). Sociotechnical Envelopment of Artificial Intelligence: An Approach to Organizational Deployment of Inscrutable Artificial Intelligence Systems. *Journal of the Association for Information Systems*, 22(2), 325–352.
- Baird, A., & Maruping, L. M. (2021). The Next Generation of Research on IS Use: A Theoretical Framework of Delegation to and from Agentic IS Artifacts. *MIS Quarterly*, 45(1), 315–341.
- Bajarin, T. (2024). Nvidia’s CEO On The Democratization Of Coding. <https://www.forbes.com/sites/timbajarin/2024/03/20/nvidias-ceo-on-the-democratization-of-coding/?sh=4ce975567a95>
- Bamberger, P. (2008). From the editors beyond contextualization: Using context theories to narrow the micro-macro gap in management research. *Academy of Management Journal*, 51(5), 839–846.
- Banker, R. D., & Kauffman, R. J. (1991). Reuse and productivity in integrated computer-aided software engineering: An empirical study. *MIS Quarterly*, 15(3), 375–401.
- Barke, S., James, M. B., & Polikarpova, N. (2023). Grounded copilot: How programmers interact with code-generating models. *Proceedings of the ACM on Programming Languages*, 7, 85–111.
- Benbya, H., Leidner, D. E., & Preston, D. (2019). MIS Quarterly research curation on information systems alignment.
- Benbya, H., & McKelvey, B. (2006). Using coevolutionary and complexity theories to improve IS alignment: a multi-level approach. *Journal of Information Technology*, 21, 284–298.
- Benbya, H., Strich, F., & Tamm, T. (2024). Navigating Generative Artificial Intelligence Promises and Perils for Knowledge and Creative Work. *Journal of the Association for Information Systems*, 25(1), 23–36.

- Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing Artificial Intelligence. *MIS Quarterly*, 45(3), 1433–1450.
- Berente, N., Lyytinen, K., Yoo, Y., & King, J. L. (2016). Routines as shock absorbers during organizational transformation: Integration, control, and NASA's enterprise information system. *Organization Science*, 27(3), 551–572.
- Berg, J. M., Raj, M., & Seamans, R. (2023). Capturing value from artificial intelligence. *Academy of Management Discoveries*, 9(4), 424–428.
- Bird, C., Ford, D., Zimmermann, T., Forsgren, N., Kalliamvakou, E., Lowdermilk, T., & Gazit, I. (2023). Taking flight with Copilot. *Communications of the ACM*, 66(6), 56–62.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., & Askell, A. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
- Brynjolfsson, E., Li, D., & Raymond, L. R. (2023). Generative AI at work.
- Burch, G., Lee, D., & Chen, Z. (2024). Generative AI Degrades Online Communities. *Communications of the ACM*, 67(3), 40–42.
- Burton-Jones, A., & Grange, C. (2013). From use to effective use: A representation theory perspective. *Information Systems Research*, 24(3), 632–658.
- Burton-Jones, A., Recker, J., Indulska, M., Green, P., & Weber, R. (2017). Assessing representation theory with a framework for pursuing success and failure. *MIS Quarterly*, 41(4), 1307–1334.
- Chui, M., Hazan, E., Roberts, R., Singla, A., Smaje, K., Sukharevsky, A., Yee, L., & Zimmel, R. (2023a). The economic potential of generative AI: The next productivity frontier. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier>
- Chui, M., Yee, L., Hall, B., & Singla, A. (2023b). The state of AI in 2023: generative AI's breakout year. <https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai-in-2023-generative-ais-breakout-year>
- Dell'Acqua, F., McFowland, E., Mollick, E. R., Lifshitz-Assaf, H., Kellogg, K., Rajendran, S., Kraymer, L., Candelon, F., & Lakhani, K. R. (2023). Navigating the jagged technological frontier: Field experimental evidence of the effects of AI on knowledge worker productivity and quality.
- Denning, P. J. (2023). Can generative AI bots be trusted? *Communications of the ACM*, 66(6), 24–27.
- Dwivedi, Y. K., Kshetri, N., Hughes, L., Slade, E. L., Jeyaraj, A., Kar, A. K., Baabdullah, A. M., Koohang, A., Raghavan, V., & Ahuja, M. (2023). “So what if ChatGPT wrote it?” Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy. *International Journal of Information Management*, 71, 1–63.

- Dybå, T., Sjøberg, D. I. K., & Cruzes, D. S. (2012). What works for whom, where, when, and why? On the role of context in empirical software engineering. *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 19–28.
- Eisenhardt, K. M. (1989b). Building theories from case study research. *Academy of Management Review*, 14(4), 532–550.
- Eloundou, T., Manning, S., Mishkin, P., & Rock, D. (2023). Gpts are gpts: An early look at the labor market impact potential of large language models. *arXiv preprint arXiv:2303.10130*.
- Ernst, N. A., & Bavota, G. (2022). AI-driven development is here: Should you worry? *IEEE Software*, 39(2), 106–110.
- Faraj, S., von Krogh, G., Monteiro, E., & Lakhani, K. R. (2016). Special Section Introduction Online Community as Space for Knowledge Flows. *Information Systems Research*, 27(4), 668–684.
- Foster, D. (2022). *Generative Deep Learning*. O'Reilly Media, Inc.
- Fügener, A., Grahl, J., Gupta, A., & Ketter, W. (2021). Will Humans-in-the-Loop Become Borks? Merits and Pitfalls of Working with AI. *MIS Quarterly*, 45(3), 1527–1556.
- Fügener, A., Grahl, J., Gupta, A., & Ketter, W. (2022). Cognitive challenges in human–artificial intelligence collaboration: Investigating the path toward productive delegation. *Information Systems Research*, 33(2), 678–696.
- Fürstenau, D., Baiyere, A., & Kliewer, N. (2019). A dynamic model of embeddedness in digital infrastructures. *Information Systems Research*, 30(4), 1319–1342.
- Holland, C. P., Gibson, N., Light, B., & Kelly, S. (1999). A Business Perspective of Legacy Information Systems. *Communications of the AIS*, 2(7), 1–27.
- Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., & Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12), 1–38.
- Johns, G. (2006). The essential impact of context on organizational behavior. *Academy of Management Review*, 31(2), 386–408.
- Lebovitz, S., Levina, N., & Lifshitz-Assaf, H. (2021). Is AI Ground Truth Really ‘True’? The Dangers of Training and Evaluating AI Tools Based on Experts’ Know-What. *MIS Quarterly*, 45(3), 1501–1525.
- Leonardi, P. M. (2011). When flexible routines meet flexible technologies: Affordance, constraint, and the imbrication of human and material agencies. *MIS Quarterly*, 35(1), 147–167.
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., & Rocktäschel, T. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459–9474.

- Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9), 1–35.
- Markus, M. L., & Silver, M. S. (2008). A foundation for the study of IT effects: A new look at DeSanctis and Poole’s concepts of structural features and spirit. *Journal of the Association for Information Systems*, 9(10), 609–632.
- Miceli-Barone, A. V., Barez, F., Konstas, I., & Cohen, S. B. (2023). The larger they are, the harder they fail: Language models do not recognize identifier swaps in python.
- Millière, R. (2023). *The Alignment Problem in Context*. arXiv: 2311.02147 [cs.AI]. arxiv.org
- Ngo, R., Chan, L., & Mindermann, S. (2022). The alignment problem from a deep learning perspective.
- Nguyen-Duc, A., Cabrero-Daniel, B., Przybylek, A., Arora, C., Khanna, D., Herda, T., Rafiq, U., Melegati, J., Guerra, E., & Kemell, K.-K. (2023). Generative Artificial Intelligence for Software Engineering—A Research Agenda. *arXiv preprint arXiv:2310.18648*.
- OpenAI. (2023). *GPT-4 technical report*. arXiv 2303.08774 (tech. rep.). OpenAI.
- Østerlie, T., & Monteiro, E. (2020). Digital sand: The becoming of digital representations. *Information and Organization*, 30(1).
- Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., & Wu, X. (2024). Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*.
- Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of AI on developer productivity: Evidence from GitHub Copilot. *arXiv preprint arXiv:2302.06590*.
- Pentland, B. T., Yoo, Y., Recker, J., & Kim, I. (2022). From lock-in to transformation: A path-centric theory of emerging technology and organizing. *Organization Science*, 33(1), 194–211.
- Poldrack, R. A., Lu, T., & Beguš, G. (2023). AI-assisted coding: Experiments with GPT-4. *arXiv preprint arXiv:2304.13187*.
- Rahwan, I., Cebrian, M., Obradovich, N., Bongard, J., Bonnefon, J.-F., Breazeal, C., Crandall, J. W., Christakis, N. A., Couzin, I. D., & Jackson, M. O. (2019). Machine behaviour. *Nature*, 568(7753), 477–486.
- Ram, O., Levine, Y., Dalmedigos, I., Muhlgay, D., Shashua, A., Leyton-Brown, K., & Shoham, Y. (2023). In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11, 1316–1331.
- Rinta-Kahila, T., Penttinen, E., & Lyytinen, K. (2023). Getting trapped in technical debt: Sociotechnical analysis of a legacy system’s replacement. *MIS Quarterly*, 47(1), 1–31.
- Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Sauvestre, R., & Remez, T. (2023). Code Llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.
- Russell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach*. Pearson.

- Saldaña, J. (2009). *The Qualitative Coding Manual*. Thousand Oaks.
- Scott, K. (2023). Kevin Scott on 5 Ways Generative AI Will Transform Work.
- Seidel, S., Recker, J., & Vom Brocke, J. (2013). Sensemaking and sustainable practicing: Functional affordances of information systems in green transformations. *MIS Quarterly*, 37(4), 1275–1299.
- Shypula, A., Madaan, A., Zeng, Y., Alon, U., Gardner, J., Hashemi, M., Neubig, G., Ranganathan, P., Bastani, O., & Yazdanbakhsh, A. (2023). Learning performance-improving code edits.
- Sia, S. K., & Soh, C. (2007). An assessment of package–organisation misalignment: institutional and ontological structures. *European Journal of Information Systems*, 16(5), 568–583.
- Strauss, A., & Corbin, J. (1998). *Basics of qualitative research techniques*. Sage Publications.
- Strong, D. M., & Volkoff, O. (2010). Understanding Organization—Enterprise system fit: A path to theorizing the information technology artifact. *MIS Quarterly*, 34(4), 731–756.
- Susarla, A., Gopal, R., Thatcher, J. B., & Sarker, S. (2023). The Janus effect of generative AI: Charting the path for responsible conduct of scholarly activities in information systems. *Information Systems Research*, 34(2), 399–408.
- Thomas, D., & Hunt, A. (2019). *The Pragmatic Programmer: your journey to mastery*. Addison-Wesley Professional.
- Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., & Bhosale, S. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
- Von Krogh, G., Spaeth, S., & Haefliger, S. (2005). Knowledge reuse in open source software: An exploratory study of 15 open source projects. *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 198b–198b.
- Waardenburg, L., & Huysman, M. (2022). From coexistence to co-creation: Blurring boundaries in the age of AI. *Information and Organization*, 32(4), 1–11.
- Wand, Y., & Weber, R. (1995). On the deep structure of information systems. *Information Systems Journal*, 5(3), 203–223.
- Weisz, J. D., Muller, M., Ross, S. I., Martinez, F., Houde, S., Agarwal, M., Talamadupula, K., & Richards, J. T. (2022). Better together? an evaluation of AI-supported code translation. *27th International Conference on Intelligent User Interfaces*, 369–391.
- White, J., Hays, S., Fu, Q., Spencer-Smith, J., & Schmidt, D. C. (2023). Chatgpt prompt patterns for improving code quality, refactoring, requirements elicitation, and software design. *arXiv preprint arXiv:2303.07839*.
- Wolf, Y., Wies, N., Avnery, O., Levine, Y., & Shashua, A. (2023). Fundamental limitations of alignment in large language models.

- Wu, Z., Qiu, L., Ross, A., Akyürek, E., Chen, B., Wang, B., Kim, N., Andreas, J., & Kim, Y. (2023). Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. *arXiv preprint arXiv:2307.02477*.
- Xu, F. F., Vasilescu, B., & Neubig, G. (2022). In-ide code generation from natural language: Promise and challenges. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 31(2), 1–47.
- Yin, R. K. (2009). *Case study research: Design and methods* (Vol. 5). Sage Publications.
- Yuan, Z., Lou, Y., Liu, M., Ding, S., Wang, K., Chen, Y., & Peng, X. (2023). No more manual tests? evaluating and improving chatgpt for unit test generation. *arXiv preprint arXiv:2305.04207*.
- Zhang, J., Chen, Y., Liu, C., Niu, N., & Wang, Y. (2023). Empirical Evaluation of ChatGPT on Requirements Information Retrieval Under Zero-Shot Setting. *2023 International Conference on Intelligent Computing and Next Generation Networks (ICNGN)*, 1–6.
- Zmud, R. W. (1980). Management of large software development efforts. *MIS Quarterly*, 4(2), 45–55.

Paper VI

Developers' Job Crafting and Individual Work Outcomes in Artificial Intelligence-augmented Information Systems Development

Mei Wu-Gehbauer
University of Cologne

Florian Wagner
University of Cologne

Christoph Rosenkranz
University of Cologne

Phil Hennel
University of Bremen

Completed Working Paper

Developers' Job Crafting and Individual Work Outcomes in Artificial Intelligence-augmented Information Systems Development

Abstract

Generative AI promises productivity gains, yet we still know little about how AI-augmented information systems development (ISD) shapes developers' proactive job adaptation and individual-level work outcomes. Building on the approach-avoidance, role-resource model of job crafting, this study theorizes how generative AI use in ISD relates to work engagement, job satisfaction, and work meaning, and whether job crafting mediates these relationships. We test the model via a cross-sectional survey of software developers at the internal IT provider of a large European telecommunications organization (N=98) using PLS-SEM. Results show that generative AI use is positively associated with work engagement, but not with job satisfaction or work meaning. Mediation via job crafting is not supported. However, when Copilot is used for high-complexity work, use is positively related to avoidance resource crafting. The study extends AI-augmented ISD research beyond technical productivity metrics in organizational settings.

Keywords: Generative AI, Information Systems Development, Artificial Intelligence-augmented Information Systems Development, Job Crafting, Individual Work Outcomes

1 Introduction

As generative AI becomes embedded in ISD processes as a general-purpose technology that can support a wide range of software development tasks, it is expected to reshape how software development work is carried out and, in turn, how developers experience their work (Ahmed & Devanbu, 2022; Banh et al., 2025; Barke et al., 2023; Nguyen-Duc et al., 2023; Ozkaya, 2020; Yuan et al., 2023; Zhang et al., 2023). Existing research on the impact of generative AI in ISD has primarily focused on examining technical productivity factors, which are factors related to the ISD output or process (Wagner & Ruhe, 2018). This research stream demonstrates that generative AI can increase output and speed in programming tasks (Cui et al., 2024; Peng et al., 2023; Ziegler et al., 2024). In contrast, first empirical studies on generative AI in workplace settings point to a more ambivalent picture at the level of individual workers. Generative AI can introduce new demands and technostress that undermine individual well-being (Chuang et al., 2025). Studies further show that while generative AI can improve performance, it has also been associated with decreased motivation and reduced critical thinking (Gerlich, 2025; Wu et al., 2025). These mixed findings indicate that focusing on technical factors alone is insufficient to understand the implications of generative AI-augmented ISD. Moreover, research on developers' productivity in organizational settings shows that productivity in ISD is not captured by technical factors alone (e.g., lines of code or the number of completed features).

Instead, developers' productivity and willingness to contribute depend on individual-level work outcomes, such as being satisfied with one's job and feeling enthusiastic and energized at work (Murphy-Hill et al., 2019; Storey et al., 2019). In other words, developers' individual-level outcomes are key predictors of productivity in ISD. Against this backdrop, focusing solely on the technical productivity effects of generative AI in ISD is problematic because it provides an incomplete understanding and risks overlooking situations in which technical indicators improve while individual-level outcomes may deteriorate, thereby potentially undermining the productivity gains.

Thus, to gain a more comprehensive understanding of the implications of AI-augmented ISD, it is crucial to examine how generative AI affects developers' individual-level work outcomes. Consequently, this paper investigates the impact of generative AI on the individual-level work outcomes of software developers in organizational ISD settings. Specifically, this paper focuses on three individual-level work outcomes that are central in the organizational behavior literature: work engagement, job satisfaction, and meaningful work (Bowling & Hammond, 2008; Schaufeli et al., 2006; Spreitzer, 1995).

Generative AI offers developers a broad range of possibilities for supporting their development work, but at the same time, the outputs of generative AI need to be evaluated and adapted to specific development contexts to avoid inappropriate or incorrect results (Bird et al., 2023; Huang et al., 2025; Santos et al., 2025). Qualitative IS research indicates that, as workers integrate generative AI into their work, they proactively shape their tasks, responsibilities, and collaboration patterns (Freise et al., 2025; Mayer et al., 2025a). The concept of job crafting can capture these bottom-up adjustments to the job. Job crafting refers to the self-initiated changes workers make to their jobs to better align work with their skills and needs (Wrzesniewski & Dutton, 2001). Building on the role-resource, approach-avoidance model of job crafting (Bruning & Campion, 2018), this paper distinguishes between approach crafting, in which developers expand their roles and resources to pursue valued aspects of work, and avoidance crafting, in which they reduce their roles or interactions to distance themselves from less desirable aspects of work.

Prior research shows that approach job crafting is associated with higher work engagement, job satisfaction, and work meaning, whereas avoidance crafting tends to be weakly or even negatively related to such positive outcomes (Bruning & Campion, 2018; Lichtenthaler & Fischbach, 2019; Zhang & Parker, 2019). Emerging studies on generative AI and job crafting suggest that generative AI can lead to both approach and avoidance forms of job crafting (Freise et al., 2025; Mayer et al., 2025a). This indicates that generative AI's impact on individual developers can depend on how the developers craft their jobs in AI-augmented ISD. Accordingly, this paper conceptualizes job crafting as a key mediator through which the impact of generative AI use on developers' individual-level work outcomes can be explained, with approach versus avoidance crafting expected to have different implications for job satisfaction, work engagement, and work meaning. As the relationship between generative AI use and job crafting has been exam-

ined primarily in qualitative studies so far, this paper aims to quantitatively investigate how job crafting mediates the relationship between generative AI use and developers' individual-level work outcomes in organizations. This leads to the following research question:

"How does developers' use of generative AI influence their individual-level work outcomes (work engagement, job satisfaction, and work meaning), and what is the mediating role of developers' job crafting in this relationship?"

2 Theoretical Background

2.1 Generative AI in software development

Software developers face high cognitive demands related to workload, creativity, and problem-solving (Allamanis et al., 2018; Thomas & Hunt, 2019; Zmud, 1980). Thus, software developers traditionally have looked for ways to make software development easier, for example, by actively trying to reuse code (Banker & Kauffman, 1991), moving common functionality into libraries to facilitate reuse (Von Krogh et al., 2005), or turning to online knowledge communities to get answers for a problem (Faraj et al., 2016). Generative AI, and in particular large language models (LLM), which are the focus of our study, can be of enormous benefit for software developers because coding shares many features with human language and text (Allamanis et al., 2018). Code is a highly structured form of text, and generative AI excels in learning syntactic and semantic patterns from the vast amount of text-based data it is trained on. At the same time, generative AI is not limited to specific tasks or predefined goals, as its general-purpose models apply to a broad range of developer scenarios (Nguyen-Duc et al., 2023). Generative AI-based systems can support to generate requirements (Zhang et al., 2023), translate code from one language to another (Weisz et al., 2022), suggest and generate code (Barke et al., 2023; Bird et al., 2023), generate unit tests (Yuan et al., 2023), or create documentation for code (Ahmed & Devanbu, 2022).

We conceptualize developers collaborating with generative AI systems as a generative AI use scenario (Barke et al., 2023; Ziegler et al., 2022). In this scenario, generative AI is recognized as having the capabilities and knowledge to support developers in their tasks, as well as developers' intentions and expectations to use generative AI to complete the task (Venkatesh et al., 2008). Beyond whether and how often developers use generative AI, the acceptance rate of the shown suggestions is an important predictor of improved work outcomes (Ziegler et al., 2024). Generative AI is closely integrated into developer IDEs, and code completion suggestions are shown as you type. The rate of accepting these code suggestions unchanged is a good indicator of the code quality and context understanding of generative AI (Ziegler et al., 2024).

Because generative AI systems are probabilistic, adaptive, and context-sensitive, effective use and safe delegation require continuous monitoring of generative AI outputs (Van Dis et al.,

2023). First, output quality can fluctuate with task context and prompts, leading to plausible but incorrect or insecure code (“hallucinations”), so human validation is needed (Amershi et al., 2019a). Second, the evolution of dependencies, APIs, data, and guardrails can lead to model and context drift, which can degrade performance or introduce new failure modes (Sculley et al., 2015). Third, when developers delegate parts of problem-solving to generative AI, monitoring is the ex-post coordination mechanism that maintains accountability and alignment with goals, complementing appraisal and distribution in the delegation relationship (Baird & Maruping, 2021). Taken together, monitoring behavior is a constitutive element of generative AI use in software work.

2.2 Job crafting and individual work outcomes

We employ job crafting as our theoretical foundation (Bruning & Campion, 2018; Tims & Bakker, 2010; Wrzesniewski & Dutton, 2001). Job crafting is the bottom-up and proactive behavior by employees to alter the tasks and relations of their jobs to better fit their needs (Tims & Bakker, 2010; Wrzesniewski & Dutton, 2001). Job crafting puts employees in the driver’s seat to either broaden their work roles by taking on additional tasks or extending their social relations, or reduce their work roles by reducing tasks or social interactions (Bruning & Campion, 2018). For employees to engage in job crafting, they require job resources, such as job autonomy and support (Lee et al., 2017), and demonstrate individual characteristics, such as a proactive personality and self-efficacy (Demerouti, 2014). By crafting their jobs, employees are more likely to experience positive individual work outcomes, such as enhanced job satisfaction, work meaningfulness, as well as higher work engagement. Employees have a fundamental desire to find positive meaning in their work (Rosso et al., 2010) and construct a positive identity within their organization (Dutton et al., 2010). Job crafting can help them to increase job satisfaction (Li et al., 2023), work meaning (Berg et al., 2013), and work engagement (Bakker et al., 2012).

For this study, we adopt the conceptualization of job crafting by Bruning and Campion (2018) as role and resource crafting, both for approach and avoidance forms of crafting. Approach role crafting activities are active, effortful, motivated, and directed toward problem-focused and improvement-based goals to increase one’s fit with the job through expansion (Bruning & Campion, 2018). This can include the enlargement or modification of the work role or expanding the social activities. Approach resource crafting aims to increase job resources to increase efficiency and performance. This can include the adoption of technology to enhance a work process, creating structures and strategies to organize the work, or putting oneself in a positive state of mind to make things happen. Avoidance role crafting serves the purpose of proactively and systematically reducing the work role, work requirements, effort expenditures, or task accountability. This includes avoiding tasks through delegation or reducing interactions. Avoidance resource crafting is the removal of oneself, either mentally or physically, from a person, situ-

ation, or event through changes to one's job. This may include avoiding bothersome tasks or avoiding interacting with specific people when working. This job crafting framework can help us to create a nuanced perspective on the impacts of generative AI use and its mediating impact on individual work outcomes.

Technology, such as generative AI, has recently been identified as an enabler of job crafting (Demerouti, 2022; Tarafdar & Saunders, 2022). Introducing generative AI in knowledge workers' work practices can reshape their tasks, adjust work relationships, and shift their perception of their roles (Law & Varanasi, 2025). However, technology can impact job characteristics in a positive or negative way (Demerouti, 2022; Parker & Grote, 2022). The introduction of generative AI in employees' work may also lead to negative outcomes, such as skill degradation, undermining reflexivity (Lindebaum & Fleming, 2024), and decrease the ability for critical thinking (Gerlich, 2025). This study aims to provide insights into how generative AI impacts job crafting and ultimately individual work outcomes.

3 Hypothesis Development

3.1 Impact of generative AI use on individual work outcomes

Generative AI is proposed to be a technology that has the potential to fundamentally transform the software development profession (Nguyen-Duc et al., 2023; Sætra, 2023; Sauvola et al., 2024; Ulfsnes et al., 2023). With its broad applicability for use cases across the software development lifecycle, it is expected to reshape how software development work is carried out, and how developers experience their work (Ahmed & Devanbu, 2022; Banh et al., 2025; Barke et al., 2023; Nguyen-Duc et al., 2023; Ozkaya, 2023; Zhang et al., 2023). How developers experience their work is related to several core job characteristics, which drive positive job attitude in the form of satisfaction, positive work engagement, and meaningfulness of the work (Christian et al., 2011; Hackman & Oldham, 1976). These job characteristics include drivers, such as task variety (performing different tasks in a job), problem solving (extent to which a job requires innovative solutions or new ideas), job complexity (extent to which a job is multifaceted and difficult to perform), and social support (the extent to which a job provides opportunities for assistance and advice from supervisors or coworkers) (Christian et al., 2011). For software development, generative AI directly influences these job characteristics (Thenoz et al., 2024). Generative AI can increase autonomy and problem-solving opportunities by giving developers suggestions and explanations, allowing them to decide which solutions to adopt and enabling them to tackle more complex or novel tasks (Ulfsnes et al., 2023). Generative AI can also raise task variety and job complexity, because developers can move away from repetitive boilerplate coding toward more challenging problems (Barke et al., 2023). Generative AI also acts as an immediate feedback and knowledge resource, enabling developers to get assistance about alternative implementations, potential errors, or better coding patterns (Nguyen-Duc et al., 2023).

As a result of the influence of generative AI on the job characteristics of developers, the use of generative AI may be positively linked to improved individual-level work outcomes, including work engagement, job satisfaction, and work meaning, all of which are widely accepted to be positively correlated with work performance (Christian et al., 2011; Judge et al., 2001; Storey et al., 2019). Work engagement is defined as developers feeling energized, absorbed, and dedicated at work (Schaufeli et al., 2006). Engaged employees experience more positive emotions, which are positively correlated with work performance (Corbeanu & Iliescu, 2023). Job satisfaction is defined as the extent to which employees have a positive affective reaction toward their job and is also a strong predictor of improved performance (Judge et al., 2001). Finally, employees have a fundamental desire to find meaning in their work (Rosso et al., 2010). When employees experience their work as meaningful (i.e., significant, challenging, and complete), it contributes positively to work performance (Rosso et al., 2010). Together, we therefore hypothesize:

Hypothesis 1. *Developers' use of generative AI is positively related to the individual-level work outcomes of work engagement, job satisfaction, and work meaning.*

3.2 The mediating effect of job crafting

With generative AI integrated in the ISD process, the way developers work may fundamentally change (Welsh, 2022). What previously has been the core responsibility of developers, writing code, is now shifting to reviewing AI-generated code or collaborating with generative AI assistants (Barke et al., 2023). Because such tools can be seamlessly embedded into existing workflows and offer general-purpose support across many tasks, developers rapidly incorporate generative AI into their everyday work practices (Microsoft, 2025). Generative AI can impact developers' jobs in two ways. On the one hand, developers expand or redefine their roles, for example, by taking on more complex work or leveraging generative AI as a coding partner (Dakhel et al., 2023; Imai, 2022). At the same time, generative AI can help developers reduce interruptions in their development process, allowing them to tackle more challenging problems and expand their skills as a result (Shah et al., 2025; Stray et al., 2025). Generative AI also enables automation of routine coding tasks, allowing developers to shift their attention to problem-solving of more significant and rewarding tasks (Ulfsnes et al., 2023). This proactive adaptation of their jobs may lead to approach-oriented crafting behavior through which the benefits of generative AI use are translated into more positive individual-level work outcomes. On the other hand, generative AI can lead to withdrawal from demanding aspects of their roles or bypass learning and deep problem-solving (Cheng et al., 2023; Yan et al., 2025). These changes to the job create conditions under which avoidance-oriented crafting behavior is likely to occur (Tims et al., 2012; Zhang & Parker, 2019). Such avoidance-oriented behavior is a pathway through which generative AI use can result in less favorable individual-level work outcomes, because it has longer-term negative implications for skill development and psychological attachment to the job (Bruning & Campion, 2018).

Job crafting theory conceptualizes these changes in approach and avoidance crafting as central mechanisms through which employees adapt to changing environments and shape their own work outcomes (Bruning & Campion, 2018). Building on this literature, we conceptualize approach role and resource crafting, and avoidance role and resource crafting as mediating mechanisms that translate the effects of generative AI use to developers' work outcomes. Accordingly, we hypothesize:

Hypothesis 2. *Approach role and resource crafting positively mediate the relationship between generative AI use and the individual-level work outcomes of job satisfaction, work engagement, and work meaning.*

Hypothesis 3. *Avoidance role and resource crafting negatively mediate the relationship between generative AI use and work outcomes of job satisfaction, work engagement, and work meaning.*

3.3 The moderating effects

Job complexity is a central job characteristic in job crafting research (Bai et al., 2021; Tian et al., 2022). It relates to the extent to which work is non-routine, requires independent judgment, and involves problem-solving and creativity rather than repetitive execution (Gould, 1979). Prior studies show that higher job complexity motivates workers to proactively adjust their jobs, both by expanding and enriching their roles and resources (approach crafting) and by withdrawing from particularly demanding aspects (avoidance crafting) (Bai et al., 2021; Tian et al., 2022). Complex jobs provide more opportunities and reasons to apply generative AI, thereby strengthening the link between generative AI use and job crafting behavior.

When developers apply generative AI to more complex parts of their work, they are more likely to see generative AI as a lever to reshape their jobs. At the same time, complexity can also heighten uncertainty and strain, making it more attractive to rely on generative AI to shield oneself from cognitively demanding aspects of the job; thus, generative AI use in complex work may more strongly support avoidance role and resource crafting, for instance, by delegating difficult reasoning or learning tasks to the system.

In other words, we expect that when the generative AI-supported work is more complex, the association between generative AI use and both approach and avoidance forms of crafting should be stronger. Accordingly, we model job complexity of generative AI-augmented work as a moderator of the relationship between generative AI use and the four job crafting dimensions and hypothesize that the effect of generative AI use on approach and avoidance role and resource crafting is amplified at higher levels of job complexity. We therefore hypothesize:

Hypothesis 4. *Job complexity of AI-supported work moderates the effect of generative AI use on approach and avoidance role and resource crafting in a way that the effect is stronger for high-complexity AI-supported work.*

When a new technology is adopted, professionals must often reconfigure their professional roles (Lifshitz-Assaf, 2018). AI's ability to take over complete work processes forces developers to renegotiate their occupational boundaries and rethink the meaning they attach to the questions "who am I as a software developer"? and "what do I do"?, which define their professional role identity (Faraj et al., 2018; Strich et al., 2021). Identity theory suggests that individuals actively strive to maintain identity congruence when afforded new resources or when faced with threats to their identity (Chreim et al., 2007; Reay et al., 2017). Accordingly, professional role identity should shape how developers translate generative AI use into job crafting responses. Developers with a strong role identity are more motivated to ensure that new tools support, rather than undermine, who they are at work (Barke et al., 2023; Riemenschneider & Armstrong, 2021). At the same time, a strong role identity should reduce avoidance responses to AI-augmented work. Developers with a strong professional role identity should be less likely to respond through withdrawal, because maintaining identity congruence motivates them to stay engaged and retain ownership of core responsibilities. Thus, professional role identity should strengthen approach crafting responses to GenAI use while reducing the tendency for generative AI use to translate into avoidance crafting. In other words, a strong professional role identity should moderate the effect of GenAI use on approach and avoidance role and resource crafting as follows:

Hypothesis 5a. *Professional role identity moderates the effect of generative AI use on approach job crafting in a way that a high professional role identity strengthens the effect of generative AI use on approach roles and resource crafting.*

Hypothesis 5b. *Professional role identity moderates the effect of generative AI use on avoidance job crafting in a way that a high professional role identity reduces the effect of generative AI use on avoidance roles and resource crafting.*

In AI-assisted programming, the acceptance rate of code suggestions is a behavioral signal of the fit between the generative AI output and the developer's goal and context (Ziegler et al., 2024). When generative AI suggestions are consistently accepted, this reflects a perceived usefulness of the code suggestions and their fit with the code context. A high acceptance rate, therefore, signals that generative AI is a dependable resource.

From a job crafting perspective, employees are more likely to reshape their work around technologies they perceive as reliable and beneficial. Job crafting research emphasizes that workers proactively alter their roles and resources when they perceive opportunities to gain valued job resources or manage demands more effectively (Bruning & Campion, 2018; Tims et al., 2012; Wrzesniewski & Dutton, 2001). When the acceptance rate is high, generative AI use reliably produces usable output, allowing developers not only to expand their roles and resources (approach crafting) but also to step back from tasks and learning efforts they find draining. Under these conditions, generative AI use should more strongly support job crafting behavior. Conversely, when acceptance rate is low and most suggestions are discarded, generative AI is experienced less as a resource and is less likely to translate into job crafting behavior. Thus, we

hypothesize:

Hypothesis 6. *Acceptance rate moderates the effect of generative AI use on job crafting approach and avoidance behavior in a way that the effect is stronger for higher acceptance rate.*

The extent to which these crafting behaviors translate into work outcomes depends on the context in which crafting occurs (Dierdorff & Jensen, 2018; Wrzesniewski & Dutton, 2001). In generative AI-supported work, employees often must monitor the AI's output for correctness and appropriateness to the context. Studies of AI coding assistants show developers routinely verify, edit, and test suggestions due to hallucinations, logic errors, and style or dependency mismatches, which adds verification time and cognitive load (Huang et al., 2025). Even when generative AI accelerates some steps, the overhead of scrutiny can offset or negate net benefits for work outcomes, particularly when suggestions are frequent or accuracy varies frequently. Information systems research further suggests that technology-driven oversight demands can trigger techno-overload and techno-complexity, which are linked to reduced job satisfaction, productivity, and work engagement (Benlian, 2020).

Taken together, this literature implies that when monitoring behavior is high, crafting behaviors, which may be experienced as extra coordinative and evaluative work (double-checking, testing, documenting), reduces their impact on satisfaction, engagement, and meaning. When monitoring requirements are low (e.g., high trust, familiar tasks), the benefits of crafting should be more fully realized. Thus, we hypothesize:

Hypothesis 7. *Monitoring behavior moderates the effects of job crafting dimensions on work outcomes in a way that the effect is weaker for higher monitoring behavior.*

4 Study Method

4.1 Research model

Figure 1 illustrates our research model. In line with our hypotheses, the model links developers' generative AI use to three individual-level work outcomes (work engagement, job satisfaction, and work meaning), with approach and avoidance job crafting specified as mediating mechanisms. In addition, professional role identity, acceptance rate, job complexity, and monitoring behavior are modeled as moderators.

4.2 Sample and procedure

We examined the hypotheses employing a cross-sectional survey design. Cross-sectional surveys are appropriate when the research objective is to estimate how constructs relate at a particular point in time for the population of interest (Pinsonneault & Kraemer, 1993). This design

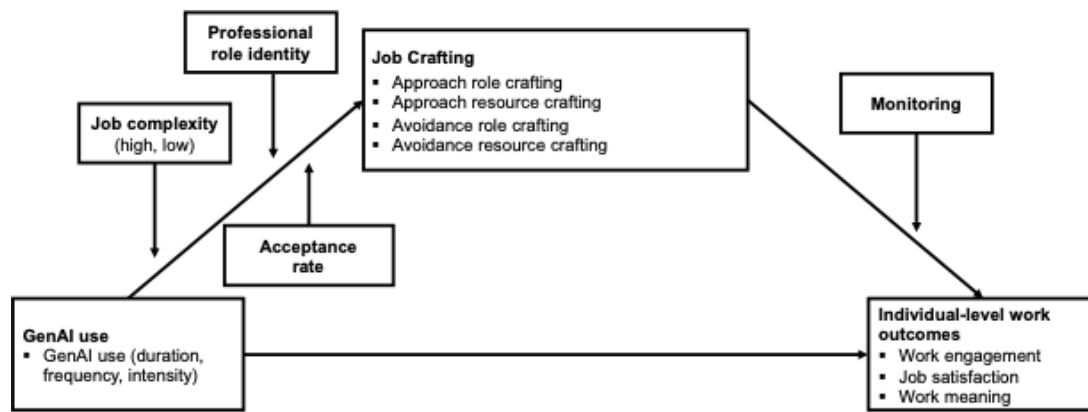


Figure 1: Research model

aligns with the research objective of this study to relate developers' generative AI use to individual work outcomes and to examine job crafting as a mediator.

The participants of the survey are employees of a large European telecommunications organization. The participating firm has over 200,000 employees and serves both private customers with broadband, mobile, internet, and internet-based TV products and services, as well as business customers with ICT solutions. Participants are all members of the internal IT service provider in active software development teams. The development teams are responsible for the design, implementation, provision, and maintenance of all IT systems and IT infrastructure services required to support the business of the organization across all its business segments. The case organization introduced GitHub Copilot at the beginning of 2023, following the launch of ChatGPT at the end of 2022.

Data were collected over a period of four months (May - August 2025) following a pilot survey to validate the clarity of the questions and the time needed to complete the survey. As a result of the pilot survey, we explicitly included a descriptive definition of the job crafting construct to ensure participants can easily understand it. We recruited 508 full-time employees from the case company. Consistent with the study's focus, participants were selected using purposive sampling: participation was open only to employees who (1) currently held a software development role and (2) actively used generative AI for development tasks during the period specified in the survey instructions. Before starting the questionnaire, participants were provided with an information sheet outlining the study purpose, voluntary participation, confidentiality, and intended data use. From the employees to whom we sent the survey, we received 98 complete responses (19%). All respondents worked as software engineers in the case company. More than 60% of respondents had used GitHub Copilot for more than 6 months.

To estimate the relationships among constructs, we applied partial least squares structural equation modeling (PLS-SEM). PLS-SEM is a variance-based SEM approach that estimates latent variable scores as weighted composites of their indicators and uses these scores to estimate the structural model (Chin, 1998; Hair et al., 2017). PLS-SEM is suitable for this study because

it supports the simultaneous estimation of a measurement model and a structural model with multiple interrelated paths, including mediation (indirect paths), thereby enabling a statistically grounded test of the proposed framework.

Model evaluation followed established PLS-SEM guidance by assessing the measurement models first and interpreting the structural model only after satisfactory measurement quality was established (Hair et al., 2017).

4.3 Measures

The details of the scales and corresponding sources are listed in the Appendix. Scales were adapted from existing scales to suit the context of generative AI. All respondents answered items on a scale from 1 (strongly disagree) to 7 (strongly agree). Exceptions are highlighted accordingly.

Generative AI use was measured on a three-item scale adapted from Venkatesh et al. (2008) containing frequency of use, intensity of use, and Copilot use in hours per week (Cronbach's $\alpha = 0.815$). An example question regarding generative AI use is, "On average, how many hours do you use GitHub Copilot each week?"

Acceptance rate was measured as the acceptance rate of Copilot suggestions adapted from Ziegler et al. (2022) on a scale from 0 to 100% (with 5% steps) with the question "How often do you accept suggestions of GitHub Copilot unchanged (in percent)?"

Job complexity was measured on the job complexity scale from Gould (1979) and included two constructs for handling low complex tasks (Cronbach's $\alpha = 0.642$) and handling high complex tasks (Cronbach's $\alpha = 0.786$). An example question for low complex tasks is, "GitHub Copilot performs routine work consisting of simple repetitive operations" and for high complex tasks is, "GitHub Copilot plans and performs highly complex or technical work where no procedures or standards are available."

Job crafting was measured on the sub-dimensions for approach and avoidance crafting from Lopper et al. (2024) and Bruning and Campion (2018), containing seven items. Work role expansion (WRE) was measured on a five-item scale (Cronbach's $\alpha = 0.895$). An example question regarding WRE is, "I expand my work by adding activities to my job that ensure the quality of my deliverables." Social expansion (SOE) was measured on a four-item scale (Cronbach's $\alpha = 0.913$). An example question regarding SOE is, "I actively initiate positive interactions with others at work." Together, WRE and SOE form the construct approach role crafting (Cronbach's $\alpha = 0.911$). Work organization (WOG) was measured on a four-item scale (Cronbach's $\alpha = 0.882$). An example question regarding WOG is, "I create structure in my work processes." Adoption (ADO) was measured on a five-item scale (Cronbach's $\alpha = 0.827$). An example question regarding ADO is, "I use new knowledge or technology to automate tasks." Metacognition

(META) was measured on a five-item scale (Cronbach's $\alpha = 0.906$). An example question regarding META is, "I use my thoughts to help me focus and be engaged at work." Together, WOG, ADO and META form the construct approach resource crafting (Cronbach's $\alpha = 0.911$). Work role reduction (WRR), which equals avoidance role crafting, was measured on a four-item scale (Cronbach's $\alpha = 0.746$). An example question regarding WRR is, "I find ways to bypass time-consuming tasks." Withdrawal (WITH), which equals avoidance resource crafting, was measured on a three-item scale (Cronbach's $\alpha = 0.813$). An example question regarding WITH is, "I work in a way that allows me to avoid bothersome tasks involved in my work."

Individual-level work outcomes were measured by job satisfaction, work meaning, and work engagement. Job satisfaction was measured on a three-item scale adopted from Lawler et al. (1979). An example question about job satisfaction is: "All in all, I am satisfied with my job." Work meaning was measured on a three-item scale from Spreitzer (1995). An example question about work meaning is: "The work I do is meaningful." Work engagement was measured on a nine-item scale from Schaufeli et al. (2006). Example questions are: "At my work, I feel that I am bursting with energy" or "My job inspires me."

Professional role identity was measured by an eight-item scale by Riemenschneider and Armstrong (2021). An example question is: "I think and act like other software developers." Monitoring behavior was measured by a six-item scale by McAllister (1995). An example question is: "The quality of the work I receive from GitHub Copilot is only maintained by my diligent monitoring."

Constructs from the job demand and job resource model were also included as control variables for job crafting, as they represent individual and situational motivation factors for job crafting (Demerouti, 2014). Items included for individual characteristics are proactivity (Cronbach's $\alpha = 0.794$) and self-efficacy (Cronbach's $\alpha = 0.890$). Proactivity is measured on a six-item scale adapted from Teye-Kwadjo and de Bruin (2022) and self-efficacy on a three-item scale adapted from Spreitzer (1995). Items included for the situational job factors include job autonomy (Cronbach's $\alpha = 0.940$) and workload (Cronbach's $\alpha = 0.916$). Job autonomy was measured on a three-item scale adapted from Spreitzer (1995) and workload was measured on a three-item scale adapted from Venkatesh et al. (2020).

Further control variables included organizational tenure, job role level, and attitude towards AI.

Further control variables included organizational tenure, job role level, and attitude toward AI to account for alternative explanations of developers' job crafting and work outcomes. Organizational tenure captures accumulated firm-specific domain expertise, which can shape how confidently developers integrate new technologies and proactively redesign their work (Ng & Feldman, 2010). Job role level (e.g., junior vs. senior developers) reflects systematic differences in autonomy, task scope, and accountability that are known to have an impact on work outcomes (Brynjolfsson et al., 2023). Finally, attitude toward AI captures acceptance-related beliefs and

trust tendencies that influence whether developers approach generative AI as an enabling resource or experience it as uncertain, thereby affecting both crafting responses and subsequent work outcomes (Venkatesh et al., 2003).

4.4 Evaluation of reflective measurement model

To evaluate our reflective measurement model, we assess indicator reliability, internal consistency reliability, convergent reliability, and discriminant validity (Hair, 2014). The first step is to assess indicator reliability, which examines how much of each indicator's variance is explained by its construct. Acceptable indicator loadings should be above 0.70, meaning that they indicate that the construct explains more than 50 percent of the indicator's variance. All indicators were above the threshold except one indicator for professional role identity (pri7) ($\beta = 0.526$). However, the indicator was statistically significant, and the overall constructs met the reliability and validity criteria; we kept this indicator. The second step is to examine internal consistency reliability, which is the extent to which indicators measuring the same construct are associated with one another. This is measured with Cronbach's alpha (α) and the composite reliability rhoC (ρC) (Jöreskog, 1971). Values ranging from 0.70 to 0.95 are acceptable; values above 0.95 are problematic since they indicate that the indicators are redundant. We also report rhoA (ρA) as a consistent reliability coefficient recommended for PLS-SEM. It typically lies between Cronbach's α and ρC and complements both. The Cronbach α and ρC of all constructs are in the range from 0.70 to 0.95, except for monitoring behavior ($\alpha = 0.538$) and job complexity low ($\alpha = 0.642$). Given the theoretical importance of both constructs and the significance of their indicators, we kept both constructs. The third step is to assess the convergent validity of each construct, which is the extent to which the constructs converge in explaining the variance in their indicators. This is measured with the average variance extracted (AVE). The AVE is defined as the mean value of the squared loadings of the indicators (Hair, 2014). The minimum acceptable AVE is 0.50. All constructs are above this threshold. Finally, discriminant validity needs to be assessed, which measures the extent to which a construct is empirically distinct from other constructs in the structural model. To assess discriminant validity, we use the Fornell-Larcker criterion and heterotrait-monotrait ratio (HTMT) of correlations (Fornell & Larcker, 1981; Henseler et al., 2015). The Fornell-Larcker criterion requires that the square root of each construct's AVE exceeds its correlations with all other constructs, indicating that a construct shares more variance with its own indicators than with other constructs. The Fornell-Larcker criterion is met for all constructs except for approach resource crafting, where the $\sqrt{AVE}$ of 0.720 is lower than the correlation with approach role crafting (0.786). All HTMT values of our constructs are below the more conservative threshold of 0.85. In particular, the HTMT value between approach resource crafting and approach role crafting is 0.840, and remained below the recommended threshold of 0.850, suggesting that the constructs are empirically distinct despite conceptual proximity. Therefore, we argue that the discriminant validity of our measurement

model is supported. Together, the measurement model meets the quality criteria for reliability and validity (Table 1).

Table 1: Construct reliability and validity of reflective measures

Indicator	Alpha	ρC	AVE	ρA
Generative AI use	0.815	0.889	0.729	0.836
Self-efficacy	0.890	0.932	0.821	0.933
Proactivity	0.794	0.849	0.491	0.837
Job autonomy	0.940	0.962	0.893	0.943
Workload	0.916	0.945	0.852	0.983
Job complexity (low)	0.642	0.847	0.735	0.652
Job complexity (high)	0.786	0.899	0.817	0.902
Acceptance rate	1.000	1.000	1.000	1.000
Professional role identity	0.858	0.895	0.592	0.886
Approach role crafting	0.911	0.928	0.616	0.913
Approach resource crafting	0.911	0.924	0.506	0.914
Avoidance role crafting	0.746	0.836	0.569	0.810
Avoidance resource crafting	0.813	0.891	0.733	0.867
Monitoring behavior	0.538	0.812	0.684	0.539
Work meaning	0.878	0.925	0.805	0.880
Job satisfaction	0.720	0.838	0.634	0.754
Work engagement	0.937	0.948	0.695	0.939

4.5 Evaluation of structural measurement model

To evaluate our structural model, we assess potential collinearity issues, the model's significance and relevance, its explanatory power, and its predictive power (Hair, 2014).

First, we assess potential collinearity by calculating the variance inflation factor (VIF) for each predictor construct relative to each endogenous construct. Our model has four endogenous constructs (approach and avoidance role crafting, approach and avoidance resource crafting), and all are clearly below the conservative threshold of 3. Therefore, we conclude that collinearity is not an issue in our structural model.

Next, we evaluate the relevance and significance of the structural paths. To this end, we first assess the bootstrapped path for the exogenous driver constructs. We find that generative AI use has a direct positive and significant effect on work engagement ($\beta = 0.159$, $SD=0.081$, 95% CI [0.003, 0.306]).

Reviewing the bootstrapped paths for the endogenous constructs yields the following results. Approach role crafting has a substantial positive effect on job satisfaction ($\beta = 0.486$, $SD=0.209$,

95% CI [0.036, 0.858]), work engagement ($\beta = 0.635$, $SD=0.134$, 95% CI [0.346, 0.879]) and work meaning ($\beta = 0.644$, $SD=0.120$, 95% CI [0.390, 0.863]). All of these three impacts are statistically significant. Avoidance role crafting has a negative impact on work meaning ($\beta = -0.208$, $SD=0.080$, 95% CI [-0.363, -0.048]). Avoidance resource crafting has a negative impact on work meaning ($\beta = -0.220$, $SD=0.073$, 95% CI [-0.364, -0.079]), statistically significant.

Reviewing the moderating effects, we found that high job complexity moderates the relationship between generative AI use and avoidance resource crafting ($\beta = 0.269$, $SD = 0.128$, 95% CI [0.017, 0.514]). The acceptance rate and professional role identity does not have a statistically significant moderating effect on the relationship between generative AI use and job crafting. Monitoring behavior has no moderating effect on the relationship between job crafting and work outcomes.

In the next step, we evaluate the explanatory power of the structural model by analyzing the R^2 of the endogenous constructs of the predictor constructs. The R^2 values of approach role crafting ($R^2 = 0.479$), approach resource crafting ($R^2 = 0.394$), avoidance role crafting ($R^2 = 0.205$), and avoidance resource crafting ($R^2 = 0.305$) can be considered moderate given the complexity of the research model.

Finally, we test the predictive power of our structural model, i.e., its ability to predict new or future observations. For this, we use PLS predict, which executes k-fold cross-validation on our data to mimic how our model will eventually be used to predict a new observation. All our indicators have lower root mean squared errors compared to the linear regression benchmark model and thus have a high predictive power.

5 Study Results

Table 2 presents the descriptive statistics and correlations of the variables used in our model. As expected, the job crafting dimensions are moderately to highly correlated (Bruning & Campion, 2018; Tims et al., 2012). While the job crafting dimensions are similar in some ways, the four job crafting sub-dimensions offer more detailed insights into the different aspects of job crafting. To address potential concerns, we conducted a multicollinearity analysis. The results showed an average variance inflation factor of 2.64 and a maximum value of 6.08. The results strongly suggest that multicollinearity is not affecting our results, as it is below the cut-off value of 10 (Chatterjee & Simonoff, 2013).

5.1 Test of hypothesis

Table 3 presents the relevance and significance of the structural paths. For this, we first evaluate the bootstrapped path (5,000 iterations) for the exogenous driver constructs. Table 3 shows that all impacts on job crafting dimensions are not significant. Generative AI use has a positive

Table 2: Means, standard deviation, and correlations

Variable	Mean	SD	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
(1) Generative AI use	3.31	0.96	1.00																			
(2) Workload	3.86	1.42	0.16	1.00																		
(3) Job autonomy	5.61	0.98	0.08	-0.07	1.00																	
(4) Self-efficacy	5.75	0.99	0.03	0.00	0.59	1.00																
(5) Proactivity	5.19	0.76	0.09	-0.05	0.47	0.66	1.00															
(6) Attitude AI	3.34	0.70	0.27	-0.07	0.15	0.12	0.19	1.00														
(7) Organizational tenure	3.48	0.70	0.23	0.18	0.13	0.08	0.07	0.07	1.00													
(8) Job role level	2.28	0.77	0.12	0.12	0.05	0.18	0.10	-0.27	0.15	1.00												
(9) Job complexity low	5.35	0.98	0.20	-0.14	0.24	0.32	0.37	0.21	0.00	0.11	1.00											
(10) Job complexity high	3.51	1.29	-0.12	0.06	-0.01	-0.04	0.01	0.19	0.11	-0.16	0.01	1.00										
(11) Acceptance rate	3.51	0.84	0.28	0.05	-0.09	-0.17	-0.03	0.32	0.09	-0.18	0.00	0.21	1.00									
(12) Professional role identity	5.55	0.83	0.01	-0.10	0.39	0.57	0.59	0.11	0.05	0.13	0.27	-0.02	-0.16	1.00								
(13) Approach role crafting	4.98	0.96	0.02	-0.05	0.37	0.55	0.63	0.16	0.13	0.07	0.25	0.04	-0.15	0.56	1.00							
(14) Approach resource crafting	5.13	0.84	0.00	-0.07	0.38	0.51	0.54	0.13	0.17	0.02	0.27	-0.06	-0.11	0.57	0.78	1.00						
(15) Avoidance role crafting	3.09	1.09	0.12	0.24	0.04	-0.10	-0.02	0.08	0.24	0.02	-0.08	0.22	0.02	-0.07	0.16	0.19	1.00					
(16) Avoidance resource crafting	3.17	1.26	0.04	0.10	0.04	-0.13	-0.12	-0.12	-0.01	0.17	-0.11	0.08	-0.13	-0.40	-0.19	-0.21	0.21	1.00				
(17) Monitoring behavior	4.46	1.04	0.12	0.03	0.23	0.19	0.24	0.10	-0.13	-0.02	0.06	0.11	-0.03	0.22	0.33	0.10	0.06	0.02	1.00			
(18) Job satisfaction	4.89	1.04	-0.03	-0.26	0.40	0.27	0.30	0.12	0.11	-0.01	0.16	-0.02	-0.10	0.48	0.48	0.35	0.01	-0.17	0.25	1.00		
(19) Work engagement	5.35	1.19	0.15	-0.12	0.37	0.43	0.55	0.24	0.11	0.03	0.30	0.06	-0.12	0.48	0.72	0.60	0.18	-0.09	0.30	0.58	1.00	
(20) Work meaning	5.79	0.99	0.02	-0.15	0.36	0.47	0.59	0.19	0.11	-0.01	0.29	0.01	-0.07	0.64	0.61	0.44	-0.14	-0.38	0.30	0.53	0.64	1.00

effect on work engagement ($\beta = 0.159$, $SD=0.077$, 95% CI [0.003, 0.306]). Reviewing the bootstrapped paths for the endogenous constructs yields the following results. Approach role crafting has a substantial positive impact on job satisfaction ($\beta = 0.486$, $SD=0.209$, 95% CI [0.036, 0.858]), work engagement ($\beta = 0.635$, $SD=0.134$, 95% CI [0.346, 0.879]) and work meaning ($\beta = 0.644$, $SD=0.120$, 95% CI [0.390, 0.863]). Avoidance role crafting has a negative impact on work meaning ($\beta = -0.208$, $SD=0.080$, 95% CI [-0.363, -0.048]). Avoidance resource crafting has a negative impact on work meaning ($\beta = -0.220$, $SD=0.073$, 95% CI [-0.364, -0.079]), statistically significant.

Reviewing the moderating effects, we found that high job complexity has a moderating effect on the relationship between generative AI use and avoidance resource crafting ($\beta = 0.269$, $SD=0.128$, 95% CI [0.017, 0.514]). Acceptance rate or professional role identity has no moderating impact on the relationship between generative AI use and job crafting.

Table 3: Coefficients for PLS-SEM

Variables	AppRoC	AppReC	AvRoC	AvReC	Job Satisfaction	Work Engagement	Work Meaning
Generative AI use	-0.010	-0.070	0.099	0.173	-0.022	0.159*	0.030
Generative AI use*Job complexity high	-0.108	-0.022	-0.011	0.269**			
Generative AI use*Job complexity low	-0.073	-0.016	-0.203	-0.162			
Generative AI use*Acceptance rate	-0.026	0.065	0.142	-0.166			
Generative AI use*Professional role identity	-0.048	-0.061	0.247	0.110			
Proactivity	0.393***	0.250**	0.021	0.068			
Workload	-0.019	-0.011	0.207	0.043			
Job autonomy	-0.015	0.038	0.196	0.309**			
Self-efficacy	0.142	0.107	-0.229	-0.089			
Attitude	0.067	0.026	0.075	-0.090			
Job role level	0.000	-0.091	0.045	0.155			
Organizational tenure	0.094	0.173	0.108	-0.081			
Approach role crafting (AppRoC)					0.486*	0.635**	0.644**
Approach resource crafting (AppReC)					-0.041	0.062	-0.062
Avoidance role crafting (AvRoC)					-0.012	0.051	-0.208*
Avoidance resource crafting (AvReC)					-0.097	0.004	-0.220**
Approach role crafting*Monitoring behavior					-0.136	-0.071	-0.203
Approach resource crafting*Monitoring behavior					0.068	-0.055	0.150
Avoidance role crafting*Monitoring behavior					-0.114	-0.023	0.064
Avoidance resource crafting*Monitoring behavior					-0.059	-0.073	0.011

Notes. N=98, ***p<0.001, **p<0.01, *p<0.05

Table 3 shows partial support for Hypothesis 1, which predicted a positive impact of generative AI use on work engagement ($\beta = 0.159$, $SD=0.081$, 95% CI [0.003, 0.306]). Impacts on job satisfaction and work meaning could not be supported. Our Hypothesis 2 proposed that there is a positive mediating effect of approach job crafting on the relationship between generative AI use and individual-level work outcomes. Hypothesis 3 proposed that there is a negative mediating

effect of avoidance job crafting on the relationship between generative AI use and individual-level work outcomes. However, the results do not support these mediation hypotheses. Indirect paths of generative AI use on job satisfaction are small and not significant. Total-effects analysis shows no uniform overall mediating impact (Table 4).

Table 4: Bootstrapped results and confidence intervals of estimated paths of direct effect and mediation

Estimated Path	Type of Effect	Coeff	SD	95% CI	
H1. Job satisfaction $\leftarrow$ Generative AI use	Direct effect	-0.022	-0.107	-0.237	0.178
H1. Work engagement $\leftarrow$ Generative AI use	Direct effect	0.159*	0.081	0.003	0.306
H1. Work meaning $\leftarrow$ Generative AI use	Direct effect	0.030	0.084	-0.138	0.186
H2. Approach role crafting $\leftarrow$ Generative AI use	Leg 1 of mediation	-0.010	0.121	-0.262	0.217
H2. Approach resource crafting $\leftarrow$ Generative AI use	Leg 1 of mediation	-0.070	0.120	-0.296	0.178
H3. Avoidance role crafting $\leftarrow$ Generative AI use	Leg 1 of mediation	0.099	0.127	-0.175	0.337
H3. Avoidance resource crafting $\leftarrow$ Generative AI use	Leg 1 of mediation	0.173	0.125	-0.097	0.390
H2. Job satisfaction $\leftarrow$ Approach role crafting	Leg 2 of mediation	0.486*	0.209	0.036	0.858
H2. Work engagement $\leftarrow$ Approach role crafting	Leg 2 of mediation	0.635**	0.134	0.346	0.879
H2. Work meaning $\leftarrow$ Approach role crafting	Leg 2 of mediation	0.644**	0.120	0.390	0.863
H2. Job satisfaction $\leftarrow$ Approach resource crafting	Leg 2 of mediation	-0.041	0.197	-0.408	0.376
H2. Work engagement $\leftarrow$ Approach resource crafting	Leg 2 of mediation	-0.062	0.136	-0.190	0.341
H2. Work meaning $\leftarrow$ Approach resource crafting	Leg 2 of mediation	-0.062	0.124	-0.298	0.185
H3. Job satisfaction $\leftarrow$ Avoidance role crafting	Leg 2 of mediation	-0.012	0.102	-0.194	0.197
H3. Work engagement $\leftarrow$ Avoidance role crafting	Leg 2 of mediation	0.051	0.088	-0.125	0.222
H3. Work meaning $\leftarrow$ Avoidance role crafting	Leg 2 of mediation	-0.208*	0.080	-0.363	-0.048
H3. Job satisfaction $\leftarrow$ Avoidance resource crafting	Leg 2 of mediation	-0.097	0.114	-0.321	0.121
H3. Work engagement $\leftarrow$ Avoidance resource crafting	Leg 2 of mediation	0.004	0.088	-0.187	0.157
H3. Work meaning $\leftarrow$ Avoidance resource crafting	Leg 2 of mediation	-0.220*	0.072	-0.364	-0.079
H2. Work engagement $\leftarrow$ Generative AI use * Approach role crafting	Indirect effect	-0.017	0.080	-0.197	0.133
H3. Work meaning $\leftarrow$ Generative AI use * Avoidance role crafting	Indirect effect	-0.009	0.019	-0.058	0.022
H3. Work meaning $\leftarrow$ Generative AI use * Avoidance resource crafting	Indirect effect	-0.039	0.033	-0.110	0.014
H2. Work engagement $\leftarrow$ (AppRoC + Generative AI use * AppRoC)	Total effects	0.158	0.145	-0.082	0.350
H2. Work meaning $\leftarrow$ (AppRoC + Generative AI use * AppRoC)	Total effects	-0.031	-0.041	-0.243	0.152
H4. Avoidance resource crafting $\leftarrow$ Generative AI use * Job complexity high	Moderating effect	0.269**	0.128	0.017	0.514

5.2 Additional analysis

With the additional analyses we aimed to investigate whether the effect on job satisfaction, work engagement, and work meaning is driven by alternative constructs besides generative AI use. Therefore, we conducted a series of regressions, which tested whether individual characteristics (proactivity, self-efficacy) and job characteristics (autonomy, workload) have an effect on individual work outcomes mediated through job crafting. The results are shown in Table 5.

The regressions show strong support that proactive behavior drives work engagement and work meaning mediated by job approach role crafting behavior. The indirect effect on work engagement positive ($\beta = 0.312$, $SD=0.106$, 95% CI [0.113, 0.529]) and work meaning ($\beta = 0.302$, $SD=0.097$, 95% CI [0.063, 0.390]) is significant and positive. This provides support that job crafting mediates the relationship between proactive behavior and work engagement and work meaning.

Table 5: Bootstrapped results and confidence intervals of additional estimated paths

Estimated Path	Type of Effect	Coeff	SD	95% CI	
Work meaning $\leftarrow$ Self-efficacy	Direct effect	0.160	0.171	0.005	0.357
Work engagement $\leftarrow$ Proactivity	Direct effect	0.317	0.308	0.115	0.518
Job satisfaction $\leftarrow$ Professional Role Identity	Direct effect	0.421	0.418	0.111	0.694
Work meaning $\leftarrow$ Professional Role Identity	Direct effect	0.356	0.359	0.157	0.540
Work meaning $\leftarrow$ Proactivity	Direct effect	0.229	0.222	0.063	0.390
Approach role crafting $\leftarrow$ Proactivity	Leg 1 of mediation	0.474	0.473	0.225	0.709
Approach resource crafting $\leftarrow$ Proactivity	Leg 1 of mediation	0.362	0.361	0.125	0.589
Avoidance resource crafting $\leftarrow$ Job autonomy	Leg 1 of mediation	0.315	0.297	0.055	0.537
Avoidance resource crafting $\leftarrow$ Self-efficacy	Leg 1 of mediation	-0.235	-0.257	-0.528	0.005
Work engagement $\leftarrow$ Proactivity * Approach role crafting	Indirect effect	0.312	0.106	0.113	0.529
Work meaning $\leftarrow$ Proactivity * Approach role crafting	Indirect effect	0.302	0.097	0.128	0.503
Work engagement $\leftarrow$ (Proactivity * Approach role crafting)	Total effect	0.317	0.308	0.115	0.518
Work meaning $\leftarrow$ (Proactivity * Approach role crafting)	Total effect	0.229	0.222	0.063	0.390

6 Discussion

This study examines how developers' use of generative AI relates to three individual work outcomes job satisfaction, work engagement, and work meaning and whether job crafting mediates these relationships. We have derived a research model from literature that outlines the impact of generative AI use on individual work outcomes among developers mediated by job crafting behavior in an organizational context.

The findings provide partial support for the hypothesis that generative AI use is positively related to the developers' individual-level work outcomes. Generative AI use shows a significant positive relationship with work engagement, whereas its relationships with job satisfaction and work meaning are not significant. In the study sample, where developers had an average of around six months of experience with GitHub Copilot, this study thus finds that higher generative AI usage is associated with feeling more absorbed, vigorous, and dedicated at work, but not with feeling more satisfied or perceiving work as more meaningful.

The hypotheses that approach job crafting positively mediates and avoidance job crafting negatively mediates the relationship between generative AI use and the three individual-level outcomes are not supported. The analysis of job complexity in generative AI-augmented work, however, shows an effect. The interaction between generative AI use and high job complexity shows a significant and positive relationship with avoidance resource crafting, whereas no such effect is found for low job complexity. This indicates that when developers use generative AI for high-complexity work, generative AI use is positively related to avoidance resource crafting.

Finally, this study finds that proactive personality and job autonomy are predictors of job crafting. Proactive personality is positively related to both approach role crafting and approach resource crafting, whereas job autonomy is positively related to avoidance resource crafting. Furthermore, we found that approach and avoidance job crafting are differently related to individual-level outcomes. Approach role crafting shows a significant positive relationship to work engagement and work meaning, while avoidance resource crafting shows a significant negative relationship to work meaning.

In summary, this study addresses the research question by showing that generative AI use is positively related to work engagement but not to job satisfaction or work meaning. Developers' job crafting does not mediate the relationship between generative AI use and individual-level work outcomes in the study sample. Instead, the findings indicate that when generative AI is used for high-complexity work, its use is positively related to avoidance resource crafting.

In the following sections, we elaborate on the theoretical and practical contributions of our study.

6.1 Theoretical contributions

This study addresses the question and contributes to research on AI-augmented ISD in several ways.

First, it extends the emerging IS and software engineering literature on generative AI in software development beyond its predominant focus on technical productivity factors. Prior studies show that generative AI can increase output and reduce completion time on programming tasks (Cui et al., 2024; Peng et al., 2023; Ziegler et al., 2022, 2024), but provide little insight into how generative AI use impacts developers' individual-level work outcomes. By theorizing and empirically examining work engagement, job satisfaction, and work meaning as outcomes of generative AI use in an organizational ISD setting, our study shows that AI augmentation in ISD is positively associated with developers' work engagement, but not with job satisfaction or work meaning. This refines the current understanding of productivity gains in AI-augmented ISD by demonstrating that generative AI use is positively related not only to technical performance metrics but also to work engagement as an individual-level work outcome that is an important antecedent of performance and developers' productivity in ISD (Christian et al., 2011; Murphy-Hill et al., 2019; Storey et al., 2019). Our study thus contributes to the literature on AI-augmented ISD.

It broadens our understanding of its impact on developers' productivity from a narrow focus on technical metrics to a more differentiated perspective that incorporates non-technical factors. It provides initial evidence that generative AI use in ISD is positively associated with work engagement as a key individual-level work outcome.

Second, our study contributes to research on generative AI and the reshaping of software developers' jobs. Existing work argues that generative AI will substantially transform developers' roles and tasks (Sætra, 2023; Sauvola et al., 2024; Ulfsnes et al., 2024). Furthermore, emerging studies suggest that generative AI can influence workers' job crafting behavior (Freise et al., 2025; Mayer et al., 2025a; Ritz et al., 2025). Contrary to these expectations, our study finds no evidence that developers' job crafting mediates the relationship between generative AI use and individual-level work outcomes. Instead, the results show that proactive personality is positively related to approach role and approach resource crafting, and that job autonomy is positively related to avoidance resource crafting. This finding is in line with job crafting research, which identifies individual and job-level factors as central antecedents of job crafting behavior (Bruning & Campion, 2018; Rudolph et al., 2017; Tims & Bakker, 2010). The results further demonstrate that approach role crafting is positively related to work engagement and work meaning, whereas avoidance resource crafting has a negative relationship with work meaning. This also confirms findings from the job crafting literature (Lichtenthaler & Fischbach, 2019; Petrou et al., 2012; Zhang & Parker, 2019).

Together, these findings indicate that, in the studied setting, developers' job crafting is primarily shaped by individual and job-level characteristics rather than by the extent of generative AI use. In the large telecommunications organization examined, software developers work in an incumbent organizational environment, and, at the time of the study, had an average of about six months of experience using GitHub Copilot. In this context, job crafting is not identified as a mediator in the relationship between generative AI use and individual-level outcomes, nor is generative AI use significantly related to any of the four job crafting dimensions. Our study thus advances prior work on generative AI-enabled job crafting by showing that in certain real-world organizational contexts, individual and job-level characteristics remain the dominant antecedents of how developers shape their jobs, and generative AI use alone is an insufficient condition for developers' job crafting.

Third, this study contributes to the job crafting research and the perspective on job complexity in the context of generative AI by introducing the concept of job complexity of generative AI-augmented work and by examining its role in the relationship between generative AI use and job crafting. Drawing on the job crafting research that conceptualizes job complexity as a central job characteristic that can be associated with both approach and avoidance forms of job crafting (Bai et al., 2021; Tian et al., 2022), our study shifts the focus from overall job complexity to the complexity of those parts of the job for which developers use generative AI. The results show no general relationship between generative AI use and any of the four job crafting

dimensions. However, when the generative AI-augmented work is highly complex, generative AI use is positively associated with avoidance resource crafting, whereas no such association is found for low-complexity AI-augmented work. Paper VI thus provides initial evidence that job complexity of generative AI-augmented work is a useful concept for job crafting research in the context of generative AI. It suggests that the impact of generative AI use on job crafting may be contingent on the complexity of the parts of the job that are AI-augmented.

6.2 Practical contribution

This study has practical implications for organizations that introduce generative AI into their ISD processes to support software developers in their work. The findings suggest that generative AI-augmented ISD should be evaluated not only through technical productivity indicators but also through developers' individual-level work outcomes. In the studied setting, a greater extent of generative AI use is associated with greater work engagement of developers (feeling more absorbed, vigorous, and dedicated in their work), but not with higher job satisfaction or perceived work meaning. This indicates that generative AI can support certain facets of developers' work experience while leaving others unchanged. To obtain a more comprehensive picture of the impact of generative AI-augmented ISD, organizations should therefore complement technical productivity metrics with regular monitoring of key individual-level work outcomes. This helps avoid situations in which technical indicators improve while individual-level outcomes stagnate or deteriorate, potentially undermining productivity gains.

This study further indicates that, in the incumbent organization studied and at the current stage of generative AI adoption (on average six months of GitHub Copilot use), developers' job crafting is driven primarily by individual and job-level characteristics rather than by the extent of generative AI use. This result stands in contrast to existing work that argues that generative AI is fundamentally reshaping developers' jobs and job crafting. For practice, this suggests that introducing generative AI into ISD processes alone is not sufficient to achieve transformative changes in how developers shape their work in real-world organizational settings. Organizations that expect such changes should not assume that the use of the technology itself will give rise to developers crafting their jobs. Instead, they should create opportunities for developers to reflect on how generative AI fits into their tasks and workflows, and to discuss under which conditions generative AI can add greater value to their work.

6.3 Limitations and future directions

While our study provides theoretical and practical insights on the use of generative AI, it is also important to recognize limitations.

Our study investigates the early adoption of generative AI by developers, reflecting that the use

of generative AI in development work is a recent phenomenon and can currently be examined primarily in its early adoption. As a result, the findings of this dissertation are most informative for the development and early adoption phases, while long-term dynamics, such as how development practices, delegation conditions, or generative AI-related job crafting and individual work outcomes stabilize or change over extended periods of use, remain outside the temporal scope of this dissertation. Our study provides early indications of how generative AI impacts job crafting and novel insights were uncovered that can inform future research. Future research should expand this research to an investigation over a longer period of time to further substantiate our initial findings.

6.4 Conclusion

Moving beyond a narrow productivity focus, our study provides initial evidence that a greater extent of generative AI use in AI-augmented ISD is positively associated with developers' work engagement. Additionally, in contrast to claims that generative AI reshapes developers' jobs, the findings in this study do not support job crafting as a mediator between generative AI use and individual-level work outcomes. Instead, developers' job crafting is primarily explained by established individual and job-level antecedents. This suggests that in certain real-world organizational settings, introducing generative AI into ISD processes alone may not achieve substantial changes in how developers proactively shape their jobs. Organizations seeking such changes may therefore need to complement tool rollout with opportunities for developers to reflect, experiment, and exchange with peers, so they can identify where generative AI adds value to their tasks and under which conditions it can be integrated into their everyday work practices to support desirable work outcomes.

References

- Ahmed, T., & Devanbu, P. (2022). Few-shot training llms for project-specific code summarization. *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, 1–5.
- Allamanis, M., Barr, E. T., Devanbu, P., & Sutton, C. (2018). A survey of machine learning for big code and naturalness. *ACM Computing Surveys (CSUR)*, 51(4), 1–37.
- Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmermann, T. (2019a). Software engineering for machine learning: A case study. *IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 291–300.
- Bai, J. Y., Tian, Q., & Liu, X. (2021). Examining job complexity on job crafting within conservation of resources theory: A dual-path mediation model. *Frontiers in Psychology*, 12, 1–12.
- Baird, A., & Maruping, L. M. (2021). The Next Generation of Research on IS Use: A Theoretical Framework of Delegation to and from Agentic IS Artifacts. *MIS Quarterly*, 45(1), 315–341.
- Bakker, A. B., Tims, M., & Derks, D. (2012). Proactive personality and job performance: The role of job crafting and work engagement. *Human Relations*, 65(10), 1359–1378.
- Banh, L., Holldack, F., & Strobel, G. (2025). Copiloting the future: How generative AI transforms Software Engineering. *Information and Software Technology*, 183, 1–14.
- Banker, R. D., & Kauffman, R. J. (1991). Reuse and productivity in integrated computer-aided software engineering: An empirical study. *MIS Quarterly*, 15(3), 375–401.
- Barke, S., James, M. B., & Polikarpova, N. (2023). Grounded copilot: How programmers interact with code-generating models. *Proceedings of the ACM on Programming Languages*, 7, 85–111.
- Benlian, A. (2020). A daily field investigation of technology-driven spillovers from work to home. *MIS Quarterly*, 44(3), 1259–1300.
- Berg, J. M., Dutton, J. E., & Wrzesniewski, A. (2013). Job crafting and meaningful work. American Psychological Association.
- Bird, C., Ford, D., Zimmermann, T., Forsgren, N., Kalliamvakou, E., Lowdermilk, T., & Gazit, I. (2023). Taking flight with Copilot. *Communications of the ACM*, 66(6), 56–62.
- Bowling, N. A., & Hammond, G. D. (2008). A meta-analytic examination of the construct validity of the Michigan Organizational Assessment Questionnaire Job Satisfaction Subscale. *Journal of Vocational Behavior*, 73(1), 63–77.
- Bruning, P. F., & Campion, M. A. (2018). A role–resource approach–avoidance model of job crafting: A multimethod integration and extension of job crafting theory. *Academy of Management Journal*, 61(2), 499–522.
- Brynjolfsson, E., Li, D., & Raymond, L. R. (2023). Generative AI at work.

- Chatterjee, S., & Simonoff, J. S. (2013). *Handbook of Regression Analysis*. John Wiley & Sons.
- Chin, W. W. (1998). Commentary: Issues and Opinion on Structural Equation Modeling. *MIS Quarterly*, 22(1), vii–xvi.
- Chreim, S., Williams, B. E. (, & Hinings, C. R. ((2007). Interlevel influences on the reconstruction of professional role identity. *Academy of Management Journal*, 50(6), 1515–1539.
- Christian, M. S., Garza, A. S., & Slaughter, J. E. (2011). Work engagement: A quantitative review and test of its relations with task and contextual performance. *Personnel Psychology*, 64(1), 89–136.
- Chuang, Y.-T., Chiang, H.-L., & Lin, A.-P. (2025). Insights from the Job Demands–Resources Model: AI’s dual impact on employees’ work and life well-being. *International Journal of Information Management*, 83, 1–17.
- Corbeanu, A., & Iliescu, D. (2023). The link between work engagement and job performance: A meta-analysis. *Journal of Personnel Psychology*, 22(3).
- Cui, Z. K., Demirer, M., Jaffe, S., Musolff, L., Peng, S., & Salz, T. (2024). The effects of generative AI on high skilled work: Evidence from three field experiments with software developers. *Available at SSRN 4945566*.
- Dakhel, A. M., Majdinasab, V., Nikanjam, A., Khomh, F., Desmarais, M. C., & Jiang, Z. M. J. (2023). Github Copilot AI Pair Programmer: Asset or liability? *Journal of Systems and Software*, 203, 1–27.
- Demerouti, E. (2014). Design your own job through job crafting. *European Psychologist*.
- Demerouti, E. (2022). Turn digitalization and automation to a job resource. *Applied Psychology*, 71(4), 1205–1209.
- Dierdorff, E. C., & Jensen, J. M. (2018). Crafting in context: Exploring when job crafting is dysfunctional for performance effectiveness. *Journal of Applied Psychology*, 103(5), 463–477.
- Dutton, J. E., Roberts, L. M., & Bednar, J. (2010). Pathways for positive identity construction at work: Four types of positive identity and the building of social resources. *Academy of Management Review*, 35(2), 265–293.
- Faraj, S., von Krogh, G., Monteiro, E., & Lakhani, K. R. (2016). Special Section Introduction Online Community as Space for Knowledge Flows. *Information Systems Research*, 27(4), 668–684.
- Faraj, S., Pachidi, S., & Sayegh, K. (2018). Working and organizing in the age of the learning algorithm. *Information and Organization*, 28(1), 62–70.
- Fornell, C., & Larcker, D. F. (1981). Evaluating structural equation models with unobservable variables and measurement error. *Journal of Marketing Research*, 18(1), 39–50.
- Freise, L. R., Bruhin, O., Ritz, E., Li, M. M., & Leimeister, J. M. (2025). Code and Craft: How Generative AI Tools Facilitate Job Crafting in Software Development. *Hawaii International Conference on System Sciences (HICSS) Proceedings*.

- Gerlich, M. (2025). AI tools in society: Impacts on cognitive offloading and the future of critical thinking. *Societies*, 15(6), 1–28.
- Gould, S. (1979). Age, job complexity, satisfaction, and performance. *Journal of Vocational Behavior*, 14(2), 209–223.
- Hackman, J. R., & Oldham, G. R. (1976). Motivation through the design of work: Test of a theory. *Organizational Behavior and Human Performance*, 16(2), 250–279.
- Hair, J. F. (2014). *A primer on partial least squares structural equation modeling (PLS-SEM)*. Sage.
- Hair, J. F., Matthews, L. M., Matthews, R. L., & Sarstedt, M. (2017). PLS-SEM or CB-SEM: updated guidelines on which method to use. *International Journal of Multivariate Data Analysis*, 1(2), 107–123.
- Henseler, J., Ringle, C. M., & Sarstedt, M. (2015). A new criterion for assessing discriminant validity in variance-based structural equation modeling. *Journal of the Academy of Marketing Science*, 43(1), 115–135.
- Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., & Qin, B. (2025). A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2), 1–55.
- Imai, S. (2022). Is github copilot a substitute for human pair-programming? an empirical study. *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*, 319–321.
- Jöreskog, K. G. (1971). Simultaneous factor analysis in several populations. *Psychometrika*, 36(4), 409–426.
- Judge, T. A., Thoresen, C. J., Bono, J. E., & Patton, G. K. (2001). The job satisfaction–job performance relationship: A qualitative and quantitative review. *Psychological Bulletin*, 127(3), 376.
- Law, M., & Varanasi, R. A. (2025). Generative AI and Changing Work: Systematic Review of Practitioner-Led Work Transformations Through the Lens of Job Crafting. *International Conference on Human-Computer Interaction*, 131–152.
- Lawler, E., Cammann, C., Nadler, D., & Jenkins, D. (1979). Michigan organizational assessment questionnaire. *Journal of Vocational Behavior*.
- Lee, S.-H., Shin, Y., & Baek, S. I. (2017). The impact of job demands and resources on job crafting. *Journal of Applied Business Research*, 33(4), 827–840.
- Li, J., Yang, H., Weng, Q., & Zhu, L. (2023). How different forms of job crafting relate to job satisfaction: The role of person-job fit and age. *Current Psychology*, 42(13), 11155–11169.
- Lichtenthaler, P. W., & Fischbach, A. (2019). A meta-analysis on promotion-and prevention-focused job crafting. *European Journal of Work and Organizational Psychology*, 28(1), 30–50.

- Lifshitz-Assaf, H. (2018). Dismantling knowledge boundaries at NASA: The critical role of professional identity in open innovation. *Administrative Science Quarterly*, 63(4), 746–782.
- Lindebaum, D., & Fleming, P. (2024). ChatGPT undermines human reflexivity, scientific responsibility and responsible management research. *British Journal of Management*, 35(2), 566–575.
- Lopper, E., Horstmann, K. T., & Hoppe, A. (2024). The Approach–Avoidance Job Crafting Scale: Development and validation of a measurement of the hierarchical structure of job crafting. *Applied Psychology*, 73(1), 93–134.
- Mayer, A.-S., Baygi, R. M., & Buwalda, R. (2025a). Generation AI: Job Crafting by Entry-Level Professionals in the Age of Generative AI: A.-S. Mayer et al. *Business & Information Systems Engineering*, 1–19.
- McAllister, D. J. (1995). Affect-and cognition-based trust as foundations for interpersonal cooperation in organizations. *Academy of Management Journal*, 38(1), 24–59.
- Microsoft. (2025). *Earnings Release FY25 Q4*. Microsoft. REDMOND, Wash.
- Murphy-Hill, E., Jaspan, C., Sadowski, C., Shepherd, D., Phillips, M., Winter, C., Knight, A., Smith, E., & Jorde, M. (2019). What predicts software developers' productivity? *IEEE Transactions on Software Engineering*, 47(3), 582–594.
- Ng, T. W. H., & Feldman, D. C. (2010). Organizational Tenure and Job Performance. *Journal of Management*, 36(5), 1220–1250.
- Nguyen-Duc, A., Cabrero-Daniel, B., Przybylek, A., Arora, C., Khanna, D., Herda, T., Rafiq, U., Melegati, J., Guerra, E., & Kemell, K.-K. (2023). Generative Artificial Intelligence for Software Engineering—A Research Agenda. *arXiv preprint arXiv:2310.18648*.
- Ozkaya, I. (2020). What is really different in engineering AI-enabled systems? *IEEE Software*, 37(4), 3–6.
- Ozkaya, I. (2023). Application of large language models to software engineering tasks: Opportunities, risks, and implications. *IEEE Software*, 40(3), 4–8.
- Parker, S. K., & Grote, G. (2022). Automation, algorithms, and beyond: Why work design matters more than ever in a digital world. *Applied Psychology*, 71(4), 1171–1204.
- Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of AI on developer productivity: Evidence from GitHub Copilot. *arXiv preprint arXiv:2302.06590*.
- Petrou, P., Demerouti, E., Peeters, M. C. W., Schaufeli, W. B., & Hetland, J. (2012). Crafting a job on a daily basis: Contextual correlates and the link to work engagement. *Journal of Organizational Behavior*, 33(8), 1120–1141.
- Pinsonneault, A., & Kraemer, K. (1993). Survey research methodology in management information systems: an assessment. *Journal of Management Information Systems*, 10(2), 75–105.

- Reay, T., Goodrick, E., Waldorff, S. B., & Casebeer, A. (2017). Getting leopards to change their spots: Co-creating a new professional role identity. *Academy of Management Journal*, 60(3), 1043–1070.
- Riemenschneider, C. K., & Armstrong, D. J. (2021). The development of the perceived distinctiveness antecedent of information systems professional identity. *MIS Quarterly*, 45(3), 1149–1186.
- Ritz, E., Freise, L. R., & Li, M. M. (2025). Offloading to Digital Minds: How Generative AI Can Help to Craft Jobs. *Information Systems and Neuroscience: NeuroIS Retreat 2024, Vienna, Austria*, 66, 28–37.
- Rosso, B. D., Dekas, K. H., & Wrzesniewski, A. (2010). On the meaning of work: A theoretical integration and review. *Research in Organizational Behavior*, 30, 91–127.
- Rudolph, C. W., Katz, I. M., Lavigne, K. N., & Zacher, H. (2017). Job crafting: A meta-analysis of relationships with individual differences, job characteristics, and work outcomes. *Journal of Vocational Behavior*, 102, 112–138.
- Sætra, H. S. (2023). Generative AI: Here to stay, but for good? *Technology in Society*, 75, 1–5.
- Santos, I., Magalhaes, C., & Santos, R. d. S. (2025). Model-Assisted and Human-Guided: Perceptions and Practices of Software Professionals Using LLMs for Coding. *arXiv preprint arXiv:2510.09058*.
- Sauvola, J., Tarkoma, S., Klemettinen, M., Riekk, J., & Doermann, D. (2024). Future of software development with generative AI. *Automated Software Engineering*, 31(1), 1–26.
- Schaufeli, W. B., Bakker, A. B., & Salanova, M. (2006). The measurement of work engagement with a short questionnaire: A cross-national study. *Educational and Psychological Measurement*, 66(4), 701–716.
- Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in machine learning systems. *Advances in Neural Information Processing Systems*, 28.
- Shah, A., Chernova, A., Tomson, E., Porter, L., Griswold, W. G., & Soosai Raj, A. G. (2025). Students' Use of GitHub Copilot for Working with Large Code Bases. *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*, 1050–1056.
- Spreitzer, G. M. (1995). Psychological empowerment in the workplace: Dimensions, measurement, and validation. *Academy of Management Journal*, 38(5), 1442–1465.
- Storey, M.-A., Zimmermann, T., Bird, C., Czerwonka, J., Murphy, B., & Kalliamvakou, E. (2019). Towards a theory of software developer job satisfaction and perceived productivity. *IEEE Transactions on Software Engineering*, 47(10), 2125–2142.
- Stray, V., Barbala, A., & Wivestad, V. T. (2025). Human-AI Collaboration in Software Development: A Mixed-Methods Study of Developers' Use of GitHub Copilot and ChatGPT. *Proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering*, 1325–1332.

- Strich, F., Mayer, A.-S., & Fiedler, M. (2021). What do I do in a world of artificial intelligence? Investigating the impact of substitutive decision-making AI systems on employees' professional role identity. *Journal of the Association for Information Systems*, 22(2), 9.
- Tarafdar, M., & Saunders, C. (2022). Remote, Mobile, and Blue-Collar: ICT-Enabled Job Crafting to Elevate Occupational Well-Being. *Journal of the Association for Information Systems*, 23(3), 707–749.
- Teye-Kwadjo, E., & de Bruin, G. P. (2022). Rasch analysis of the proactive personality scale. *Psychological Reports*, 125(5), 2788–2806.
- Thenoz, E., Raymond, M., Monnier-Senicourt, L., Jacob, F., Hemon-Hildgen, A., & Rowe, F. (2024). Integrating Generative AI into Work: A Case Study of Changes to Core Job Characteristics. *International Conference on Information Systems (ICIS) Proceedings*.
- Thomas, D., & Hunt, A. (2019). *The Pragmatic Programmer: your journey to mastery*. Addison-Wesley Professional.
- Tian, Q., Bai, J., & Wu, T. (2022). Should we be "challenging" employees? A study of job complexity and job crafting. *International Journal of Hospitality Management*, 102, 1–15.
- Tims, M., & Bakker, A. B. (2010). Job crafting: Towards a new model of individual job redesign. *SA Journal of Industrial Psychology*, 36(2), 1–9.
- Tims, M., Bakker, A. B., & Derks, D. (2012). Development and validation of the job crafting scale. *Journal of Vocational Behavior*, 80(1), 173–186.
- Ulfesnes, R., Mikalsen, M., Sporse, T. T., & Hatling, M. (2023). TECHNOLOGY FOR KNOWLEDGE WORK: A RELATIONAL PERSPECTIVE. *European Conference on Information Systems (ECIS) Proceedings*.
- Ulfesnes, R., Moe, N. B., Stray, V., & Skarpen, M. (2024). Transforming software development with generative AI: Empirical insights on collaboration and workflow. In *Generative ai for effective software development* (pp. 219–234). Springer.
- Van Dis, E. A. M., Bollen, J., Zuidema, W., Van Rooij, R., & Bockting, C. L. (2023). ChatGPT: five priorities for research. *Nature*, 614(7947), 224–226.
- Venkatesh, V., Brown, S. A., Maruping, L. M., & Bala, H. (2008). Predicting different conceptualizations of system use: The competing roles of behavioral intention, facilitating conditions, and behavioral expectation. *MIS Quarterly*, 32(3), 483–502.
- Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User Acceptance of Information Technology: Toward a Unified View. *MIS Quarterly*, 27(3), 425–478. <http://www.jstor.org/stable/30036540>
- Venkatesh, V., Thong, J. Y. L., Chan, F. K. Y., Hoehle, H., & Spohrer, K. (2020). How agile software development methods reduce work exhaustion: Insights on role perceptions and organizational skills. *Information Systems Journal*, 30(4), 733–761.

- Von Krogh, G., Spaeth, S., & Haefliger, S. (2005). Knowledge reuse in open source software: An exploratory study of 15 open source projects. *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 198b–198b.
- Wagner, S., & Ruhe, M. (2018). A systematic review of productivity factors in software development. *arXiv preprint arXiv:1801.06475*.
- Weisz, J. D., Muller, M., Ross, S. I., Martinez, F., Houde, S., Agarwal, M., Talamadupula, K., & Richards, J. T. (2022). Better together? an evaluation of AI-supported code translation. *27th International Conference on Intelligent User Interfaces*, 369–391.
- Welsh, M. (2022). The end of programming. *Communications of the ACM*, 66(1), 34–35.
- Wrzesniewski, A., & Dutton, J. E. (2001). Crafting a job: Revisioning employees as active crafters of their work. *Academy of Management Review*, 26(2), 179–201.
- Wu, S., Liu, Y., Ruan, M., Chen, S., & Xie, X.-Y. (2025). Human-generative AI collaboration enhances task performance but undermines human’s intrinsic motivation. *Scientific Reports*, 15(1–30), 15105.
- Yuan, Z., Lou, Y., Liu, M., Ding, S., Wang, K., Chen, Y., & Peng, X. (2023). No more manual tests? evaluating and improving chatgpt for unit test generation. *arXiv preprint arXiv:2305.04207*.
- Zhang, F., & Parker, S. K. (2019). Reorienting job crafting research: A hierarchical structure of job crafting concepts and integrative review. *Journal of Organizational Behavior*, 40(2), 126–146.
- Zhang, J., Chen, Y., Liu, C., Niu, N., & Wang, Y. (2023). Empirical Evaluation of ChatGPT on Requirements Information Retrieval Under Zero-Shot Setting. *2023 International Conference on Intelligent Computing and Next Generation Networks (ICNGN)*, 1–6.
- Ziegler, A., Kalliamvakou, E., Li, X. A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., & Aftandilian, E. (2022). Productivity assessment of neural code completion. *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, 21–29.
- Ziegler, A., Kalliamvakou, E., Li, X. A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., & Aftandilian, E. (2024). Measuring github copilot’s impact on productivity. *Communications of the ACM*, 67(3), 54–63.
- Zmud, R. W. (1980). Management of large software development efforts. *MIS Quarterly*, 4(2), 45–55.

Appendix

Table 6: Measurement items

Construct	Dimensions	Measurement items	Loading (t-value)	Scale	Properties
Generative AI use (Venkatesh et al. 2008; Agarwal & Karahanna, 2000)	Frequency	- How often do you use Github Copilot? (use1)	0.877*** (9.845)	1 = Don't use at all, 7 = Use several times each day	Alpha: 0.815, CR: 0.889, AVE: 0.729
	Intensity	- How do you consider the extent of your current use of Github Copilot (use2)	0.878*** (7.861)	1 = Non use, 7 = Heavy use	
	Duration of use (free text)	- On average, how many hours do you use GitHub Copilot each week? (useh)	0.805*** (6.526)	Free text in hours	
Job complexity (Gould, 1979)	Generative AI-handled job complexity low	- GitHub Copilot performs routine work consisting of simple repetitive operations (djc1)	0.872*** (5.975)	(1 = Strongly Disagree, 7 = Strongly Agree)	Alpha: 0.642 CR: 0.847 AVE: 0.735
		- GitHub Copilot performs work from detailed instructions or where variations in procedures are limited (djc2)	0.844*** (5.072)		
	Generative AI-handled job complexity high	- GitHub Copilot plans and performs complex work where only general policies or procedures are available (djc4)	0.864*** (6.251)	(1 = Strongly Disagree, 7 = Strongly Agree)	Alpha: 0.786, CR: 0.899, AVE: 0.817
		- GitHub Copilot plans and performs highly complex or technical work where no procedures or standards are available (djc5)	0.943*** (8.070)		
Copilot Usage Behavior (Ziegler et al., 2024)	Acceptance rate (0-100%)	- How often do you accept suggestions of GitHub Copilot unchanged (in percent)? (usea)	1.000	Slider (0-100%), 5% steps	
Job crafting – Role Crafting (Bruning & Campion, 2018)	Approach Role Crafting	Work role expansion		(1 = Strongly Disagree, 7 = Strongly Agree)	Alpha: 0.895, CR: 0.923, AVE: 0.705
		- I expand my role by providing opinions on important issues (wre1)	0.847*** (24.817)		
		- I expand my work activities to make sure I take care of myself (wre2)	0.742*** (12.301)		
		- I expand my work activities to acquire resources that will help me do my job (wre3)	0.872*** (21.549)		

Continued on next page

Table 6: Measurement items (continued)

Construct	Dimensions	Measurement items	Loading (t-value)	Scale	Properties
		- I expand my work by adding activities to my job that ensure the quality of my deliverables (wre4)	0.878*** (29.389)		
		- I expand my work by adding activities to my job that enhance safety or security (wre5)	0.852*** (28.612)		
		<i>Social expansion</i>			Alpha: 0.913, CR: 0.939, AVE: 0.794
		- I actively initiate positive interactions with others at work (soe1)	0.906*** (40.655)		
		- I actively work to improve my communication quality with others at work (soe2)	0.912*** (52.569)		
		- I actively develop my professional network at my job (soe3)	0.848*** (25.961)		
		- I actively work to improve the quality of group interactions (soe4)	0.898*** (38.051)		
	Avoidance	<i>Work role reduction</i>		(1 =	Alpha:
	Role	- I find ways to get others to take my place in meetings. (wrr1)	0.853*** (12.279)	Strongly	0.746, CR:
	Crafting	- I find ways to outsource my work to others outside my group. (wrr2)	0.826*** (11.093)	Disagree, 7 = Strongly	0.836, AVE: 0.569
		- I find ways to reduce the time I spend in meetings. (wrr3)	0.593*** (4.201)	Agree)	
		- I find ways to bypass time-consuming tasks. (wrr4)	0.733*** (4.814)		
Job crafting – Resource Crafting (Bruning & Campion, 2018)	Approach Resource Crafting	<i>Work organization</i>		(1 =	Alpha:
		- I create structure in my work processes. (wog1)	0.912*** (39.636)	Strongly	0.882, CR:
		- I create organization in my work environment. (wog2)	0.831*** (15.979)	Disagree, 7 = Strongly	0.919, AVE: 0.740
		- I create structure in my work schedule. (wog3)	0.873*** (26.808)	Agree)	
		- I create plans and prioritize my work in an organized manner. (wog4)	0.822*** (21.001)		
		<i>Adoption / Technology Crafting</i>			Alpha:
		- On my own, I seek training on new technology. (ado1)	0.804*** (21.400)		0.827, CR: 0.884, AVE: 0.657

Continued on next page

Table 6: Measurement items (continued)

Construct	Dimensions	Measurement items	Loading (t-value)	Scale	Properties
		- On my own, I seek training to improve my work. (ado2)	0.832*** (16.178)		
		- I use new knowledge or technology to automate tasks. (ado3)	0.814*** (15.906)		
		- I use new knowledge or technology to structure my work. (ado4)	0.790*** (15.589)		
		Metacognition			Alpha: 0.906, CR: 0.930, AVE: 0.728
		- I use my thoughts to put myself into a good mood at work. (meta1)	0.879*** (31.994)		
		- I use my thoughts to get me out of a bad mood at work. (meta2)	0.765*** (11.360)		
		- I use my thoughts to help me focus and be engaged at work. (meta3)	0.899*** (46.408)		
		- I use my thoughts to create a personal mental approach to work. (meta4)	0.890*** (39.385)		
		- I use my thoughts to help me prepare for future work I will be doing. (meta5)	0.828*** (19.160)		
	Avoidance	Withdrawal		(1 =	Alpha:
	Resource	- I work in a way that allows me to avoid others at work. (with1)	0.908*** (40.224)	Strongly	0.813, CR:
	Crafting	- I work in a way that allows me to avoid interacting with people when working. (with2)	0.923*** (45.008)	Disagree, 7 = Strongly	0.891, AVE: 0.733
		- I work in a way that allows me to avoid bothersome tasks involved in my work. (with3)	0.725*** (10.534)	Agree)	
Professional Role Identity (Riemen-schneider et al., 2021)	Professional Role Identity	- I think and act like other software developers (pri1)	0.706*** (10.520)	(1 = Strongly	Alpha: 0.858, CR:
		- I share a lot in common with other software developers (pri2)	0.786*** (15.344)	Disagree, 7 = Strongly	0.895, AVE: 0.592
		- In general, I feel good as a software developer (pri4)	0.825*** (17.984)	Agree)	
		- I would like to work further as a software development professional. (pri6)	0.824*** (19.680)		

Continued on next page

Table 6: Measurement items (continued)

Construct	Dimensions	Measurement items	Loading (t-value)	Scale	Properties
		- I think what happens to software development professionals will affect me greatly. (pri7)	0.531*** (4.991)		
		- I'm eager to make my contribution to the software development profession. (pri8)	0.843*** (26.712)		
Monitoring Behavior (McAllister, 1995)	Monitoring	- I keep close track of my interactions with this system, taking note of instances it does not keep up its end of the bargain. (mon4)	0.794*** (6.014)	(1 = Strongly Disagree, 7 = Strongly Agree)	Alpha: 0.538, CR: 0.812, AVE: 0.684
		- I have found it necessary to make inquiries before responding to GitHub Copilot's request for assistance. This ensures that my interests are protected. (mon5)	0.823*** (6.121)		
Work enrichment (Spreitzer, 1995; Lawler et al., 1979)	Work meaning	- The work I do is meaningful. (mea1)	0.876*** (30.833)	(1 = Strongly Disagree, 7 = Strongly Agree)	Alpha: 0.878, CR: 0.925, AVE: 0.805
		- The work I do is very important to me. (mea2)	0.882*** (22.792)		
		- My job activities are personally meaningful to me. (mea3)	0.932*** (66.258)		
	Job satisfaction	- All in all, I am satisfied with my job. (mea4)	0.847*** (16.364)	(1 = Strongly Disagree, 7 = Strongly Agree)	Alpha: 0.720, CR: 0.838, AVE: 0.634
		- I would prefer another, more ideal job. (mea5)	0.755*** (10.173)		
		- I am satisfied with the important aspects of my job. (mea6)	0.795*** (11.036)		
Work engagement (Schaufeli et al., 2006)	Vigor	- At my work, I feel that I am bursting with energy (enga1)	0.865*** (18.158)	(1 = Strongly Disagree, 7 = Strongly Agree)	Alpha: 0.937, CR: 0.948, AVE: 0.695
		- At my work, I feel strong and vigorous (enga2)	0.871*** (29.131)		
		- When I get up in the morning, I feel like going to work (enga3)	0.864*** (15.742)		
	Dedication	- My job inspires me (enga4)	0.878*** (31.724)		
		- I am proud on the work that I do (enga5)	0.832*** (24.099)		
		- I am enthusiastic about my job (enga6)	0.790*** (14.640)		

Continued on next page

Table 6: Measurement items (continued)

Construct	Dimensions	Measurement items	Loading (t-value)	Scale	Properties
	Absorption	- I feel happy when I am working intensely (enga7)	0.825*** (19.867)		
		- I am immersed in my work (enga8)	0.871*** (33.469)		

Complete References

- Aaltonen, A., Alaimo, C., & Kallinikos, J. (2021). The making of data commodities: Data analytics as an embedded process. *Journal of Management Information Systems*, 38(2), 401–429.
- Abdel-Karim, B. M., Pfeuffer, N., Carl, K. V., & Hinz, O. (2023). How AI-Based Systems Can Induce Reflections: The Case of AI-Augmented Diagnostic Work. *MIS Quarterly*, 47(4), 1395–1424.
- Ågerfalk, P. J. (2020). Artificial intelligence as digital agency. *European Journal of Information Systems*, 29(1), 1–8.
- Ågerfalk, P. J., Conboy, K., Crowston, K., Lundström, J. E., Jarvenpaa, S. L., Ram, S., & Mikalef, P. (2022). Artificial Intelligence in Information Systems: State of the Art and Research Roadmap. *Communications of the AIS*, 50, 420–438.
- Ågerfalk, P. J., & Fitzgerald, B. (2006). Session details: Flexible and distributed software processes: Old petunias in new bowls? *Communications of the ACM*, 49(10), 27–34.
- Agrawal, L. A., Kanade, A., Goyal, N., Lahiri, S. K., & Rajamani, S. K. (2023). Guiding language models of code with global context using monitors. *arXiv preprint arXiv:2306.10763*.
- Ahmed, T., & Devanbu, P. (2022). Few-shot training llms for project-specific code summarization. *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, 1–5.
- Alaimo, C., & Kallinikos, J. (2021). Managing by data: Algorithmic categories and organizing. *Organization Studies*, 42(9), 1385–1407.
- Alaimo, C., & Kallinikos, J. (2022). Organizations decentered: Data objects, technology and knowledge. *Organization Science*, 33(1), 19–37.
- Alavi, M., & Leidner, D. E. (2001). Knowledge management and knowledge management systems: Conceptual foundations and research issues. *MIS Quarterly*, 25(1), 107–136.
- Alavi, M., Leidner, D. E., & Mousavi, R. (2024). A Knowledge Management Perspective of Generative Artificial Intelligence. *Journal of the Association for Information Systems*, 25(1), 1–12.
- Allamanis, M., Barr, E. T., Devanbu, P., & Sutton, C. (2018). A survey of machine learning for big code and naturalness. *ACM Computing Surveys (CSUR)*, 51(4), 1–37.
- Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., & Zimmermann, T. (2019a). Software engineering for machine learning: A case study. *IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 291–300.
- Amershi, S., Weld, D., Vorvoreanu, M., Fournay, A., Nushi, B., Collisson, P., Suh, J., Iqbal, S., Bennett, P. N., & Inkpen, K. (2019b). Guidelines for human-AI interaction. *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–13.

- Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*.
- Ang, S., & Slaughter, S. A. (2001). Work Outcomes and Job Design for Contract Versus Permanent Information Systems Professionals On Software Development Teams1, 2. *MIS Quarterly*, 25(3), 321–350.
- Arrieta, A. B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., Chatila, R., & Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, 82–115.
- Arumugam, D., Lee, J. K., Saskin, S., & Littman, M. L. (2019). Deep reinforcement learning from policy-dependent human feedback. *arXiv preprint arXiv:1902.04257*.
- Asatiani, A., Malo, P., Nagbol, P. R., Penttinen, E., Rinta-Kahila, T., & Salovaara, A. (2021). Sociotechnical Envelopment of Artificial Intelligence: An Approach to Organizational Deployment of Inscrutable Artificial Intelligence Systems. *Journal of the Association for Information Systems*, 22(2), 325–352.
- Auernhammer, J. (2020). Human-centered AI: The role of Human-centered Design Research in the development of AI. *Synergy - DRS International Conference 2020*, 1315–1333.
- Baer, I., Waardenburg, L., & Huysman, M. (2022). What Are We Augmenting? A Multidisciplinary Analysis of AI-based Augmentation for the Future of Work. *International Conference on Information Systems (ICIS) Proceedings*.
- Bai, J. Y., Tian, Q., & Liu, X. (2021). Examining job complexity on job crafting within conservation of resources theory: A dual-path mediation model. *Frontiers in Psychology*, 12, 1–12.
- Baig, A., Merrill, D., & Sinha, M. (2024). Moving past gen AI's honeymoon phase: Seven hard truths for CIOs to get from pilot to scale. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/moving-past-gen-ais-honeymoon-phase-seven-hard-truths-for-cios-to-get-from-pilot-to-scale>
- Baird, A., & Maruping, L. M. (2021). The Next Generation of Research on IS Use: A Theoretical Framework of Delegation to and from Agentic IS Artifacts. *MIS Quarterly*, 45(1), 315–341.
- Bajarin, T. (2024). Nvidia's CEO On The Democratization Of Coding. <https://www.forbes.com/sites/timbajarin/2024/03/20/nvidias-ceo-on-the-democratization-of-coding/?sh=4ce975567a95>
- Bakker, A. B., Tims, M., & Derks, D. (2012). Proactive personality and job performance: The role of job crafting and work engagement. *Human Relations*, 65(10), 1359–1378.
- Balijepally, V., Mahapatra, R., Nerur, S., & Price, K. H. (2009). Are two heads better than one for software development? The productivity paradox of pair programming. *MIS Quarterly*, 33(1), 91–118.

- Bamberger, P. (2008). From the editors beyond contextualization: Using context theories to narrow the micro-macro gap in management research. *Academy of Management Journal*, 51(5), 839–846.
- Bandura, A. (2006). Toward a psychology of human agency. *Perspectives on Psychological Science*, 1(2), 164–180.
- Banh, L., Holldack, F., & Strobel, G. (2025). Copiloting the future: How generative AI transforms Software Engineering. *Information and Software Technology*, 183, 1–14.
- Banker, R. D., Datar, S. M., Kemerer, C. F., & Zweig, D. (1993). Software complexity and maintenance costs. *Communications of the ACM*, 36(11), 81–94.
- Banker, R. D., & Kauffman, R. J. (1991). Reuse and productivity in integrated computer-aided software engineering: An empirical study. *MIS Quarterly*, 15(3), 375–401.
- Banker, R. D., & Slaughter, S. A. (2000). The Moderating Effects of Structure on Volatility and Complexity in Software Enhancement. *Information Systems Research*, 11(3), 219–240.
- Barke, S., James, M. B., & Polikarpova, N. (2023). Grounded copilot: How programmers interact with code-generating models. *Proceedings of the ACM on Programming Languages*, 7, 85–111.
- Barki, H., & Hartwick, J. (1989). Rethinking the concept of user involvement. *MIS Quarterly*, 13(1), 53–63.
- Barki, H., & Hartwick, J. (1994). Measuring user participation, user involvement, and user attitude. *MIS Quarterly*, 18(1), 59–82.
- Baskerville, R. L., & Stage, J. (1996). Controlling Prototype Development through Risk Analysis. *MIS Quarterly*, 20(4), 481–504.
- Baylor, D., Breck, E., Cheng, H.-T., Fiedel, N., Foo, C. Y., Haque, Z., Haykal, S., Ispir, M., Jain, V., & Koc, L. (2017). Tfx: A tensorflow-based production-scale machine learning platform. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1387–1395.
- Beaudry, A., & Pinsonneault, A. (2005). Understanding user responses to information technology: A coping model of user adaptation. *MIS Quarterly*, 29(3), 493–524.
- Bellefonds, N. d., Charanya, T., Franke, M. R., Apotheker, J., Forth, P., Grebe, M., Luther, A., Laubier, R. d., Lukic, V., Martin, M., Nopp, C., & Sassine, J. (2024). Where's the Value in AI? <https://www.bcg.com/publications/2024/wheres-value-in-ai>
- Benbasat, I., Goldstein, D. K., & Mead, M. (1987). The case research strategy in studies of information systems. *MIS Quarterly*, 11(3), 369–386.
- Benbasat, I., & Zmud, R. W. (2003). The identity crisis within the IS discipline: Defining and communicating the discipline's core properties. *MIS Quarterly*, 27(2), 183–194.
- Benbya, H., Davenport, T. H., & Pachidi, S. (2020). Artificial intelligence in organizations: current state and future opportunities. *MIS Quarterly Executive*, 19(4), 1–15.
- Benbya, H., Leidner, D. E., & Preston, D. (2019). MIS Quarterly research curation on information systems alignment.

- Benbya, H., & McKelvey, B. (2006). Using coevolutionary and complexity theories to improve IS alignment: a multi-level approach. *Journal of Information Technology*, 21, 284–298.
- Benbya, H., Pachidi, S., & Jarvenpaa, S. (2021). Special issue editorial: artificial intelligence in organizations: implications for information systems research. *Journal of the Association for Information Systems*, 22(2), 281–303.
- Benbya, H., Strich, F., & Tamm, T. (2024). Navigating Generative Artificial Intelligence Promises and Perils for Knowledge and Creative Work. *Journal of the Association for Information Systems*, 25(1), 23–36.
- Benlian, A. (2020). A daily field investigation of technology-driven spillovers from work to home. *MIS Quarterly*, 44(3), 1259–1300.
- Berente, N., Gu, B., Recker, J., & Santhanam, R. (2021). Managing Artificial Intelligence. *MIS Quarterly*, 45(3), 1433–1450.
- Berente, N., Lyytinen, K., Yoo, Y., & King, J. L. (2016). Routines as shock absorbers during organizational transformation: Integration, control, and NASA's enterprise information system. *Organization Science*, 27(3), 551–572.
- Berg, J. M., Dutton, J. E., & Wrzesniewski, A. (2013). Job crafting and meaningful work. American Psychological Association.
- Berg, J. M., Raj, M., & Seamans, R. (2023). Capturing value from artificial intelligence. *Academy of Management Discoveries*, 9(4), 424–428.
- Bessey, A., Block, K., Chelf, B., Chou, A., Fulton, B., Hallem, S., Henri-Gros, C., Kamsky, A., McPeak, S., & Engler, D. (2010). A few billion lines of code later: using static analysis to find bugs in the real world. *Communications of the ACM*, 53(2), 66–75.
- Bird, C., Ford, D., Zimmermann, T., Forsgren, N., Kalliamvakou, E., Lowdermilk, T., & Gazit, I. (2023). Taking flight with Copilot. *Communications of the ACM*, 66(6), 56–62.
- Bojinov, I. (2023). Keep Your AI Projects on Track. <https://hbr.org/2023/11/keep-your-ai-projects-on-track>
- Bommasani, R. (2021). On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.
- Bostrom, R. P., Gupta, S., & Thomas, D. (2009). A meta-theory for understanding information systems within sociotechnical systems. *Journal of Management Information Systems*, 26(1), 17–48.
- Bostrom, R. P., & Heinen, J. S. (1977a). MIS Problems and Failures: A Socio-Technical Perspective, Part II: The Application of Socio-Technical Theory. *MIS Quarterly*, 1(4), 11–28.
- Bostrom, R. P., & Heinen, J. S. (1977b). MIS problems and failures: A socio-technical perspective. Part I: The causes. *MIS Quarterly*, 1(3), 17–32.
- Boussioux, L., Lane, J. N., Zhang, M., Jacimovic, V., & Lakhani, K. R. (2024). The crowdless future? Generative AI and creative problem-solving. *Organization Science*, 35(5), 1589–1607.

- Bowling, N. A., & Hammond, G. D. (2008). A meta-analytic examination of the construct validity of the Michigan Organizational Assessment Questionnaire Job Satisfaction Subscale. *Journal of Vocational Behavior*, 73(1), 63–77.
- Breck, E., Cai, S., Nielsen, E., Salib, M., & Sculley, D. (2017). The ML test score: A rubric for ML production readiness and technical debt reduction. *2017 IEEE International Conference on Big Data (Big Data)*, 1123–1132.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., & Askell, A. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877–1901.
- Browne, G. J., & Rogich, M. B. (2001). An empirical investigation of user requirements elicitation: Comparing the effectiveness of prompting techniques. *Journal of Management Information Systems*, 17(4), 223–249.
- Bruning, P. F., & Campion, M. A. (2018). A role–resource approach–avoidance model of job crafting: A multimethod integration and extension of job crafting theory. *Academy of Management Journal*, 61(2), 499–522.
- Brynjolfsson, E., Li, D., & Raymond, L. R. (2023). Generative AI at work.
- Brynjolfsson, E., Mitchell, T., & Rock, D. (2018). What can machines learn, and what does it mean for occupations and the economy? *AEA Papers and Proceedings*, 108, 43–47.
- Bürkland, S., & Zachariassen, F. (2014). Developing an ERP technology: Handling incompleteness of the system. *Scandinavian Journal of Management*, 30(4), 409–426.
- Burrell, J. (2016). How the Machine ‘Thinks’: Understanding Opacity in Machine Learning Algorithms. *Big Data & Society*, 3(1), 1–12.
- Burtch, G., Lee, D., & Chen, Z. (2024). Generative AI Degrades Online Communities. *Communications of the ACM*, 67(3), 40–42.
- Burton-Jones, A., & Grange, C. (2013). From use to effective use: A representation theory perspective. *Information Systems Research*, 24(3), 632–658.
- Burton-Jones, A., Recker, J., Indulska, M., Green, P., & Weber, R. (2017). Assessing representation theory with a framework for pursuing success and failure. *MIS Quarterly*, 41(4), 1307–1334.
- Byrd, T. A., Cossick, K. L., & Zmud, R. W. (1992). A Synthesis of Research on Requirements Analysis and Knowledge Acquisition Techniques. *MIS Quarterly*, 16(1), 117–138.
- Campion, M. A., & McClelland, C. L. (1993). Follow-up and extension of the interdisciplinary costs and benefits of enlarged jobs. *Journal of Applied Psychology*, 78(3), 339–351.
- Carlile, P. R. (2004). Transferring, translating, and transforming: An integrative framework for managing knowledge across boundaries. *Organization Science*, 15(5), 555–568.
- Carroll, J. M., & Olson, J. R. (1988). Mental Models in Human-Computer Interaction. *Handbook of Human-Computer Interaction*, 45–65.
- Castelfranchi, C., & Falcone, R. (1998). Towards a theory of delegation for agent-based systems. *Robotics and Autonomous Systems*, 24(3-4), 141–157.

- Chakraborty, S., Sarker, S., & Sarker, S. (2010). An Exploration into the Process of Requirements Elicitation: A Grounded Approach. *Journal of the Association for Information Systems*, 11(4), 212–249.
- Chalmeta, R. (2006). Methodology for customer relationship management. *Journal of Systems and Software*, 79(7), 1015–1024.
- Chatterjee, S., & Simonoff, J. S. (2013). *Handbook of Regression Analysis*. John Wiley & Sons.
- Chen, C. C., Liu, J. Y.-C., & Chen, H.-G. (2011). Discriminative effect of user influence and user responsibility on information system development processes and project management. *Information and Software Technology*, 53(2), 149–158.
- Chen, W., & Hirschheim, R. (2004). A paradigmatic and methodological examination of information systems research from 1991 to 2001. *Information Systems Journal*, 14(3), 197–235.
- Chin, W. W. (1998). Commentary: Issues and Opinion on Structural Equation Modeling. *MIS Quarterly*, 22(1), vii–xvi.
- Chollet, F. (2019). On the measure of intelligence. *arXiv preprint arXiv:1911.01547*.
- Chollet, F. (2021). *Deep learning with Python*. Simon; Schuster.
- Chreim, S., Williams, B. E. (, & Hinings, C. R. ((2007). Interlevel influences on the reconstruction of professional role identity. *Academy of Management Journal*, 50(6), 1515–1539.
- Christian, M. S., Garza, A. S., & Slaughter, J. E. (2011). Work engagement: A quantitative review and test of its relations with task and contextual performance. *Personnel Psychology*, 64(1), 89–136.
- Chuang, Y.-T., Chiang, H.-L., & Lin, A.-P. (2025). Insights from the Job Demands–Resources Model: AI’s dual impact on employees’ work and life well-being. *International Journal of Information Management*, 83, 1–17.
- Chui, M., Hazan, E., Roberts, R., Singla, A., Smaje, K., Sukharevsky, A., Yee, L., & Zimmel, R. (2023a). The economic potential of generative AI: The next productivity frontier. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier>
- Chui, M., Yee, L., Hall, B., & Singla, A. (2023b). The state of AI in 2023: generative AI’s breakout year. <https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai-in-2023-generative-ais-breakout-year>
- Clegg, C. W. (2000). Sociotechnical principles for system design. *Applied Ergonomics*, 31(5), 463–477.
- Conboy, K. (2009). Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development. *Information Systems Research*, 20(3), 329–354.
- Corbeanu, A., & Iliescu, D. (2023). The link between work engagement and job performance: A meta-analysis. *Journal of Personnel Psychology*, 22(3).

- Cousot, P., Cousot, R., Feret, J., Mauborgne, L., Miné, A., Monniaux, D., & Rival, X. (2005). The ASTREE analyzer. *Proceedings of the 14th European Conference on Programming Languages and Systems*, 21–30.
- Cui, Z. K., Demirer, M., Jaffe, S., Musolff, L., Peng, S., & Salz, T. (2024). The effects of generative AI on high skilled work: Evidence from three field experiments with software developers. *Available at SSRN 4945566*.
- Cusumano, M. A. (2023). Generative AI as a New Innovation Platform. *Communications of the ACM*, 66(10), 18–21.
- Dakhel, A. M., Majdinasab, V., Nikanjam, A., Khomh, F., Desmarais, M. C., & Jiang, Z. M. J. (2023). Github Copilot AI Pair Programmer: Asset or liability? *Journal of Systems and Software*, 203, 1–27.
- Davenport, T. H. (2021). The analytics team. In *Inform's analytics body of knowledge* (pp. 49–76). Wiley.
- Dell'Acqua, F., McFowland, E., Mollick, E. R., Lifshitz-Assaf, H., Kellogg, K., Rajendran, S., Kraymer, L., Candelon, F., & Lakhani, K. R. (2023). Navigating the jagged technological frontier: Field experimental evidence of the effects of AI on knowledge worker productivity and quality.
- Dellermann, D., Ebel, P., Söllner, M., & Leimeister, J. M. (2019). Hybrid intelligence. *Business & Information Systems Engineering*, 61(5), 637–643.
- Demerouti, E. (2014). Design your own job through job crafting. *European Psychologist*.
- Demerouti, E. (2022). Turn digitalization and automation to a job resource. *Applied Psychology*, 71(4), 1205–1209.
- Deng, Y., Zheng, J., Huang, L., & Kannan, K. (2023). Let artificial intelligence be your shelf watchdog: The impact of intelligent image processing-powered shelf monitoring on product sales. *MIS Quarterly*, 47(3), 1045–1072.
- Denning, P. J. (2023). Can generative AI bots be trusted? *Communications of the ACM*, 66(6), 24–27.
- Dhar, V. (2013). Data science and prediction. *Communications of the ACM*, 56(12), 64–73.
- Dierdorff, E. C., & Jensen, J. M. (2018). Crafting in context: Exploring when job crafting is dysfunctional for performance effectiveness. *Journal of Applied Psychology*, 103(5), 463–477.
- Dubé, L., & Paré, G. (2003). Rigor in information systems positivist case research: current practices, trends, and recommendations. *MIS Quarterly*, 27(4), 597–636.
- Dutton, J. E., Roberts, L. M., & Bednar, J. (2010). Pathways for positive identity construction at work: Four types of positive identity and the building of social resources. *Academy of Management Review*, 35(2), 265–293.
- Dwivedi, Y. K., Kshetri, N., Hughes, L., Slade, E. L., Jeyaraj, A., Kar, A. K., Baabdullah, A. M., Koohang, A., Raghavan, V., & Ahuja, M. (2023). “So what if ChatGPT wrote it?” Multidisciplinary perspectives on opportunities, challenges and implications of generative

- conversational AI for research, practice and policy. *International Journal of Information Management*, 71, 1–63.
- Dybå, T., Sjøberg, D. I. K., & Cruzes, D. S. (2012). What works for whom, where, when, and why? On the role of context in empirical software engineering. *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 19–28.
- Edmondson, A. C., & McManus, S. E. (2007). Methodological fit in management field research. *Academy of Management Review*, 32(4), 1246–1264.
- Eisenhardt, K. M. (1989a). Agency theory: An assessment and review. *Academy of Management Review*, 14(1), 57–74.
- Eisenhardt, K. M. (1989b). Building theories from case study research. *Academy of Management Review*, 14(4), 532–550.
- Eisenhardt, K. M., & Graebner, M. E. (2007). Theory building from cases: Opportunities and challenges. *Academy of Management Journal*, 50(1), 25–32.
- Eloundou, T., Manning, S., Mishkin, P., & Rock, D. (2023). Gpts are gpts: An early look at the labor market impact potential of large language models. *arXiv preprint arXiv:2303.10130*.
- Emirbayer, M., & Mische, A. (1998). What is agency? *American Journal of Sociology*, 103(4), 962–1023.
- Engelbart, D. C. (1965). *Augmenting human intellect: Experiments, concepts, and possibilities*. Stanford Research Institute Menlo Park.
- Ernst, N. A., & Bavota, G. (2022). AI-driven development is here: Should you worry? *IEEE Software*, 39(2), 106–110.
- Fadel, K. J., & Brown, S. A. (2010). Information systems appraisal and coping: The role of user perceptions. *Communications of the AIS*, 26(1), 6.
- Fahlman, S. E., & Hinton, G. E. (1987). Connectionist architectures for artificial intelligence. *Computer*, 20(01), 100–109.
- Faraj, S., von Krogh, G., Monteiro, E., & Lakhani, K. R. (2016). Special Section Introduction Online Community as Space for Knowledge Flows. *Information Systems Research*, 27(4), 668–684.
- Faraj, S., Pachidi, S., & Sayegh, K. (2018). Working and organizing in the age of the learning algorithm. *Information and Organization*, 28(1), 62–70.
- Faraj, S., & Sproull, L. (2000). Coordinating expertise in software development teams. *Management Science*, 46(12), 1554–1568.
- Fernández-Martínez, F., Luna-Jiménez, C., Kleinlein, R., Griol, D., Callejas, Z., & Montero, J. M. (2022). Fine-tuning BERT models for intent recognition using a frequency cut-off strategy for domain-specific vocabulary extension. *Applied Sciences*, 12(3), 1–15.
- Feuerriegel, S., Hartmann, J., Janiesch, C., & Zschech, P. (2024). Generative AI. *Business & Information Systems Engineering*, 66(1), 111–126.

- Fitzgerald, B. (1997). The use of systems development methodologies in practice: a field study. *Information Systems Journal*, 7(3), 201–212.
- Fornell, C., & Larcker, D. F. (1981). Evaluating structural equation models with unobservable variables and measurement error. *Journal of Marketing Research*, 18(1), 39–50.
- Forsythe, D. E. (1993). Engineering knowledge: The construction of knowledge in artificial intelligence. *Social Studies of Science*, 23(3), 445–477.
- Foster, D. (2022). *Generative Deep Learning*. O'Reilly Media, Inc.
- Fountaine, T., McCarthy, B., & Saleh, T. (2019). Building the AI-powered organization. *Harvard Business Review*, 97(4), 62–73.
- Freise, L. R., Bruhin, O., Ritz, E., Li, M. M., & Leimeister, J. M. (2025). Code and Craft: How Generative AI Tools Facilitate Job Crafting in Software Development. *Hawaii International Conference on System Sciences (HICSS) Proceedings*.
- Fügener, A., Grahl, J., Gupta, A., & Ketter, W. (2021). Will Humans-in-the-Loop Become Borgs? Merits and Pitfalls of Working with AI. *MIS Quarterly*, 45(3), 1527–1556.
- Fügener, A., Grahl, J., Gupta, A., & Ketter, W. (2022). Cognitive challenges in human–artificial intelligence collaboration: Investigating the path toward productive delegation. *Information Systems Research*, 33(2), 678–696.
- Fürstenau, D., Baiyere, A., & Kliwer, N. (2019). A dynamic model of embeddedness in digital infrastructures. *Information Systems Research*, 30(4), 1319–1342.
- Gallivan, M. J., & Keil, M. (2003). The user–developer communication process: a critical case study. *Information Systems Journal*, 13(1), 37–68.
- Garcia, M. (2024). What Air Canada Lost In ‘Remarkable’ Lying AI Chatbot Case.
- Gerlich, M. (2025). AI tools in society: Impacts on cognitive offloading and the future of critical thinking. *Societies*, 15(6), 1–28.
- Gioia, D. A., Corley, K. G., & Hamilton, A. L. (2013). Seeking qualitative rigor in inductive research: Notes on the Gioia methodology. *Organizational Research Methods*, 16(1), 15–31.
- Glaser, V. L., Pollock, N., & D’Adderio, L. (2021). The biography of an algorithm: Performing algorithmic technologies in organizations. *Organization Theory*, 2(2), 1–27.
- Glikson, E., & Woolley, A. W. (2020). Human Trust in Artificial Intelligence: Review of Empirical Research. *Academy of Management Annals*, 14(2), 627–660.
- Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1). MIT press Cambridge.
- Gould, S. (1979). Age, job complexity, satisfaction, and performance. *Journal of Vocational Behavior*, 14(2), 209–223.
- Graves, T. L., Karr, A. F., Marron, J. S., & Siy, H. (2002). Predicting fault incidence using software change history. *IEEE Transactions on Software Engineering*, 26(7), 653–661.
- Gregor, S., & Benbasat, I. (1999). Explanations from intelligent systems: Theoretical foundations and implications for practice. *MIS Quarterly*, 23(4), 497–530.

- Gronsund, T., & Aanestad, M. (2020). Augmenting the algorithm: Emerging human-in-the-loop work configurations. *Journal of Strategic Information Systems*, 29(2), 1–16.
- Gulati, R., Wohlgezogen, F., & Zhelyazkov, P. (2012). The two facets of collaboration: Cooperation and coordination in strategic alliances. *Academy of Management Annals*, 6(1), 531–583.
- Hackman, J. R., & Oldham, G. R. (1976). Motivation through the design of work: Test of a theory. *Organizational Behavior and Human Performance*, 16(2), 250–279.
- Hadfield-Menell, D., & Hadfield, G. K. (2019). Incomplete contracting and AI alignment. *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, 417–422.
- Hahn, J., & Lee, G. (2021). The Complex Effects of Cross-Domain Knowledge on IS Development: A Simulation-Based Theory Development1. *MIS Quarterly*, 45(4), 2023–2054.
- Hair, J. F. (2014). *A primer on partial least squares structural equation modeling (PLS-SEM)*. Sage.
- Hair, J. F., Hult, G. T. M., Ringle, C. M., Sarstedt, M., Danks, N. P., & Ray, S. (2021). *Partial least squares structural equation modeling (PLS-SEM) using R: A workbook*. Springer Nature.
- Hair, J. F., Matthews, L. M., Matthews, R. L., & Sarstedt, M. (2017). PLS-SEM or CB-SEM: updated guidelines on which method to use. *International Journal of Multivariate Data Analysis*, 1(2), 107–123.
- Hair, J. F., Risher, J. J., Sarstedt, M., & Ringle, C. M. (2019). When to use and how to report the results of PLS-SEM. *European Business Review*, 31(1), 2–24.
- Halloran, D., Manchester, S., Moriarty, J., Riley, R., Rohrman, J., & Skramstad, T. (1978). Systems Development Quality Control. *MIS Quarterly*, 2(4), 1–13.
- Hanelt, A., Wiesche, M., Benlian, A., & Hinz, O. (2025). Opening The Network of Trust: How Domain Experts in Triadic Relationships Build Trust in Ai-Based Counterparts. *MIS Quarterly*, 32(1), 1–32.
- Hanseth, O., & Monteiro, E. (1997). Inscribing behaviour in information infrastructure standards. *Accounting, Management and Information Technologies*, 7(4), 183–211.
- Harter, D. E., Krishnan, M. S., & Slaughter, S. A. (2000). Effects of process maturity on quality, cycle time, and effort in software product development. *Management Science*, 46(4), 451–466.
- Hartmann, P., & Henkel, J. (2020). The rise of corporate science in AI: Data as a strategic resource. *Academy of Management Discoveries*, 6(3), 359–381.
- Hayes-Roth, F. (1997). Artificial intelligence: what works and what doesn't? *AI Magazine*, 18(2), 99–113.
- Hayes-Roth, F., & Jacobstein, N. (1994). The state of knowledge-based systems. *Communications of the ACM*, 37(3), 26–39.

- He, J., & King, W. R. (2008). The role of user participation in information systems development: Implications from a meta-analysis. *Journal of Management Information Systems*, 25(1), 301–331.
- He, Z., Li, S., Song, Y., & Cai, Z. (2024). Towards Building Condition-Based Cross-Modality Intention-Aware Human-AI Cooperation under VR Environment. *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 1–13.
- Henfridsson, O., & Lindgren, R. (2010). User involvement in developing mobile and temporarily interconnected systems. *Information Systems Journal*, 20(2), 119–135.
- Henseler, J., Ringle, C. M., & Sarstedt, M. (2015). A new criterion for assessing discriminant validity in variance-based structural equation modeling. *Journal of the Academy of Marketing Science*, 43(1), 115–135.
- Herbst, H. (1996). Business rules in systems analysis: a meta-model and repository system. *Information Systems*, 21(2), 147–166.
- Hickey, A. M., & Davis, A. M. (2004). A unified model of requirements elicitation. *Journal of Management Information Systems*, 20(4), 65–84.
- Hirschheim, R., Klein, H. K., & Newman, M. (1991). Information systems development as social action: Theoretical perspective and practice. *Omega*, 19(6), 587–608.
- Holland, C. P., Gibson, N., Light, B., & Kelly, S. (1999). A Business Perspective of Legacy Information Systems. *Communications of the AIS*, 2(7), 1–27.
- Hsu, J. S.-C., Lin, T.-C., Zheng, G.-T., & Hung, Y.-W. (2012). Users as knowledge co-producers in the information system development project. *International Journal of Project Management*, 30(1), 27–36.
- Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., & Qin, B. (2025). A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2), 1–55.
- Hunton, J. E., & Beeler, J. D. (1997). Effects of user participation in systems development: a longitudinal field experiment. *MIS Quarterly*, 21(4), 359–388.
- IDC. (2024). *IDC Unveils 2025 FutureScapes: Worldwide IT Industry Predictions*. <https://www.idc.com/getdoc.jsp?containerId=prUS52691924>
- Iivari, J., & Maansaari, J. (1998). The usage of systems development methods: are we stuck to old practices? *Information and Software Technology*, 40(9), 501–510.
- Imai, S. (2022). Is github copilot a substitute for human pair-programming? an empirical study. *Proceedings of the ACM/IEEE 44th International Conference on Software Engineering: Companion Proceedings*, 319–321.
- Jain, H., Padmanabhan, B., Pavlou, P. A., & Raghu, T. S. (2021). Editorial for the special section on humans, algorithms, and augmented intelligence: The future of work, organizations, and society. *Information Systems Research*, 32(3), 675–687.

- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An introduction to statistical learning - second edition*. Springer.
- Jenkin, T. A., Chan, Y. E., & Sabherwal, R. (2019). Changes Within and Across Projects. *MIS Quarterly*, 43(2), 649–A17.
- Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., & Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12), 1–38.
- Johns, G. (2006). The essential impact of context on organizational behavior. *Academy of Management Review*, 31(2), 386–408.
- Jöreskog, K. G. (1971). Simultaneous factor analysis in several populations. *Psychometrika*, 36(4), 409–426.
- Judge, T. A., Thoresen, C. J., Bono, J. E., & Patton, G. K. (2001). The job satisfaction–job performance relationship: A qualitative and quantitative review. *Psychological Bulletin*, 127(3), 376.
- Jussupow, E., Spohrer, K., Heinzl, A., & Gawlitza, J. (2021). Augmenting medical diagnosis decisions? An investigation into physicians’ decision-making process with artificial intelligence. *Information Systems Research*, 32(3), 713–735.
- Kane, G. C., Young, A. G., Majchrzak, A., & Ransbotham, S. (2021). Avoiding an oppressive future of machine learning: A design theory for emancipatory assistants. *MIS Quarterly*, 45(1), 371–396.
- Kaplan, B., & Duchon, D. (1988). Combining qualitative and quantitative methods in information systems research: a case study. *MIS Quarterly*, 12(4), 571–586.
- Kasthuriarachchy, B., Chetty, M., Karmakar, G., & Walls, D. (2020). Pre-trained language models with limited data for intent classification. *2020 International Joint Conference on Neural Networks*, 1–9.
- Keil, M., Rai, A., & Liu, S. (2013). How user risk and requirements risk moderate the effects of formal and informal control on the process performance of IT projects. *European Journal of Information Systems*, 22, 650–672.
- Kemp, A. (2024). Competitive advantage through artificial intelligence: Toward a theory of situated AI. *Academy of Management Review*, 49(3), 618–635.
- Kim, M., Zimmermann, T., DeLine, R., & Begel, A. (2016). The emerging role of data scientists on software development teams. *Proceedings of the 38th International Conference on Software Engineering*, 96–107.
- Kim, M., Zimmermann, T., DeLine, R., & Begel, A. (2017). Data scientists in software teams: State of the art and challenges. *IEEE Transactions on Software Engineering*, 44(11), 1024–1038.
- Kirsch, L. J. (1996). The management of complex tasks in organizations: Controlling the systems development process. *Organization Science*, 7(1), 1–21.

- Kling, R. (1977). The organizational context of user-centered software designs. *MIS Quarterly*, 1(4), 41–52.
- Kozar, K. A., & Mahlum, J. M. (1987). A user generated information system: an innovative development approach. *MIS Quarterly*, 11(2), 163–174.
- Krakowski, S. (2025). Human-AI agency in the age of generative AI. *Information and Organization*, 35(1–25), 100560.
- Kreuzberger, D., Kühl, N., & Hirschl, S. (2023). Machine learning operations (mlops): Overview, definition, and architecture. *IEEE Access*, 11, 31866–31879.
- Krishnan, M. S., Mukhopadhyay, T., & Zubrow, D. (1999). Software process models and project performance. *Information Systems Frontiers*, 1(3), 267–277.
- Kudaravalli, S., Faraj, S., & Johnson, S. L. (2017). A Configural Approach to Coordinating Expertise in Software Development Teams. *MIS Quarterly*, 41(1), 43–64.
- Laato, S., Mäntymäki, M., Minkkinen, M., Birkstedt, T., Islam, A. K. M., & Dennehy, D. (2022). Integrating machine learning with software development lifecycles: Insights from experts. *European Conference on Information Systems (ECIS) Proceedings*.
- Law, M., & Varanasi, R. A. (2025). Generative AI and Changing Work: Systematic Review of Practitioner-Led Work Transformations Through the Lens of Job Crafting. *International Conference on Human-Computer Interaction*, 131–152.
- Lawler, E., Cammann, C., Nadler, D., & Jenkins, D. (1979). Michigan organizational assessment questionnaire. *Journal of Vocational Behavior*.
- Lawrence, N. D. (2017). Living together: Mind and machine intelligence. *arXiv preprint arXiv:1705.07996*.
- Leavitt, H. J. (2013). Applied organizational change in industry: Structural, technological and humanistic approaches. In *Handbook of organizations* (pp. 1144–1170). Routledge.
- Lebovitz, S., Levina, N., & Lifshitz-Assaf, H. (2021). Is AI Ground Truth Really ‘True’? The Dangers of Training and Evaluating AI Tools Based on Experts’ Know-What. *MIS Quarterly*, 45(3), 1501–1525.
- Lebovitz, S., Lifshitz-Assaf, H., & Levina, N. (2022). To engage or not to engage with AI for critical judgments: How professionals deal with opacity when using AI for medical diagnosis. *Organization Science*, 33(1), 126–148.
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444.
- Lee, A. S. (1989). A scientific methodology for MIS case studies. *MIS Quarterly*, 13(1), 33–50.
- Lee, A. S., & Baskerville, R. L. (2003). Generalizing generalizability in information systems research. *Information Systems Research*, 14(3), 221–243.
- Lee, A. S., & Baskerville, R. L. (2012). Conceptualizing Generalizability: New Contributions and a Reply. *MIS Quarterly*, 36(3), 749–761.
- Lee, G., & Xia, W. (2010). Toward agile: an integrated analysis of quantitative and qualitative field data on software development agility. *MIS Quarterly*, 34(1), 87–114.

- Lee, S.-H., Shin, Y., & Baek, S. I. (2017). The impact of job demands and resources on job crafting. *Journal of Applied Business Research*, 33(4), 827–840.
- Leonardi, P. M. (2011). When flexible routines meet flexible technologies: Affordance, constraint, and the imbrication of human and material agencies. *MIS Quarterly*, 35(1), 147–167.
- Levina, N., & Vaast, E. (2005). The Emergence of Boundary Spanning Competence in Practice: Implications for Implementation and Use of Information Systems. *MIS Quarterly*, 29(2), 335–363.
- Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., & Rocktäschel, T. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459–9474.
- Li, J., Yang, H., Weng, Q., & Zhu, L. (2023). How different forms of job crafting relate to job satisfaction: The role of person-job fit and age. *Current Psychology*, 42(13), 11155–11169.
- Li, M., Peters, C., Poser, M., Eilers, K., & Elshan, E. (2022). ICT-enabled job crafting: How Business Unit Developers use Low-code Development Platforms to craft jobs. *International Conference on Information Systems (ICIS) Proceedings*.
- Lichtenthaler, P. W., & Fischbach, A. (2019). A meta-analysis on promotion-and prevention-focused job crafting. *European Journal of Work and Organizational Psychology*, 28(1), 30–50.
- Lifshitz-Assaf, H. (2018). Dismantling knowledge boundaries at NASA: The critical role of professional identity in open innovation. *Administrative Science Quarterly*, 63(4), 746–782.
- Lindebaum, D., & Fleming, P. (2024). ChatGPT undermines human reflexivity, scientific responsibility and responsible management research. *British Journal of Management*, 35(2), 566–575.
- Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9), 1–35.
- Lopper, E., Horstmann, K. T., & Hoppe, A. (2024). The Approach–Avoidance Job Crafting Scale: Development and validation of a measurement of the hierarchical structure of job crafting. *Applied Psychology*, 73(1), 93–134.
- Lou, B., & Wu, L. (2021). AI on Drugs: Can Artificial Intelligence Accelerate Drug Development? Evidence from a large-scale Examination of Bio-Pharma Firms. *MIS Quarterly*, 45(3), 1451–1482.
- Lyytinen, K., & Newman, M. (2008). Explaining information systems change: a punctuated socio-technical change model. *European Journal of Information Systems*, 17, 589–613.

- Lyytinen, K., Nickerson, J. V., & King, J. L. (2021). Metahuman systems= humans+ machines that learn. *Journal of Information Technology*, 36(4), 427–445.
- MacCormack, A., & Sturtevant, D. J. (2016). Technical debt and system architecture: The impact of coupling on defect-related activity. *Journal of Systems and Software*, 120, 170–182.
- Mahmood, M. A. (1987). System Development Methods-A Comparative Investigation. *MIS Quarterly*, 11(3), 293–311.
- Majchrzak, A., More, P. H. B., & Faraj, S. (2012). Transcending knowledge differences in cross-functional teams. *Organization Science*, 23(4), 951–970.
- Malone, T. W., & Crowston, K. (1994). The interdisciplinary study of coordination. *ACM Computing Surveys (CSUR)*, 26(1), 87–119.
- Mao, J.-Y., Vredenburg, K., Smith, P. W., & Carey, T. (2005). The state of user-centered design practice. *Communications of the ACM*, 48(3), 105–109.
- Markus, M. L., Majchrzak, A., & Gasser, L. (2002). A design theory for systems that support emergent knowledge processes. *MIS Quarterly*, 26(3), 179–212.
- Markus, M. L., & Silver, M. S. (2008). A foundation for the study of IT effects: A new look at DeSanctis and Poole's concepts of structural features and spirit. *Journal of the Association for Information Systems*, 9(10), 609–632.
- Maruping, L. M., & Matook, S. (2020). The Multiplex Nature of the Customer Representative Role in Agile Information Systems Development. *MIS Quarterly*, 44(3), 1411–1437.
- Matook, S., Lee, G., & Fitzgerald, B. (2021). MISQ research curation on information systems development. *MIS Quarterly*, 1–20.
- Matsuo, Y., LeCun, Y., Sahani, M., Precup, D., Silver, D., Sugiyama, M., Uchibe, E., & Morimoto, J. (2022). Deep learning, reinforcement learning, and world models. *Neural Networks*, 152, 267–275.
- Mayer, A.-S., Baygi, R. M., & Buwalda, R. (2025a). Generation AI: Job Crafting by Entry-Level Professionals in the Age of Generative AI: A.-S. Mayer et al. *Business & Information Systems Engineering*, 1–19.
- Mayer, H., Yee, L., Chui, M., & Roberts, R. (2025b). Superagency in the Workplace - Empowering people to unlock AI's full potential. <https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/superagency-in-the-workplace-empowering-people-to-unlock-ais-full-potential-at-work#/>
- McAllister, D. J. (1995). Affect-and cognition-based trust as foundations for interpersonal cooperation in organizations. *Academy of Management Journal*, 38(1), 24–59.
- Meyer, A. N., Barton, L. E., Murphy, G. C., Zimmermann, T., & Fritz, T. (2017). The work life of developers: Activities, switches and perceived productivity. *IEEE Transactions on Software Engineering*, 43(12), 1178–1193.
- Miceli-Barone, A. V., Barez, F., Konstantas, I., & Cohen, S. B. (2023). The larger they are, the harder they fail: Language models do not recognize identifier swaps in python.
- Microsoft. (2025). *Earnings Release FY25 Q4*. Microsoft. REDMOND, Wash.

- Mikalef, P., & Gupta, M. (2021). Artificial intelligence capability: Conceptualization, measurement calibration, and empirical study on its impact on organizational creativity and firm performance. *Information & Management*, 58(3), 1–20.
- Miles, M. B., & Huberman, A. M. (1994). *Qualitative Data Analysis: An Expanded Sourcebook*. Sage.
- Millière, R. (2023). *The Alignment Problem in Context*. arXiv: 2311.02147 [cs.AI]. arxiv.org
- Minelli, R., Mocci, A., & Lanza, M. (2015). I know what you did last summer—an investigation of how developers spend their time. *2015 IEEE 23rd international conference on program comprehension*, 25–35.
- Mingers, J. (2001). Combining IS research methods: towards a pluralist methodology. *Information Systems Research*, 12(3), 240–259.
- Mingers, J., & Standing, C. (2017). Why things happen—Developing the critical realist view of causal mechanisms. *Information and Organization*, 27(3), 171–189.
- Monaghan, B. D., & Bass, J. M. (2020). Redefining legacy: a technical debt perspective. *International Conference on Product-Focused Software Process Improvement*, 254–269.
- Muldoon, J., Cant, C., Wu, B., & Graham, M. (2024). A typology of artificial intelligence data work. *Big Data & Society*, 11(1), 1–13.
- Murphy-Hill, E., Jaspan, C., Sadowski, C., Shepherd, D., Phillips, M., Winter, C., Knight, A., Smith, E., & Jorde, M. (2019). What predicts software developers’ productivity? *IEEE Transactions on Software Engineering*, 47(3), 582–594.
- Murray, A., Rhymer, J., & Sirmon, D. G. (2021). Humans and technology: Forms of conjoined agency in organizations. *Academy of Management Review*, 46(3), 552–571.
- Naumann, J. D., & Jenkins, A. M. (1982). Prototyping: the new paradigm for systems development. *MIS Quarterly*, 6(3), 29–44.
- Newman, M., & Robey, D. (1992). A social process model of user-analyst relationships. *MIS Quarterly*, 16(2), 249–266.
- Ng, T. W. H., & Feldman, D. C. (2010). Organizational Tenure and Job Performance. *Journal of Management*, 36(5), 1220–1250.
- Ngo, R., Chan, L., & Mindermann, S. (2022). The alignment problem from a deep learning perspective.
- Nguyen-Duc, A., Cabrero-Daniel, B., Przybylek, A., Arora, C., Khanna, D., Herda, T., Rafiq, U., Melegati, J., Guerra, E., & Kemell, K.-K. (2023). Generative Artificial Intelligence for Software Engineering—A Research Agenda. *arXiv preprint arXiv:2310.18648*.
- Nicolini, D. (2012). *Practice theory, work, and organization: An introduction*. OUP Oxford.
- Nitzl, C., Roldan, J. L., & Cepeda, G. (2016). Mediation analysis in partial least squares path modeling: Helping researchers discuss more sophisticated models. *Industrial Management & Data Systems*, 116(9), 1849–1864.
- OECD. (2024). *Scoping the OECD AI principles*. OECD. <https://legalinstruments.oecd.org/en/instruments/OECD-LEGAL-0449>

- Okhuysen, G. A., & Bechky, B. A. (2009). 10 coordination in organizations: An integrative perspective. *Academy of Management Annals*, 3(1), 463–502.
- OpenAI. (2023). *GPT-4 technical report*. *arXiv 2303.08774* (tech. rep.). OpenAI.
- Orlikowski, W. J. (2000). Using technology and constituting structures: A practice lens for studying technology in organizations. *Organization Science*, 11(4), 404–428.
- Orlikowski, W. J., & Gash, D. C. (1994). Technological frames: making sense of information technology in organizations. *ACM Transactions on Information Systems (TOIS)*, 12(2), 174–207.
- Østerlie, T., & Monteiro, E. (2020). Digital sand: The becoming of digital representations. *Information and Organization*, 30(1).
- Ozkaya, I. (2020). What is really different in engineering AI-enabled systems? *IEEE Software*, 37(4), 3–6.
- Ozkaya, I. (2023). Application of large language models to software engineering tasks: Opportunities, risks, and implications. *IEEE Software*, 40(3), 4–8.
- Paleyes, A., Urma, R.-G., & Lawrence, N. D. (2022). Challenges in deploying machine learning: a survey of case studies. *ACM Computing Surveys*, 55(6), 1–29.
- Pan, S., Luo, L., Wang, Y., Chen, C., Wang, J., & Wu, X. (2024). Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*.
- Parker, S. K., & Grote, G. (2022). Automation, algorithms, and beyond: Why work design matters more than ever in a digital world. *Applied Psychology*, 71(4), 1171–1204.
- Patil, D. J. (2011). *Building Data Science Teams*. O'Reilly Media, Inc.
- Patnayakuni, R., Rai, A., & Tiwana, A. (2007). Systems development process improvement: A knowledge integration perspective. 54(2), 286–300.
- Paul, S., Yuan, L., Jain, H. K., Robert Jr, L. P., Spohrer, J., & Lifshitz-Assaf, H. (2022). Intelligence augmentation: Human factors in AI and future of work. *AIS Transactions on Human-Computer Interaction*, 14(3), 426–445.
- Pawlowski, S. D., & Robey, D. (2004). Bridging user organizations: Knowledge brokering and the work of information technology professionals. *MIS Quarterly*, 28(4), 645–672.
- Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of AI on developer productivity: Evidence from GitHub Copilot. *arXiv preprint arXiv:2302.06590*.
- Pentland, B. T., Yoo, Y., Recker, J., & Kim, I. (2022). From lock-in to transformation: A path-centric theory of emerging technology and organizing. *Organization Science*, 33(1), 194–211.
- Petrou, P., Demerouti, E., Peeters, M. C. W., Schaufeli, W. B., & Hetland, J. (2012). Crafting a job on a daily basis: Contextual correlates and the link to work engagement. *Journal of Organizational Behavior*, 33(8), 1120–1141.
- Pfeuffer, N., Baum, L., Stammer, W., Abdel-Karim, B. M., Schramowski, P., Bucher, A. M., Hügel, C., Rohde, G., Kersting, K., & Hinz, O. (2023). Explanatory interactive machine

- learning: establishing an action design research process for machine learning projects. *Business & Information Systems Engineering*, 65(6), 677–701.
- Pinski, M., Adam, M., & Benlian, A. (2023). AI knowledge: Improving AI delegation through human enablement. *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, 1–17.
- Pinsonneault, A., & Kraemer, K. (1993). Survey research methodology in management information systems: an assessment. *Journal of Management Information Systems*, 10(2), 75–105.
- Poldrack, R. A., Lu, T., & Beguš, G. (2023). AI-assisted coding: Experiments with GPT-4. *arXiv preprint arXiv:2304.13187*.
- Polyzotis, N., Roy, S., Whang, S. E., & Zinkevich, M. (2018). Data lifecycle challenges in production machine learning: a survey. *ACM Sigmod Record*, 47(2), 17–28.
- Preacher, K. J., & Hayes, A. F. (2008). Asymptotic and resampling strategies for assessing and comparing indirect effects in multiple mediator models. *Behavior Research Methods*, 40(3), 879–891.
- Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). *Improving language understanding by generative pre-training*. San Francisco, CA, USA.
- Rahwan, I., Cebrian, M., Obradovich, N., Bongard, J., Bonnefon, J.-F., Breazeal, C., Crandall, J. W., Christakis, N. A., Couzin, I. D., & Jackson, M. O. (2019). Machine behaviour. *Nature*, 568(7753), 477–486.
- Raisch, S., & Krakowski, S. (2021). Artificial Intelligence and Management: The Automation-Augmentation Paradox. *Academy of Management Review*, 46(1), 192–210.
- Raisch, S., & Fomina, K. (2025). Combining human and artificial intelligence: Hybrid problem-solving in organizations. *Academy of Management Review*, 50(2), 441–464.
- Ram, O., Levine, Y., Dalmedigos, I., Muhlgay, D., Shashua, A., Leyton-Brown, K., & Shoham, Y. (2023). In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11, 1316–1331.
- Ram, S., & Khatri, V. (2005). A comprehensive framework for modeling set-based business rules during conceptual database design. *Information Systems*, 30(2), 89–118.
- Ravichandran, T., & Rai, A. (1999). Total quality management in information systems development: key constructs and relationships. *Journal of Management Information Systems*, 16(3), 119–155.
- Ravichandran, T., & Rai, A. (2000). Quality management in systems development: an organizational system perspective. *MIS Quarterly*, 24(3), 381–415.
- Reay, T., Goodrick, E., Waldorff, S. B., & Casebeer, A. (2017). Getting leopards to change their spots: Co-creating a new professional role identity. *Academy of Management Journal*, 60(3), 1043–1070.
- Recker, J. (2021). *Scientific research in information systems: a beginner's guide*. Springer Nature.

- Riemenschneider, C. K., & Armstrong, D. J. (2021). The development of the perceived distinctiveness antecedent of information systems professional identity. *MIS Quarterly*, 45(3), 1149–1186.
- Rinta-Kahila, T., Penttinen, E., & Lyytinen, K. (2023). Getting trapped in technical debt: Sociotechnical analysis of a legacy system's replacement. *MIS Quarterly*, 47(1), 1–31.
- Ritz, E., Freise, L. R., & Li, M. M. (2025). Offloading to Digital Minds: How Generative AI Can Help to Craft Jobs. *Information Systems and Neuroscience: NeuroIS Retreat 2024, Vienna, Austria*, 66, 28–37.
- Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review*, 33, 1–39.
- Rosenkranz, C., Stray, V., & Wiesche, M. (2025). Navigating the new frontier of information systems engineering. *Business & Information Systems Engineering*, 67(1), 1–5.
- Rosenkranz, C., Vranešić, H., & Holten, R. (2014). Boundary interactions and motors of change in requirements elicitation: A dynamic perspective on knowledge sharing. *Journal of the Association for Information Systems*, 15(6), 306–345.
- Rosso, B. D., Dekas, K. H., & Wrzesniewski, A. (2010). On the meaning of work: A theoretical integration and review. *Research in Organizational Behavior*, 30, 91–127.
- Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Sauvestre, R., & Remez, T. (2023). Code Llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.
- Rudolph, C. W., Katz, I. M., Lavigne, K. N., & Zacher, H. (2017). Job crafting: A meta-analysis of relationships with individual differences, job characteristics, and work outcomes. *Journal of Vocational Behavior*, 102, 112–138.
- Rus, I., Lindvall, M., & Sinha, S. (2002). Knowledge management in software engineering. *IEEE Software*, 19(3), 26–38.
- Russell, S., & Norvig, P. (2016). *Artificial Intelligence: A Modern Approach*. Pearson.
- Rzepka, C., & Berger, B. (2018). User interaction with AI-enabled systems: A systematic review of IS research. *International Conference on Information Systems (ICIS) Proceedings*.
- Sætra, H. S. (2023). Generative AI: Here to stay, but for good? *Technology in Society*, 75, 1–5.
- Saffarizadeh, K., Keil, M., & Maruping, L. (2024). Relationship Between Trust in the Artificial Intelligence Creator and Trust in Artificial Intelligence Systems: The Crucial Role of Artificial Intelligence Alignment and Steerability. *Journal of Management Information Systems*, 41(3), 645–681.
- Saidani, I., Ouni, A., Chouchen, M., & Mkaouer, M. W. (2020). Predicting continuous integration build failures using evolutionary search. *Information and Software Technology*, 128, 1–16.
- Saldaña, J. (2009). *The Qualitative Coding Manual*. Thousand Oaks.
- Santos, I., Magalhaes, C., & Santos, R. d. S. (2025). Model-Assisted and Human-Guided: Perceptions and Practices of Software Professionals Using LLMs for Coding. *arXiv preprint arXiv:2510.09058*.

- Sarker, S., Chatterjee, S., Xiao, X., & Elbanna, A. (2019). The sociotechnical axis of cohesion for the IS discipline: Its historical legacy and its continued relevance. *MIS Quarterly*, 43(3), 695–720.
- Sarker, S., Xiao, X., Beaulieu, T., & Lee, A. S. (2018). Learning from first-generation qualitative approaches in the IS discipline: An evolutionary view and some implications for authors and evaluators (PART 1/2). *Journal of the Association for Information Systems*, 19(8), 752–774.
- Sauvola, J., Tarkoma, S., Klemettinen, M., Riekk, J., & Doermann, D. (2024). Future of software development with generative AI. *Automated Software Engineering*, 31(1), 1–26.
- Schaufeli, W. B., Bakker, A. B., & Salanova, M. (2006). The measurement of work engagement with a short questionnaire: A cross-national study. *Educational and Psychological Measurement*, 66(4), 701–716.
- Schuetz, S., & Venkatesh, V. (2020). The rise of human machines: How cognitive computing systems challenge assumptions of user-system interaction. *Journal of the Association for Information Systems*, 21(2), 460–482.
- Scott, K. (2023). Kevin Scott on 5 Ways Generative AI Will Transform Work.
- Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in machine learning systems. *Advances in Neural Information Processing Systems*, 28.
- Seidel, S., Berente, N., Lindberg, A., Lyytinen, K., Martinez, B., & Nickerson, J. V. (2020). Artificial intelligence and video game creation: A framework for the new logic of autonomous design. *Journal of Digital Social Research*, 2(3), 126–157.
- Seidel, S., Berente, N., Lindberg, A., Lyytinen, K., & Nickerson, J. V. (2019). Autonomous tools and design: a triple-loop approach to human-machine learning. *Communications of the ACM*, 62(1), 50–57.
- Seidel, S., Recker, J., & Vom Brocke, J. (2013). Sensemaking and sustainable practicing: Functional affordances of information systems in green transformations. *MIS Quarterly*, 37(4), 1275–1299.
- Selbst, A. D., Boyd, D., Friedler, S. A., Venkatasubramanian, S., & Vertesi, J. (2019). Fairness and abstraction in sociotechnical systems. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, 59–68.
- Shaft, T. M., & Vessey, I. (2006). The Role of Cognitive Fit in the Relationship between Software Comprehension and Modification. *MIS Quarterly*, 30(1), 29–55.
- Shah, A., Chernova, A., Tomson, E., Porter, L., Griswold, W. G., & Soosai Raj, A. G. (2025). Students' Use of GitHub Copilot for Working with Large Code Bases. *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*, 1050–1056.
- Shypula, A., Madaan, A., Zeng, Y., Alon, U., Gardner, J., Hashemi, M., Neubig, G., Ranganathan, P., Bastani, O., & Yazdanbakhsh, A. (2023). Learning performance-improving code edits.

- Sia, S. K., & Soh, C. (2007). An assessment of package–organisation misalignment: institutional and ontological structures. *European Journal of Information Systems*, 16(5), 568–583.
- Singla, A., Sukharevsky, A., Yee, L., Chui, M., & Hall, B. (2025). The state of AI: How organizations are rewiring to capture value. *McKinsey & Company*, 12.
- Sommerville, I. (2015). *Software engineering*. 10th.
- Sprenitzer, G. M. (1995). Psychological empowerment in the workplace: Dimensions, measurement, and validation. *Academy of Management Journal*, 38(5), 1442–1465.
- Star, S. L. (2010). This is not a boundary object: Reflections on the origin of a concept. *Science, Technology, & Human Values*, 35(5), 601–617.
- Star, S. L., & Griesemer, J. R. (1989). Institutional Ecology, 'Translations' and Boundary Objects: Amateurs and Professionals in Berkeley's Museum of Vertebrate Zoology, 1907–39. *Social Studies of Science*, 19(3), 387–420.
- Storey, M.-A., Zimmermann, T., Bird, C., Czerwonka, J., Murphy, B., & Kalliamvakou, E. (2019). Towards a theory of software developer job satisfaction and perceived productivity. *IEEE Transactions on Software Engineering*, 47(10), 2125–2142.
- Strauss, A., & Corbin, J. (1998). *Basics of qualitative research techniques*. Sage Publications.
- Stray, V., Barbala, A., & Wivestad, V. T. (2025). Human-AI Collaboration in Software Development: A Mixed-Methods Study of Developers' Use of GitHub Copilot and ChatGPT. *Proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering*, 1325–1332.
- Strich, F., Mayer, A.-S., & Fiedler, M. (2021). What do I do in a world of artificial intelligence? Investigating the impact of substitutive decision-making AI systems on employees' professional role identity. *Journal of the Association for Information Systems*, 22(2), 9.
- Strong, D. M., & Volkoff, O. (2010). Understanding Organization—Enterprise system fit: A path to theorizing the information technology artifact. *MIS Quarterly*, 34(4), 731–756.
- Strong, D. M., Volkoff, O., Johnson, S. A., Pelletier, L. R., Tulu, B., Bar-On, I., Trudel, J., & Garber, L. (2014). A theory of organization-EHR affordance actualization. *Journal of the Association for Information Systems*, 15(2), 53–85.
- Studer, S., Bui, T. B., Drescher, C., Hanuschkin, A., Winkler, L., Peters, S., & Müller, K.-R. (2021). Towards CRISP-ML (Q): a machine learning process model with quality assurance methodology. *Machine Learning and Knowledge Extraction*, 3(2), 392–413.
- Sturm, T., Gerlach, J. P., Pumplun, L., Mesbah, N., Peters, F., Tauchert, C., Nan, N., & Buxmann, P. (2021). Coordinating Human and Machine Learning for Effective Organizational Learning. *MIS Quarterly*, 45(3), 1581–1602.
- Subramanyam, R., & Krishnan, M. S. (2003). Empirical analysis of ck metrics for object-oriented design complexity: Implications for software defects. *IEEE Transactions on Software Engineering*, 29(4), 297–310.
- Sundberg, L., & Holmström, J. (2024). Innovating by prompting: How to facilitate innovation in the age of generative AI. *Business Horizons*.

- Sunitha, E. V., & Samuel, P. (2019). Automatic code generation from UML state chart diagrams. *IEEE Access*, 7, 8591–8608.
- Susarla, A., Gopal, R., Thatcher, J. B., & Sarker, S. (2023). The Janus effect of generative AI: Charting the path for responsible conduct of scholarly activities in information systems. *Information Systems Research*, 34(2), 399–408.
- Swanson, E. B., & Beath, C. M. (1989). Reconstructing the systems development organization. *MIS Quarterly*, 13(3), 293–307.
- Tarafdar, M., & Saunders, C. (2022). Remote, Mobile, and Blue-Collar: ICT-Enabled Job Crafting to Elevate Occupational Well-Being. *Journal of the Association for Information Systems*, 23(3), 707–749.
- Teodorescu, M. H. M., Morse, L., Awwad, Y., & Kane, G. C. (2021). Failures of Fairness in Automation Require a Deeper Understanding of Human-ML Augmentation. *MIS Quarterly*, 45(3), 1483–1499.
- Tesch, D., Sobol, M. G., Klein, G., & Jiang, J. J. (2009). User and developer common knowledge: Effect on the success of information system development projects. *International Journal of Project Management*, 27(7), 657–664.
- Teye-Kwadjio, E., & de Bruin, G. P. (2022). Rasch analysis of the proactive personality scale. *Psychological Reports*, 125(5), 2788–2806.
- Thenoz, E., Raymond, M., Monnier-Senicourt, L., Jacob, F., Hemon-Hildgen, A., & Rowe, F. (2024). Integrating Generative AI into Work: A Case Study of Changes to Core Job Characteristics. *International Conference on Information Systems (ICIS) Proceedings*.
- Thomas, D., & Hunt, A. (2019). *The Pragmatic Programmer: your journey to mastery*. Addison-Wesley Professional.
- Tian, Q., Bai, J., & Wu, T. (2022). Should we be” challenging” employees? A study of job complexity and job crafting. *International Journal of Hospitality Management*, 102, 1–15.
- Tim, Y., Pan, S. L., Ractham, P., & Kaewkitipong, L. (2017). Digitally enabled disaster response: the emergence of social media as boundary objects in a flooding disaster. *Information Systems Journal*, 27(2), 197–232.
- Tims, M., & Bakker, A. B. (2010). Job crafting: Towards a new model of individual job redesign. *SA Journal of Industrial Psychology*, 36(2), 1–9.
- Tims, M., Bakker, A. B., & Derks, D. (2012). Development and validation of the job crafting scale. *Journal of Vocational Behavior*, 80(1), 173–186.
- Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., & Bhosale, S. (2023). Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
- Ulfesnes, R., Mikalsen, M., Sporse, T. T., & Hatling, M. (2023). TECHNOLOGY FOR KNOWLEDGE WORK: A RELATIONAL PERSPECTIVE. *European Conference on Information Systems (ECIS) Proceedings*.

- Ullsnes, R., Moe, N. B., Stray, V., & Skarpen, M. (2024). Transforming software development with generative AI: Empirical insights on collaboration and workflow. In *Generative ai for effective software development* (pp. 219–234). Springer.
- Vaast, E., & Pinsonneault, A. (2021). When Digital Technologies Enable and Threaten Occupational Identity: The Delicate Balancing Act of Data Scientists. *MIS Quarterly*, 45(3), 1087–1112.
- van den Broek, E., Sergeeva, A., & Huysman, M. (2021). When The Machine Meets The Expert: An Ethnography of Developing AI for Hiring. *MIS Quarterly*, 45(3), 1557–1580.
- Van Dis, E. A. M., Bollen, J., Zuidema, W., Van Rooij, R., & Bockting, C. L. (2023). ChatGPT: five priorities for research. *Nature*, 614(7947), 224–226.
- Vara, J. M., & Marcos, E. (2012). A framework for model-driven development of information systems: Technical decisions and lessons learned. *Journal of Systems and Software*, 85(10), 2368–2384.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.
- Venkatesh, V., Brown, S. A., Maruping, L. M., & Bala, H. (2008). Predicting different conceptualizations of system use: The competing roles of behavioral intention, facilitating conditions, and behavioral expectation. *MIS Quarterly*, 32(3), 483–502.
- Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User Acceptance of Information Technology: Toward a Unified View. *MIS Quarterly*, 27(3), 425–478. <http://www.jstor.org/stable/30036540>
- Venkatesh, V., Thong, J. Y. L., Chan, F. K. Y., Hoehle, H., & Spohrer, K. (2020). How agile software development methods reduce work exhaustion: Insights on role perceptions and organizational skills. *Information Systems Journal*, 30(4), 733–761.
- Vessey, I., & Galletta, D. (1991). Cognitive fit: An empirical study of information acquisition. *Information Systems Research*, 2(1), 63–84.
- von Krogh, G. (2018). Artificial intelligence in organizations: New opportunities for phenomenon-based theorizing. *Academy of Management Discoveries*, 4(4), 404–409.
- Von Krogh, G., Spaeth, S., & Haefliger, S. (2005). Knowledge reuse in open source software: An exploratory study of 15 open source projects. *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, 198b–198b.
- Waardenburg, L., & Huysman, M. (2022). From coexistence to co-creation: Blurring boundaries in the age of AI. *Information and Organization*, 32(4), 1–11.
- Wagner, S., & Ruhe, M. (2018). A systematic review of productivity factors in software development. *arXiv preprint arXiv:1801.06475*.
- Walz, D. B., Elam, J. J., & Curtis, B. (1993). Inside a software design team: knowledge acquisition, sharing, and integration. *Communications of the ACM*, 36(10), 63–77.

- Wan, Z., Xia, X., Lo, D., & Murphy, G. C. (2019). How does machine learning change software development practices? *IEEE Transactions on Software Engineering*, 47(9), 1857–1871.
- Wand, Y., & Weber, R. (1995). On the deep structure of information systems. *Information Systems Journal*, 5(3), 203–223.
- Watson, H. J., & Frolick, M. N. (1993). Determining Information Requirements for an EIS. *MIS Quarterly*, 17(3), 255–269.
- Weber, T., Brandmaier, M., Schmidt, A., & Mayer, S. (2024). Significant productivity gains through programming with large language models. *Proceedings of the ACM on Human-Computer Interaction*, 8(256), 1–29.
- Weisz, J. D., Muller, M., Ross, S. I., Martinez, F., Houde, S., Agarwal, M., Talamadupula, K., & Richards, J. T. (2022). Better together? an evaluation of AI-supported code translation. *27th International Conference on Intelligent User Interfaces*, 369–391.
- Welsh, M. (2022). The end of programming. *Communications of the ACM*, 66(1), 34–35.
- White, J., Hays, S., Fu, Q., Spencer-Smith, J., & Schmidt, D. C. (2023). Chatgpt prompt patterns for improving code quality, refactoring, requirements elicitation, and software design. *arXiv preprint arXiv:2303.07839*.
- Wolf, Y., Wies, N., Avnery, O., Levine, Y., & Shashua, A. (2023). Fundamental limitations of alignment in large language models.
- Wrzesniewski, A., & Dutton, J. E. (2001). Crafting a job: Revisioning employees as active crafters of their work. *Academy of Management Review*, 26(2), 179–201.
- Wu, S., Liu, Y., Ruan, M., Chen, S., & Xie, X.-Y. (2025). Human-generative AI collaboration enhances task performance but undermines human’s intrinsic motivation. *Scientific Reports*, 15(1–30), 15105.
- Wu, Z., Qiu, L., Ross, A., Akyürek, E., Chen, B., Wang, B., Kim, N., Andreas, J., & Kim, Y. (2023). Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. *arXiv preprint arXiv:2307.02477*.
- Wu-Gehbauer, M., & Rosenkranz, C. (2024). Towards a Framework for Developing Artificial Intelligence Systems for Human-Artificial Intelligence Augmentation. *European Conference on Information Systems (ECIS) Proceedings*.
- Xu, F. F., Vasilescu, B., & Neubig, G. (2022). In-ide code generation from natural language: Promise and challenges. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 31(2), 1–47.
- Yang, J., Jin, H., Tang, R., Han, X., Feng, Q., Jiang, H., Zhong, S., Yin, B., & Hu, X. (2024). Harnessing the power of LLMs in practice: A survey on ChatGPT and beyond. *ACM Transactions on Knowledge Discovery from Data*, 18(6), 1–32.
- Yin, R. K. (2009). *Case study research: Design and methods* (Vol. 5). Sage Publications.
- Yin, R. K. (2018). *Case study research and applications* (Vol. 6). Sage Thousand Oaks, CA.

- Yuan, Z., Lou, Y., Liu, M., Ding, S., Wang, K., Chen, Y., & Peng, X. (2023). No more manual tests? evaluating and improving chatgpt for unit test generation. *arXiv preprint arXiv: 2305.04207*.
- Zhang, F., & Parker, S. K. (2019). Reorienting job crafting research: A hierarchical structure of job crafting concepts and integrative review. *Journal of Organizational Behavior*, 40(2), 126–146.
- Zhang, J., & Norman, D. A. (1994). Representations in Distributed Cognitive Tasks. *Cognitive Science*, 18(1), 87–122.
- Zhang, J., Chen, Y., Liu, C., Niu, N., & Wang, Y. (2023). Empirical Evaluation of ChatGPT on Requirements Information Retrieval Under Zero-Shot Setting. *2023 International Conference on Intelligent Computing and Next Generation Networks (ICNGN)*, 1–6.
- Zhang, Z., Hummel, J. T., Nandhakumar, J., & Waardenburg, L. (2020). Addressing the Key Challenges of Developing Machine Learning AI Systems for Knowledge-Intensive Work. *MIS Quarterly Executive*, 19(4), 221–238.
- Zhang, Z., Yoo, Y., Lyytinen, K., & Lindberg, A. (2021). The unknowability of autonomous tools and the liminal experience of their use. *Information Systems Research*, 32(4), 1192–1213.
- Zhao, X., Lynch Jr, J. G., & Chen, Q. (2010). Reconsidering Baron and Kenny: Myths and truths about mediation analysis. *Journal of Consumer Research*, 37(2), 197–206.
- Ziegler, A., Kalliamvakou, E., Li, X. A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., & Aftandilian, E. (2022). Productivity assessment of neural code completion. *Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming*, 21–29.
- Ziegler, A., Kalliamvakou, E., Li, X. A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., & Aftandilian, E. (2024). Measuring github copilot’s impact on productivity. *Communications of the ACM*, 67(3), 54–63.
- Zmud, R. W. (1980). Management of large software development efforts. *MIS Quarterly*, 4(2), 45–55.