Please quote as: Bruhin, O., Ebel, P. & Elshan, E. (2024). Exploring The Paradoxical Tensions in Low Code Environments. International Conference on Information Systems (ICIS), Bangkok, Thailand.

Exploring The Paradoxical Tensions in Low Code Environments

Completed Research Paper

Olivia Bruhin
Institute of Information Systems and Digital Business, University of St.Gallen, Switzerland
olivia.bruhin@unisg.ch

Philipp Ebel
Institute of Information Systems and Digital Business, University of St.Gallen, Switzerland
philipp.ebel@unisg.ch

Edona Elshan
KIN Center for Digital Innovation, Amsterdam, The Netherlands
e.elshan@vu.nl

Abstract

The rapid acceleration of digitalization has increased the demand for dynamic software development, exposing a significant skills gap in IT. Low Code Development Platforms (LCDP) have emerged as a crucial solution, enabling faster development cycles and democratizing software creation. However, integrating citizen and professional developers within these platforms introduces new challenges, particularly in task division, collaboration, and governance. This study explores the dynamic of the interactions in low code environments, using paradox theory to analyze the resulting tensions – the paradox of empowerment, the paradox of security, and the paradox of harmonization – and their impact on both innovation and efficiency. Our findings provide critical insights into the roles, collaborative processes, and governance frameworks necessary for effective and balanced LCDP implementation. These insights provide a framework for effective and balanced LCDP implementation, offering organizations strategies to enhance innovation while managing the inherent complexities within these environments.

Keywords: Collaboration Dynamics, Low Code Development, Paradox Theory

Introduction

The rapid acceleration of digitalization in recent years has significantly increased the demand for dynamic software development. This growth is driven by the demand for both frontend and backend solutions across various industries, as organizations strive to keep pace with technological advancements and foster digital transformation initiatives. For example, Carroll and Maher (2023) describe how Shell's digital transformation necessitated a shift towards do-it-yourself (DIY) software development, reflecting a broader industry trend towards internal capability building. This changing landscape has created a strong demand for IT specialists with both theoretical knowledge and practical experience. Despite this growing need, there is still a significant gap in meeting this demand. A survey by Statista (2022) found that 50% of 1,785 companies worldwide have faced a persistent shortage of IT skills over the past five years, which has hindered their technological progress. This identified skills gap underscores the urgent need for strategic solutions to fully leverage the benefits of digitalization.

To address these challenges, Low Code Development Platforms (LCDP) have emerged as a promising solution. LCDP enable rapid software and workflow development with minimal need for extensive programming expertise, effectively bridging the existing IT skill gap. This approach not only enables shorter development cycles, leading to cost savings but also results in enhanced end-user experiences (Lethbridge, 2021; Ullrich et al., 2021). Additionally, LCDP play a crucial role in democratizing software development by empowering 'citizen developers', which can be characterized as domain experts with limited programming knowledge that are developing software for their respective business unit using development and runtime environments sanctioned by corporate IT (Tisi et al., 2021; Lebens et al. 2021). This democratization leads to a more diverse pool of contributors in software development, potentially fostering increased creativity and effectiveness in digital solutions (Krejci et al., 2021; Iho et al. 2021).

Despite these benefits, the increased diversity introduced by LCDP can also pose challenges for organizations (Rokis & Kirikova, 2022). Thus, adopting a balanced approach becomes imperative to harness the benefits of democratization while mitigating potential associated drawbacks. This perspective is vital for organizations seeking to leverage LCDP. A key issue lies in the division of tasks between citizen developers and professional developers. With the rise of LCDP, IT departments are increasingly tasked with overseeing citizen development, which includes managing LCDP infrastructure and setting guidelines for these developers (Hoogsteen & Borgmann, 2022). In this regard, previous research suggested that IT departments should work collaboratively with citizen developers, providing support and partnership rather than merely regulating them (Wang et al., 2021; Carroll & Maher, 2023). Such collaborative efforts are often facilitated through regular interactions and knowledge sharing within dedicated communities or by forming cross-functional teams (Iho et al. 2021). However, as pointed out by Binzer and Winkler (2022), this raises several unresolved questions regarding the organization of collaboration between citizen developers and professional developers. Consequently, this exploratory study aims to examine the collaboration dynamics within low code environments, particularly focusing on the interplay and conflicts between these two groups. Specifically, it focuses on understanding the tensions resulting from the different demands of these two groups, with the intent to provide insights into how these demands can be managed in a productive manner. By exploring this specific aspect, we aim to contribute a first foundation for future on this topic. Our research is guided by the following refined research question:

RQ*: How do the tensions between citizen and professional developers manifest in low code development environments, and what strategies can organizations employ to effectively manage these tensions?*

The remainder of this paper is organized as follows: In Section 2, we present the theoretical background of LCDPs, focusing on their role in fostering innovation and the application of paradox theory as our conceptual framework. In Section 3, we describe our methodological approach. In Section 4, we present the study's findings. In Section 5, we discuss the conclusions, limitations, as well as theoretical and practical implications of our research. Finally, in Section 6, we conclude this study by outlining future research directions.

Theoretical Background

Low Code Development Platforms (LCDPs) and Their Contribution to Innovation

The term LCDPs, initially coined by Forrester Research, refers to “platforms that enable rapid application delivery with a minimum of hand coding, and quick setup and deployment” (Richardson et al., 2014, p. 2). These cloud-based software platforms, leveraging model-driven engineering principles (Luo et al., 2021), facilitate the design, development, and deployment of software applications. They enable organizations to rapidly translate business and development requirements into applications with minimal manual coding. This agility is particularly crucial as organizations face increasing pressure to adapt quickly to market changes and technological advancements.

LCDPs democratize software development by enabling both professional developers and citizen developers to contribute to application development (Adrian et al., 2020; Bock & Frank, 2021; Kletti, 2021). While professional developers, whose primary job function lies in the development of new software, are typically located in the IT-department, citizen developers are typically located outside the IT-department and address immediate business needs. This division of labor, however, requires a clear delineation of roles and responsibilities to avoid overlap and ensure effective teamwork. From a business perspective, LCDPs

enhance understanding of IT issues, fostering better communication between technical and non-technical staff (Waszkowski, 2019). They lower barriers for non-technical employees to contribute to software development, leading to more innovative and responsive adaptations to market demands (Sahay et al., 2020). On the IT side, involving citizen developers can reduce the burden of bug fixes and maintenance, allowing professional developers to concentrate on more complex and innovative projects (Alsaadi et al., 2021; Rafi et al., 2022). While LCDPs offer numerous benefits, they also present challenges. The involvement of citizen developers can lead to issues in software quality and integration with existing systems. Additionally, managing the collaboration between professional and citizen developers requires careful coordination to ensure that both groups contribute effectively without stepping into each other's domains.

Paradox Theory

Paradox theory is an interdisciplinary perspective that seeks to understand and explain the complex, contradictory, and competing demands that exist in various aspects of organizational life, including strategy, structure, leadership, and innovation (Cameron & Quinn, 1988). It acknowledges that organizations and individuals often face paradoxical situations where they must simultaneously manage opposing forces, such as stability and change, cooperation and competition, or autonomy and control (Smith & Lewis, 2011). Unlike other approaches that advocate for resolving tensions by prioritizing one demand over another, paradox theory posits that organizations can and should attend to these competing demands simultaneously to ensure long-term sustainability (Cameron, 1986; Lewis, 2000). A paradox, in this context, is defined as the coexistence of contradictory yet interdependent elements that persist over time (Smith & Lewis, 2011).

A central premise of paradox theory is that these tensions are not only inevitable but also valuable, as they can serve as catalysts for learning, creativity, and adaptation (Andriopoulos & Lewis, 2009). Rather than attempting to resolve or avoid these contradictions, paradox theory encourages embracing them, seeking to understand their underlying dynamics, and leveraging them as opportunities for growth and development (Poole & Van de Ven, 1989). By adopting this perspective, researchers can explore the interplay between various technological, organizational, and individual factors, shedding light on the challenges and opportunities that emerge from these complex relationships.

Paradox theory has been employed in various information systems research to explore the complex and often contradictory nature of technology implementation, usage, and management. For example, Marabelli and Newell (2012) employed paradox theory to examine the adoption of electronic medical record systems in healthcare organizations. They identified the paradoxes that emerged between standardization and customization, illustrating the tensions between the need for standardization to improve data sharing and the need for customization to address local practices and requirements. In another study, Robey et al. (2002) utilized paradox theory to investigate the relationship between IT-enabled organizational control and emergent self-organization in a global software development project. They identified the coexistence of contradictory forces, such as centralized control and decentralized autonomy, which fostered both challenges and opportunities for the organization.

In the context of low code environments, the relationship between citizen and professional developers can be framed as paradoxical because it involves ongoing, seemingly contradictory demands for innovation and control. These demands do not merely represent conflicting interests that need resolution; rather, they are interwoven elements that must be managed concurrently to achieve sustained organizational success. Thus, employing paradox theory as a lens to investigate the tensions between citizen and professional developers in the context of LCDPs and user groups can provide valuable insights into the challenges and opportunities that arise from these tensions. By recognizing and embracing these paradoxes, organizations can develop strategies to leverage these tensions for innovation and efficiency.

In sum, paradox theory offers a suitable framework for understanding the complex dynamics between professional developers and citizen developers in LCDPs. Paradox theory allows us to identify and manage contradictory yet interdependent elements, such as autonomy vs. governance, democratization vs. protection, and individualization vs. integration.

This is in line with existing research, which argues that examining paradoxical tensions and their underlying realities, can provide important insights into how conflicts manifest and affect organizational outcomes

(Schad & Bansal, 2018). In addition to that, exploring paradoxes can help organizations move beyond simplistic either/or thinking to more integrated both/and approaches, facilitating a more collaborative and innovative development environment (Lewis, 2000). Thus, paradox theory proves instrumental in transforming potential development hurdles into opportunities for growth and innovation within organizations.

Methodological Approach

To address the research question, we adopted a qualitative research design to explore the dynamics of Low Code Development (LCD) and their impact on both professional and citizen developers. We approached the study with a combination of inductive and deductive reasoning. Initially, our exploration was inductive, focusing on emergent themes from the data. Subsequently, we applied deductive reasoning to interpret these themes through the lens of existing theoretical frameworks.

We conducted interviews with citizen developers, non-IT professionals, who create new business applications using development and runtime environments sanctioned by corporate IT (Lebens et al. 2021) and with professional developers, whose primary job involves software development and who are typically located in the IT-department. Following the methodological guidelines proposed by Gioia et al. (2013), we employed an inductive approach, focusing on interviewing "knowledgeable agents" – participants with substantial expertise in LCD. Our sample included individuals proficient in creating applications for either internal use or external customers with LCDPs. This criterion was consistently applied across citizen and professional developers, ensuring a comprehensive understanding of LCDP usage patterns across diverse skill levels. After completing and transcribing the expert interviews, we engaged in an iterative analysis process to derive conclusive insights. This approach enabled a detailed exploration of the experiences and perspectives shared by participants, providing a robust foundation for drawing conclusions on the role and impact of citizen and professional developers in modern software development practices. Through this inductive process, we identified key themes and patterns that emerged naturally from the data.

After identifying these emergent themes, we applied Paradox Theory as a deductive framework to further analyze and interpret the data. This theoretical lens allowed us to understand the underlying tensions between autonomy versus control, accessibility versus compliance, and individualization versus integration.

Data Collection

Data was collected through individual interviews to avoid bias from group dynamics. We developed an interview guide with eight questions to understand how participants work with LCDPs. The guide covered: (1) Experience with LCDPs, (2) Role Differentiation and (3) Collaboration Dynamics. To gather rich qualitative insights, we conducted semi-structured interviews following the method outlined by Myers & Newman (2007) via the video conferencing tools Microsoft Teams[1] and Zoom[2]. The average interview lasted 48 minutes, allowing for in-depth exploration of each participant's experience.

The study includes a diverse range of organizations, classified based on size: small (up to 250 employees), medium (250-1000 employees), and large (over 1000 employees). The participants' roles varied across industries, providing a rich tapestry of insights (refer to Table 1 for detailed participant demographics).

ID	Firm Size	Industry	Job Role	Role in LCD
I1	Large	Auditing	Senior Manager Digital Transformation	Citizen Developer
I2	Large	IT-Services & IT-Consulting	Head of Low Code	Professional Developer
I3	Large	IT-Services & IT-Consulting	Business & Integration Architecture Associate	Professional Developer

[1] https://www.microsoft.com/microsoft-teams

[2] https://zoom.us/

I4	Small	Education	Co-Founder Low Code Lab	Professional Developer
I5	Large	Business Consulting	Practice IT Expert	Professional Developer
I6	Small	IT-Services & IT-Consulting	Co-Founder	Professional Developer
I7	Large	Manufacturing	Lead Business Process Automation	Professional Developer
I8	Large	Software Development	Enterprise Account Executive	Professional Developer
I9	Large	Banking	Lead Web-Development	Professional Developer
I10	Large	Retail	IT Business Partner	Citizen Developer
I11	Large	Biotech	Owner	Citizen Developer
I12	Large	Biotech	Director Software Architecture	Professional Developer
I13	Small	Banking	Project Lead	Citizen Developer
I14	Small	Software Development	Senior Software Engineer	Professional Developer
I15	Small	IT-Services & IT-Consulting	Senior Consultant	Citizen Developer
I16	Large	IT-Services & IT-Consulting	Head of Consulting	Professional Developer
I17	Large	Utility Supplier	Project Lead	Citizen Developer
I18	Large	Retail	Head of IT	Professional Developer
I19	Small	IT-Services & IT-Consulting	Business Development	Citizen Developer
I20	Small	Business Consulting	IT Strategy Consultant	Citizen Developer
I21	Large	Utility Supplier	Chief Digital Officer	Citizen Developer
I22	Small	Business Consulting	Consultant	Citizen Developer
I23	Large	Software Development	Senior Advisor	Professional Developer
I24	Small	Business Consulting	Consultant	Citizen Developer
I25	Medium	Software Development	Senior Advisor	Citizen Developer
I26	Large	IT-Services & IT-Consulting	Product Leader	Citizen Developer
I27	Large	IT-Services & IT-Consulting	Chief Technology Officer	Professional Developer
I28	Medium	Software Development	IT-Consultant	Professional Developer

Table 1. Overview of Interview Participants.

Data Analysis

Two researchers were involved in the data extraction process, using interview transcripts and applying open, axial, and selective coding techniques (Saldaña, 2021), as outlined by Strauss & Corbin (1998). This approach allowed us to capture expert expressions while systematically organizing the data. Any coding discrepancies were discussed thoroughly within our broader co-author team, ensuring a consensus-based approach to interpretation and analysis. The AI-transcription tool Atlas.ti[3] facilitated the transcription and

[3] https://atlasti.com/

coding of elements identified in the interviews. Most interviews and analyses were conducted in German, with the final coding translated into English to preserve the intended meaning.

To validate the coding iterations and develop first-order constructs, iterative discussions were held in the co-author team following the guidelines of Forman and Damschroder (2007), ensuring the stability, validity, and reproducibility of the research results. It is acknowledged, however, that deriving codes from interview data inherently involves subjective bias due to individual human judgment.

The process of identifying the paradoxes, underlying causes of tensions and developing strategies to manage them was conducted through a systematic analysis of the interview data. Initially, the interviews were designed to explore the experiences of both citizen and professional developers with LCDPs. During the interviews, participants shared their perspectives on various challenges and conflicts they encountered in their work, particularly those arising from differing roles, expectations, and responsibilities.

To systematically identify the causes of tensions, we employed a multi-phase coding process. First, the interview transcripts were subjected to open coding, where recurring themes and patterns related to developer roles, organizational dynamics, and the use of LCDPs were identified. This stage helped in surfacing the specific areas of conflict and tension that participants repeatedly mentioned. An excerpt of illustrative open codes, along with illustrative interview data, is presented in Table 2.

Illustrative Data	**Open Code**
“Often, I find myself hitting a wall with the limitations of low code tools. That's when I have to reach out to our IT department for help, which can be a bit intimidating." (Interviewee 10)	Citizen developers requiring professional assistance
"We're constantly juggling the need for uniformity in our applications with the demand for tailored solutions that address specific departmental challenges." (Interviewee 6)	Balancing consistency and unique solutions
" Sometimes, I feel like I'm not cut out for this. I rely heavily on our tech team, and it makes me question my own capabilities in handling these projects. " (Interviewee 20)	Feelings of inadequacy in fulfilling roles
“I constantly second-guess my decisions, wondering if I'm too dependent on our professional developers for even the smallest issues.” (Interviewee 1)	Self-doubt due to reliance on others
"There's always this tug-of-war between what we, as business users, want to achieve and what our IT colleagues deem feasible or within policy guidelines." (Interviewee 26)	Conflicts in differing priorities and expectations

Table 2. Illustrative Example of the Open Coding Process.

Following the initial process of open coding on the interview transcripts and establishing relationships among the open codes (axial coding), we defined the core variables. Axial coding allowed us to establish relationships between the identified themes, linking them to broader categories, such as autonomy, control, accessibility, compliance, and standardization. This process enabled us to pinpoint the underlying causes of the tensions by analyzing how these tensions manifested in the interactions between citizen and professional developers.

Table 3 provides a comprehensive summary of the axial codes, the identified sub-categories, and illustrative quotes from interviews.

Axial Code	Sub-Category	Illustrative Data
Need for Autonomy vs. Need for Governance	Striking a balance between control and autonomy	*"It's about giving them the freedom to innovate yet ensuring we don't compromise on our tech standards. " (Interviewee 12)*
	Citizen developers requiring professional assistance	*"Often, I find myself hitting a wall with the limitations of low code tools. That's when I have to reach out to our IT department for help, (...)" (Interviewee 10)*
Need for Democratization vs. Need for Protection	Adherence to IT security and compliance standards	*"Every app built by our citizen developers goes through a rigorous security check. It's non-negotiable " (Interviewee 5)*
	Data protection concerns	*"There's always a fear that someone might accidentally expose sensitive data. We're working on making our systems foolproof. " (Interviewee 14)*
Need for Individualization vs. Need for Integration	Balancing consistency and unique solutions	*"We aim for a middle ground – standard enough to be reliable, but flexible enough for department-specific needs. " (Interviewee 9)*
	Democratization of software development vs. need for complex programming skills	*"LCDPs have opened doors for many, but there's still a gap that only seasoned developers can fill, especially for complex integrations." (Interviewee 4)*

Table 3. Overview of the Axial Coding Process.

In line with Paradox Theory, our analysis revealed that the interplay between the need for autonomy vs. the need for governance, the need for democratization vs. the need for protection and the need for individualization vs. the need for integration are prevalent in the context of LCDPs. These paradoxical tensions were not just challenges to be overcome; they also presented opportunities for learning, creativity, and organizational growth. Our findings suggest that recognizing and effectively managing these paradoxes is crucial for organizations to leverage the full potential of LCDPs.

Application of Paradox Theory

The theoretical coding, following focused coding (Charmaz, 2006), led us to consider paradox theory as a potential meta-theoretical lens. This shift was motivated by data highlighting the actions facilitated by LCDPs and the inherent contradictions and tensions within these actions. Paradox theory, which emphasizes coexisting contradictions and their dynamic interplay, aligns well with the Grounded Theory Method (GTM) and offers a nuanced understanding of complex phenomena (Schad et al., 2016; Smith & Lewis, 2011). This approach is consistent with the use of metatheoretical lenses in guiding later stages of grounded theorizing (Charmaz, 2006).

In applying paradox theory, we identified and explored paradoxical tensions within LCDPs, focusing on how these tensions manifested between citizen and professional developers during interactions and decisions-making. The analysis aggregated initial codes into broader paradoxical themes, providing insights into the deeper contradictions influencing organizational behavior. Paradox theory also guided in examining the conditions under which these tensions arise and how they are managed over time. This approach provided a framework for analyzing the interplay between various stakeholders in the LCDP environment, enriching our understanding of the complexities inherent in LCD and offering valuable insights into how organizations can effectively manage and leverage these paradoxical tensions.

Identification of Tensions and Development of Strategies

After identifying the causes of tensions, we proceeded to develop strategies to manage these challenges. This was achieved through a focused analysis of the interview data, where instances of successful approaches to resolving or mitigating the identified tensions were extracted. By employing selective coding, these insights were distilled into concrete strategies. To ensure the strategies were robust and reflective of real-world practices, they were validated by cross-referencing with direct quotes from the interviews. This approach ensured that the strategies were grounded in real-world experiences, offering practical guidance for organizations navigating the complexities of LCDPs.

Through this methodological approach, we were able to generate actionable insights that organizations can utilize to manage the paradoxical tensions inherent in LCDPs, ensuring that the strategies developed are both theoretically informed and practically relevant.

Results

Overview of Paradoxes in Low Code Environments

In the following, we will give an overview of the identified paradoxes that emerge between citizen and professional developers in low code environments (see Figure 1).

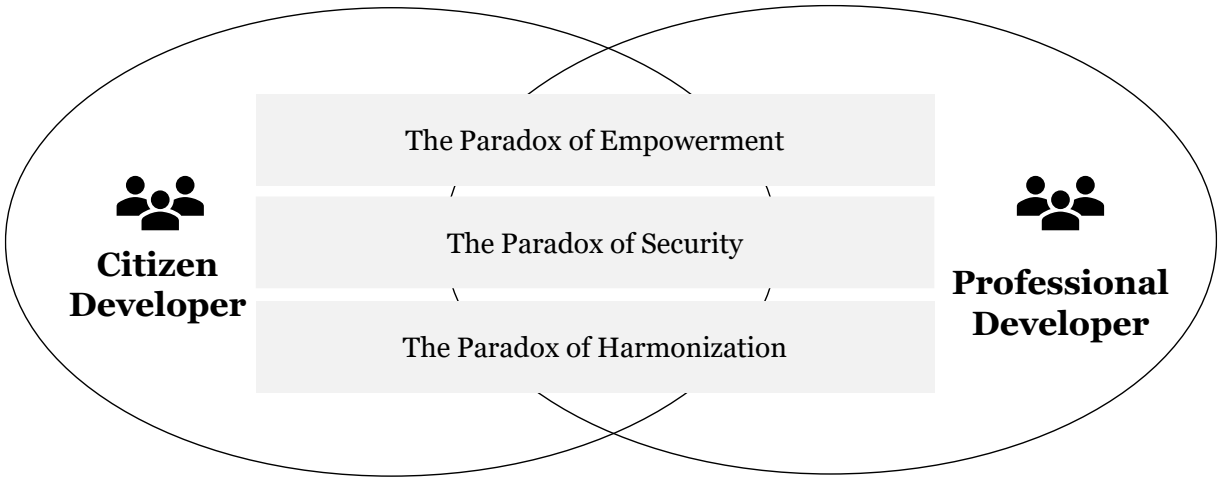

Figure 1. Overview of Identified Paradoxes in Low Code Environments

Paradox 1: The Paradox of Empowerment

The first paradox explores the tension between the **Need for Autonomy** and the **Need for Governance.** As organizations empower their non-technical staff through LCDPs, they also face the challenge of maintaining governance and control over the development process. As one project manager remarked, “LCDPs empower our non-technical staff, but it's a tightrope walk. We're constantly balancing between unleashing creativity and maintaining order” (Interviewee 1). This tension underscores the struggle to foster innovation while ensuring that the development process remains cohesive and controlled.

Paradox 2: The Paradox of Security

This paradox highlights the conflict between the **Need for Democratization** and the **Need for Protection**. While democratizing access to data and development tools can drive innovation and inclusivity, it also raises concerns about maintaining security and consistency across the organization. A citizen developer shared, “LCDPs have made data more approachable, but I’m always wary of the thin line between accessibility and exposure” (Interviewee 20). This paradox emphasizes the challenge of opening data while ensuring robust security measures are in place to protect sensitive information.

Paradox 3: The Paradox of Harmonization

This paradox examines the tension between the **Need for Individualization** and the **Need for Integration**. As organizations strive to offer personalized, tailored solutions through their development platforms, they must also balance this with the need for integrated, standardized processes that ensure efficiency and scalability. "On one hand, LCDPs offer a canvas for suggested solutions. On the other, we face the challenge of maintaining a standardized approach," explained a professional developer (Interviewee 12). This paradox captures the ongoing struggle to balance the benefits of customization, with the need for a unified, integrated development process.

These paradoxes extend beyond theoretical concepts; they materialize as tangible challenges and opportunities within the LCDP environment. For instance, the tension within **The Paradox of Empowerment** between the **Need for Autonomy** and the **Need for Governance** can lead to groundbreaking innovations, yet also poses the risk of creating a fragmented and uncontrolled landscape if not carefully balanced. Similarly, **The Paradox of Security**, highlighting the conflict between the **Need for Democratization** and the **Need for Protection**, underscores the necessity for robust governance models that make data and development tools widely accessible while ensuring consistency and security across the organization. **The Paradox of Harmonization** reflects the struggle between the **Need for Individualization** and the **Need for Integration**, emphasizing the importance of crafting personalized solutions that seamlessly align with overarching systems and processes. Navigating these paradoxes requires a nuanced understanding and a strategic approach, rather than viewing these tensions as problems to be solved, it is essential to recognize them as dynamic equilibriums to be effectively managed. Delving into the complexities of LCDPs reveals that smoothly navigating these complex and interconnected paradoxes is key to unlocking their full potential.

Causes of Tensions for Paradoxes

Each paradox is a result from tensions that are emerging due to the different needs of citizen developers and professional developers in low code environments. In the following, we will describe the different needs of the actors involved and the resulting paradoxes.

Paradox 1: The Paradox of Empowerment

The first paradox is about the balance between the **Need for Autonomy** and the **Need for Governance** within LCDPs.

The Need for Autonomy is the basis of citizen developer empowerment, as these platforms grant individuals the autonomy to innovate and engage in self-service development. This newfound autonomy fosters a culture of creativity, allowing creative ideas to flourish. However, this autonomy comes with a potential downside: it may usher in a loss of control for IT departments. A citizen developer reflected on this paradox, stating: “At first, it was liberating to build what we needed, but soon we realized our creations were out of sync with IT standards” (Interviewee 10). This tension arises when the innovative drive of citizen developers operates independently of the structured control mechanisms maintained by IT departments.

The Need for Governance is the second mechanism that fuels the paradox of empowerment. Governance plays a critical role in this dynamic by seeking to establish a balance between the freedom to innovate and the imperative to maintain control. An IT manager shared a proactive approach: “We implemented a governance board that includes both citizen and professional developers. It's about finding that sweet spot where innovation thrives within controlled boundaries” (Interviewee 18). Thus, governance emerges as a crucial tool in maintaining a harmonious equilibrium between the need for control and the drive for empowerment within the LCDP landscape.

Table 4 represents an illustrative example of the used data to identify the described causes of tension. In the quotes, we have symbolically captured statements from citizen and professional developers to characterize their perspectives and illustrate how these causes of tension arise among the actors in the low code environment.

Causes of Tension	Illustrative Data
Need for Autonomy	*"The freedom to develop solutions independently is empowering, but it's a double-edged sword." (Interviewee 13)*
	"Having the autonomy to create apps is great, but there are moments when I'm unsure if I'm following the best practices." (Interviewee 19)
Need for Governance	*"Our IT department sets clear guidelines for using the platform, which helps in maintaining a balance between freedom and control " (Interviewee 26)*
	"There's a governance framework in place that ensures we don't compromise on security while being innovative. " (Interviewee 10)

Table 4. Paradox 1: Causes of Tension and Illustrative Data.

Paradox 2: The Paradox of Security

The second paradox revolves around the tension between the **Need for Democratization** and the **Need for Protection**, especially within the realm of data security.

The Need for Democratization is pivotal in this paradox, as it involves expanding access to data and development tools through LCDPs, allowing non-technical users to participate in development activities. While this democratization enhances agility and fosters inclusivity, it also introduces a significant security challenge. As a security analyst stated, “The more people have access to sensitive data, the higher the risk of a breach. It's a constant battle between accessibility and security" (Interviewee 18). This tension often manifests in heightened security incidents and potential data breaches, illustrating the ongoing struggle to balance accessibility with security.

The Need for Protection emerges as a second mechanism fueling this paradox, particularly in terms of ensuring compliance within this democratized environment. As more individuals gain access to development tools, the complexity of maintaining consistent security and compliance standards increases. A compliance officer highlighted this challenge: “Every app built by our citizen developers must comply with GDPR, and that's a huge undertaking” (Interviewee 7). The dilemma lies in integrating compliance seamlessly into the fabric of LCDP usage without stifling innovation. This mechanism captures the ongoing struggle to navigate the landscape of data democratization and security, underscoring the need for a nuanced approach that maintains harmony between accessibility and stringent measures.

Causes of Tension	Illustrative Data
Need for Democratization	*"The ease with which our team members can access and manipulate data is revolutionary, but it does raise concerns about who has access to what." (Interviewee 14)*
	"LCDPs have made it easier for everyone to get involved in development, but this also means we have to be extra vigilant about data security. " (Interviewee 18)
Need for Protection	*“Ensuring that every application developed meets our compliance standards is a massive task, especially with the increasing number of citizen developers." (Interviewee 27)*
	"We've had to implement rigorous compliance checks for every project to ensure we're not violating any data protection laws. " (Interviewee 28)

Table 5. Paradox 2: Causes of Tension and Illustrative Data.

Table 5 provides representative insights into the data used, illustrating the empirical foundation that support the identified causes of tension within the paradox of security, especially the tensions between the Need for Democratization and the Need for Protection.

Paradox 3: The Paradox of Harmonization

The third paradox centers around the tension between the **Need for Individualization** and the **Need for Integration** within LCDPs. This paradox is driven by challenge of balancing these two conflicting needs of the actors in low code environments.

The Need for Individualization highlights the appeal of LCDPs in their ability to tailor solutions to meet specific departmental needs. This flexibility allows departments to create highly customized applications that address their unique requirements. However, this adaptability can lead to a significant downside: it often results in the fragmentation of IT systems. An IT architect expressed a concern, stating, "We've seen instances where every department had its own app for the same purpose. It's a nightmare for standardization" (Interviewee 12). This issue typically arises when there is a lack of centralized oversight or a unified IT strategy, causing a fragmented and inconsistent technology landscape.

The Need for Integration focuses on the necessity of incorporating these individualized solutions into the existing IT infrastructure. A systems integrator explained this complexity: "Each custom solution is like a puzzle piece; it needs to fit perfectly into our existing IT landscape" (Interviewee 8). The integration process often reveals the tension between allowing for individualized solutions and the requirement for a cohesive, standardized IT framework. This tension frequently leads to increased costs and delays as organizations struggle to harmonize diverse custom solutions with a unified IT framework.

Table 6 presents a representative example of the data used, offering insights into the empirical foundation that supports the existence of these causes of tension in the LCD landscape.

Causes of Tension	Illustrative Data
Need for Individualization	*" The agility LCDPs offer is fantastic for meeting unique departmental needs, (...). " (Interviewee 15)*
	" LCDPs allow for rapid customization, but this often leads to a lack of coherence in our overall IT strategy." (Interviewee 21)
Need for Integration	*"Integrating these custom apps into our main systems can be a logistical headache. Each one is built differently. " (Interviewee 8)*
	"We face significant challenges in ensuring that each department's custom solutions work seamlessly with our existing infrastructure " (Interviewee 12)

Table 6. Paradox 3: Causes of Tension and Illustrative Data.

These mechanisms highlight the complex nature of the paradoxes in LCD landscape. They emphasize the importance of a balanced approach that recognizes and addresses the varied needs and perspectives of all stakeholders involved. By understanding these underlying causes of tension, organizations can more effectively navigate the paradoxical landscape of LCDPs, leveraging their potential while minimizing the associated risks.

Strategies for Navigating the Paradoxes

Effectively managing the identified paradoxes of LCD requires a strategic approach. The following strategies are tailored to each paradox and provide a path to effective management.

The Paradox of Empowerment

To manage the tension between the Need for Autonomy, a strategy of **collaborative governance** is essential. This involves establishing a governance framework that includes both professional and citizen developers, promoting joint decision-making through cross-functional teams or committees. As one project manager explained, "We set up a cross-functional team that reviews and approves all major projects. This way, everyone feels involved and responsible" (Interviewee 1).

In addition to this approach, **clear guidelines** are crucial for maintaining a harmonious development environment within LCDPs. Transparent standards, co-created by both professional and citizen developers,

hep ensure control without stifling creativity. A senior developer noted, "Having a set of agreed-upon standards helps us maintain control without stifling creativity" (Interviewee 12).

The Paradox of Security

Balancing the Need for Democratization and the Need for Protection in terms of security requires a multifaceted approach. One effective approach is **layered security**, which incorporates user roles and permissions within the LCDP, allowing different levels of data access. A security officer explained, "We use a tiered access system, ensuring that sensitive data is only accessible to those who really need it" (Interviewee 5).

Complementing this, a **continuous education** strategy is essential. Regular workshops, e-learning modules, and communications to help educate users on data security best practices and compliance requirements. A compliance trainer pointed out: "We conduct monthly security workshops. It's crucial that everyone understands the importance of data security" (Interviewee 7).

The Paradox of Harmonization

Managing the tension between the Need for Individualization and the Need for Integration involves adopting a **modular design approach**. This strategy encourages the development of custom solutions using standardized, reusable components. An IT architect highlighted this approach: "We promote building blocks that can be reused across different applications. It helps maintain standardization while allowing customization" (Interviewee 12). This modular design approach supports standardization while leave room for a high degree of customization, providing a dynamic framework that aligns with the organization's evolving needs.

Ensuring this strategic approach, a **centralized repository** for all custom developments serves as a unified and accessible hub for developers, facilitating seamless reference and reuse of custom modules and applications. A senior developer highlighted the significance of this centralized library, stating, "Having a central repository of custom modules and apps helps us avoid reinventing the wheel" (Interviewee 23). By creating a shared space for knowledge storage and retrieval, organizations strengthen their collaborative potential, fostering a culture of efficiency and innovation.

To create a dynamic and responsive work environment, organizations must also implement overarching strategies that support the success and effectiveness of these initiatives. One key strategy is **open communication**, aiming at fostering an environment where concerns and ideas can be freely exchanged. A Chief Digital Officer (Interviewee 21) emphasized the effectiveness of regular town hall meetings as a way to stay connected and address issues early on. This approach not only promotes transparency but also ensures a collective understanding of diverse perspectives, creating an environment where common ground can be achieved.

In addition to open communication, adopting an **agile adaption strategy** is crucial for the successful implementation and sustainability of LCDP initiatives. By embracing an agile framework, organizations can smoothly adjust to changing needs and challenges. The Head of Consulting of a large IT service company (Interviewee 16) recommended working in sprints and continuously refining strategies based on feedback and emerging developments. This agile mindset allows organizations to maintain flexibility, respond promptly to evolving requirements, and ensure the LCDP initiatives remain aligned with the fast-paced landscape of technological advancements.

By implementing the suggested strategies, organizations can effectively navigate the paradoxes inherent in LCDPs. This approach not only mitigates risks but also maximizes the benefits of LCD, fostering an environment of innovation, collaboration, and security.

Figure 2 gives an overview of the causes of tension between citizen and professional developers in low code environments, the resulting paradoxes, and suggested strategies for managing the paradoxes.

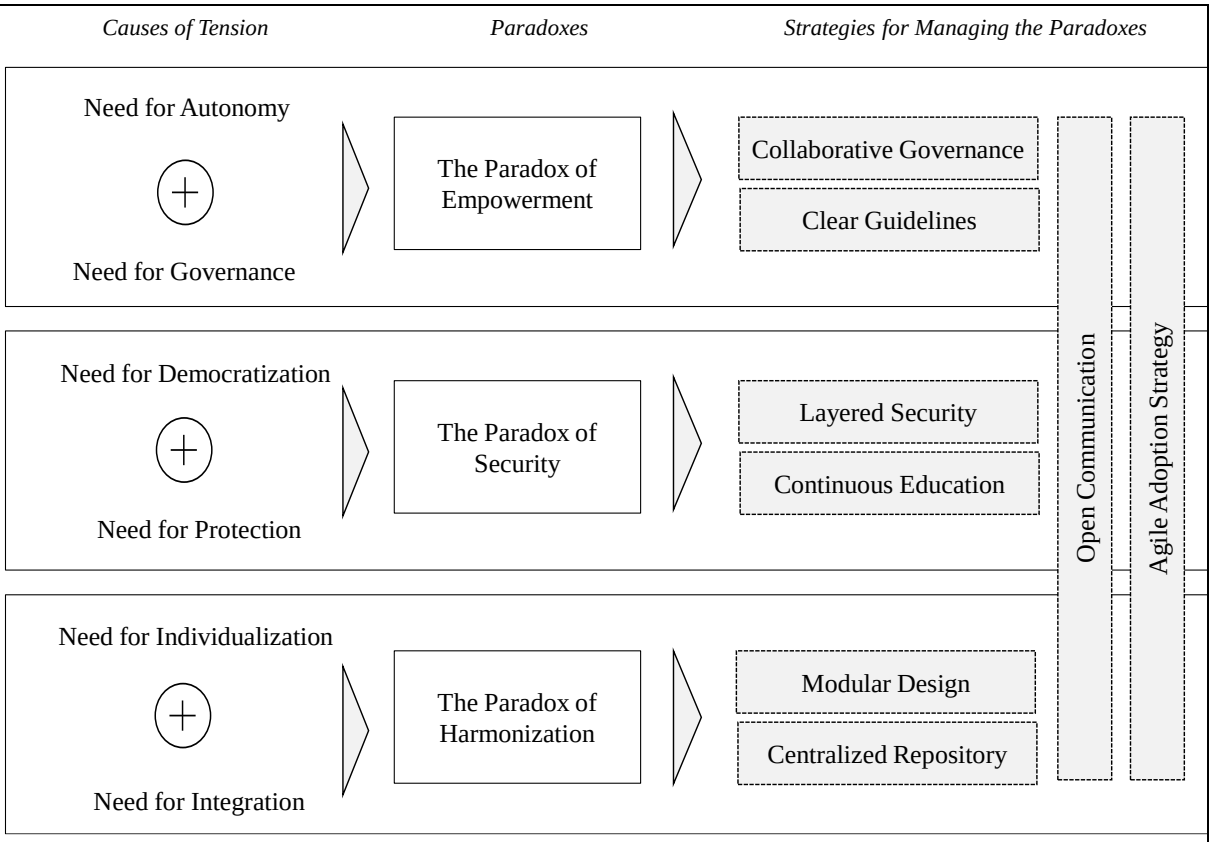

Figure 2. Overview of Causes of Tension, Resulting Paradoxes and Suggested Strategies for Managing the Paradoxes

Discussion

This study contributes to the body of knowledge in information systems development (ISD) by exploring the tensions between professional and citizen developers within the context of LCDPs, highlighting the challenges and opportunities that can arise from their interactions. In this section, we will discuss the contribution of our research to the body of knowledge in ISD, the implications of our findings, and acknowledge the limitations of our research.

Our research examines the interactions between citizen and professional developers, offering insights into the complex and often paradoxical nature of their collaboration. By focusing on the interplay between these groups, our study reveals details of working together in environments where the boundaries between professional expertise and citizen innovation are blurred. Our empirical findings suggest that while these tensions can be managed, they are not easily resolved in a traditional sense. Paradox theory provides a suitable framework for understanding how organizations can navigate these tensions constructively. Instead of viewing these tensions as conflicts to be resolved, organizations can embrace them as dynamic forces that, when managed effectively, can lead to continuous improvement and innovation (Smith & Lewis, 2011).

While some elements of the identified tensions, such as role identity tensions (Mueller et al., 2024), tensions related to project structure and project goals (Iivari, 2021), different roles and control mechanisms (Tiwana, 2010) and the differing requirements of involved stakeholders (Benbya & KcKelvey, 2006) have been identified in previous ISD literature, our study contributes to the contextualization of these established tensions within the specific setting of LCDPs, where the role of citizen developers introduce new dimensions of these existing conflicts. Additionally, we provide a framework to strategically identify and manage them. This reframing allows organizations to view these tensions not as isolated conflicts but as interconnected

challenges that need to be managed holistically. Thus, our study contributes to the existing literature on ISD by providing a deeper understanding of the specific tensions in low code environments and offering a framework that helps organizations strategically identify and manage these tensions.

From a research perspective, our study contributes to the current ISD literature by offering a comprehensive framework for future research on this topic by identifying and analyzing the various dimensions of paradoxes in LCDPs. This framework can be used to guide future research into the psychological, emotional, and organizational aspects of collaboration between citizen developers and professional developers in LCDPs, as well as their impact on productivity, innovation, and organizational culture. In addition to that, we provide recommendations for managing these paradoxical relationships. Our study calls for further research into the evolving landscape of LCDPs and the growing role of citizen developers in shaping organizational technological landscapes. In doing so, this work aims to provide valuable insights for both academics and practitioners navigating the in an era of increasing citizen participation in the development process.

In terms of practical implications, our findings are significant for organizations implementing LCDPs, as they offer insights into the potential risks and challenges associated with the tensions between professional and citizen developers. By understanding and addressing these challenges, organizations can maximize the benefits of LCDPs while minimizing the risks associated with the paradoxes identified in this study. The recommendations derived from our findings can serve as guidelines for organizations seeking to establish best practices for managing the paradoxical relationships between these two groups. Key recommendations include fostering a collaborative environment, establishing clear boundaries and responsibilities, providing ongoing training and support, promoting open communication, and balancing control with autonomy, standardization with customization. Furthermore, our findings highlight the importance of adopting a flexible and adaptable approach to managing these paradoxical relationships. In such a dynamic organizational system, leadership's role is not to eliminate tensions but to harness them, enabling the system to thrive through continuous improvement (Nonaka & Toyama, 2002; Teece & Pisano, 1994; Weick & Quinn, 1999). By following these recommendations, organizations can create a more inclusive and effective software development process that meets the needs of both professional and citizen developers, recognizing the necessity of balance and negotiation in addressing the various paradoxes and tensions that can arise.

Despite the valuable insights gained from our research, there are some limitations to be aware of. First, our study may not have covered all the possible contexts and scenarios in which LCDPs are used, limiting the generalizability of our findings. Additionally, the study primarily focuses on the paradoxes and tensions that arise from interactions between citizen and professional developers in low code environments; future research could explore other factors influencing LCDP success, such as organizational culture and leadership styles. Finally, to enhance the robustness of future studies, it would be beneficial to complement self-reported data with other sources, such as project documentation or performance metrics, to validate the findings and provide a more comprehensive understanding of these tensions, potentially within a paradoxical framework. Additionally, our study relies on self-reported data from professional and citizen developers, which may be subject to biases such as social desirability and recall bias. Using self-reported data alongside with other sources, such as project documentation or performance metrics, could help validate the findings and provide a more comprehensive understanding of paradoxical relationships in LCDPs.

Conclusion and Future Research

This paper has highlighted the complex and paradoxical relationships between citizen and professional developers within LCDPs. By identifying and addressing these tensions, organizations can effectively leverage LCDPs to drive innovation, enhance agility, and increase efficiency while minimizing the risks associated with these paradoxical dynamics. The insights gained from research not only contribute to the existing body of knowledge in ISD but also provide actionable guidance for organizations looking to implement LCDPs successfully. Several potential research directions emerge from our findings.

The recent integration of Generative Artificial Intelligence (GenAI) in LCDPs (Bruhin et al., 2024; Eun & Yong, 2024; Binzer & Winkler, 2024), which has been shown to significantly enhance productivity (Dell'Acqua et al., 2023; Sadowski & Zimmermann, 2019; Li et al., 2024), foster team collaboration (Ziegler

et al., 2022; Meyer et al., 2014), and improve code quality (Bouschery et al., 2023), could substantially change the dynamics within development teams. Future studies should contextualize these paradoxes within the emerging trends of GenAI integration, exploring how these tools might further complicate or relieve the tensions between citizen and professional developers.

Another promising direction for future research is to investigate the long-term effects of these paradoxical relationships on organizational innovation and performance. Specifically, understanding how varying levels of tension between citizen and professional developers impact the success of software development projects, user satisfaction, and overall organizational adaptability could provide deeper insights into the sustainable implementation of LCDPs.

Additionally, future studies should explore the contextual factors—such as industry-specific challenges and organizational structures—that influence the formation and management of these paradoxical relationships. Such research could help identify best practices and potential pitfalls when implementing LCDPs across diverse settings.

Finally, future research should focus on developing tools, frameworks, and methodologies that help organizations effectively manage these paradoxes. This might involve examining the roles of leadership, organizational culture, and cross-functional collaboration in fostering effective communication and decision-making between citizen and professional developers. The integration of GenAI tools, as discussed by Bruhin et al. (2024) and Eun & Yong (2024), adds an additional layer of complexity, necessitating new approaches to governance and training programs that enhance collaboration and understanding between these groups. Ensuring that the competing needs and priorities of both citizen and professional developers are balanced effectively is crucial for the successful adoption of LCDPs.

References

Adrian, B., Hinrichsen, S., & Nikolenko, A. (2020). App development via low code programming as part of modern industrial engineering education. In Advances in Human Factors and Systems Interaction: Proceedings of the AHFE 2020 Virtual Conference on Human Factors and Systems Interaction, July 16-20, 2020, USA, 45-51. *Springer International Publishing*.

Alsaadi, H., Radain, D., Alzahrani, M., Alshammari, W., Alahmadi, D., & Fakieh, B. (2021). Factors that affect the utilization of low-code development platforms: Survey study. Romanian Journal of Information *Technology and Automatic Control*, 31(3), 123–140.

Andriopoulos, C., & Lewis, M. W. (2009). Exploitation-exploration tensions and organizational ambidexterity: Managing paradoxes of innovation. *Organization Science*, 20, 696-717.

ATLAS.ti. (2024). ATLAS.ti. Retrieved from https://atlasti.com

Benbya, H. and McKelvey, B. (2006). Toward a complexity theory of information systems development, In *Information Technology & People*, (19, 1), 12-34. https://doi.org/10.1108/09593840610649952

Binzer, B., & Winkler, T. J. (2022). Democratizing software development: A systematic multivocal literature review and research agenda on citizen development. In N. Carroll, A. Nguyen-Duc, X. Wang, & V. Stray (Eds.), Software Business (Vol. 463, 244–259). *Springer International Publishing*.

Binzer, B., & Winkler, T. J. (2024). Die vier Phasen von Citizen Development-Initiativen: Treiber, Herausforderungen und Handlungsempfehlungen. *HMD Praxis der Wirtschaftsinformatik,* 1-23.

Bock, A. C., & Frank, U. (2021). Low-code platform. In *Business & Information Systems Engineering*, 63(6), 733–740.

Bouschery, S.G., Blazevic, V. and Piller, F.T. (2023), Augmenting human innovation teams with artificial intelligence: Exploring transformer-based language models. In *Journal of Product Innovation Management*, 40(2), 139–153

Bruhin O, Dickhaut E, Elshan E, Li M (2024). The rise of generative AI in low code development platforms—an analysis and future directions. In *Proceedings of the 57th Hawaii International Conference on System Sciences (HICSS)*

Cameron, K. S. (1986). Effectiveness as paradox: Consensus and conflict in conceptions of organizational effectiveness. *Management science*, 32(5), 539-553.

Cameron, K., & Quinn, R. (1988). Organizational paradox and transformation. In R. Quinn & K. Cameron (Eds.), Paradox and transformation: Toward a theory of change in organization and management, 1-18). *Cambridge*, MA: Ballinger.

Carroll, N., & Maher, M. (2023). How shell fueled a digital transformation by establishing DIY software development. *MIS Quarterly* Executive, 22(2), 3.

Charmaz, K. (2006). Constructing grounded theory: A practical guide through qualitative analysis. Sage.

Dell'Acqua, F., McFowland, E., Mollick, E. R., Lifshitz-Assaf, H., Kellogg, K., Rajendran, S., ... & Lakhani, K. R. (2023). Navigating the jagged technological frontier: Field experimental evidence of the effects of AI on knowledge worker productivity and quality. In *Harvard Business School Technology & Operations Mgt. Unit Working Paper*, 24-013.

Eun, H. J., & Yong, G. G. (2024). A Study on intent to use AI-enhanced development tools. In *Convergence Security Journal*, 24(2), 89-104.

Forman, J., & Damschroder, L. (2007). Qualitative content analysis. In L. Jacoby & L. A. Siminoff (Eds.), Empirical methods for bioethics: A primer, 11, 39–62. *Bingley: Emerald.*

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), Article 1.

Hoogsteen, D., & Borgman, H. (2022). Empower the workforce, empower the company? Citizen development adoption. *Proceedings of the 55th Hawaii International Conference on System Sciences.*

Hsu, S.-H., Wang, Y.-C., & Tzeng, S.-F. (2007). The source of innovation: Boundary spanner. *Total Quality Management & Business Excellence*, 18(10), Article 10.

Iho, S., Krejci, D., & Missonier, S. (2021). Supporting knowledge integration with low-code development platforms. ECIS 2021 Research Papers.

Iivari, J. (2021). A paradox lens to systems development projects: The case of the agile software development. *Communications of the Association for Information Systems*, 49(1), 4.

Kletti, N. L. (2021). Trends der Fertigungs-IT 2021. *Zeitschrift Für Wirtschaftlichen Fabrikbetrieb*, 116(5). 276–278.

Krejci, Désirée; Iho, Satu; and Missonier, Stephanie, "Innovating with employees: an exploratory study of idea development on low-code development platforms" (2021). *Proceedings of the Twenty-Ninth European Conference on Information Systems*.

Langley, A. (1999). Strategies for theorizing from process data. *Academy of Management Review*, 24(4), 710.

Lebens, M., Finnegan, R. J., Sorsen, S. C., & Shah, J. (2021). Rise of the citizen developer. *Muma Business Review*, 5(12), 101-111.

Lethbridge, T. C. (2021). Low-code is often high-code, so we must design low-code platforms to enable proper software engineering. In T. Margaria & B. Steffen (Eds.), Leveraging Applications of Formal Methods, Verification and Validation, 13036, 202–212. *Springer International Publishing*.

Lewis, M. W. (2000). Exploring paradox: Toward a more comprehensive guide. *The Academy of Management Review*, 25(4), 760. https://doi.org/10.2307/259204

Li, M. M., Dickhaut, E., Bruhin, O., Wache, H., and Weritz, P. (2024). More Than Just Efficiency: Impact of Generative AI on Developer Productivity. *AMCIS 2024 Proceedings*.

Locke, K., & Golden-Biddle, K. (1997). Constructing opportunities for contribution: Structuring intertextual coherence and "problematizing" in organizational studies. *Academy of Management Journal*, 40(5), 1023–1062.

Luo, Y., Liang, P., Wang, C., Shahin, M., & Zhan, J. (2021). Characteristics and challenges of low-code development: The practitioners' perspective. *Proceedings of the 15th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM),* 1–11.

Marabelli, M., & Newell, S. (2012). Knowledge risks in organizational networks: The practice perspective. *The Journal of Strategic Information Systems*, 21(1), 18-30.

Merriam, S. B., & Grenier, R. S. (Eds.). (2019). Qualitative research in practice: Examples for discussion and analysis. *John Wiley & Sons.*

Meyer, A.N., Fritz, T., Murphy, G.C. and Zimmermann, T. (2014), Software developers' perceptions of productivity. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 19–29.

Mueller, L., Albrecht, G., Toutaoui, J., Benlian, A., & Cram, W. A. (2024). Navigating role identity tensions—IT project managers' identity work in agile information systems development. *European Journal of Information Systems*, 1-24.

Myers, M. D., & Newman, M. (2007). The qualitative interview in IS research: Examining the craft. In *Information and organization*, 17(1), 2-26.

Nonaka, I. and Toyama, R. (2002) A Firm as a Dialectical Being: Toward a Dynamic Theory of a Firm. In *Industrial and Corporate Change*, 11, 995-1009. https://doi.org/10.1093/icc/11.5.995

Poole, M. S., & Van de Ven, A. H. (1989). Using paradox to build management and organizational theory. *Academy of Management Review*, 14, 562-578.

Rafi, S., Akbar, M., Sánchez-Gordón, M., & Colomo-Palacios, R. (2022). DevOps practitioners' perceptions of the low-code trend. *Proceedings of the 16th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, 301-306.

Richardson, C., Rymer, J. R., Mines, C., Cullen, A., & Whittaker, D. (2014). New development platforms emerge for customer-facing applications. *Forrester Research*.

Robey, D., Ross, J. W., & Boudreau, M. C. (2002). Learning to implement enterprise systems: An exploratory study of the dialectics of change. *Journal of Management Information Systems*, 19(1), 17-46.

Rokis, K., & Kirikova, M. (2022). Challenges of low-code/no-code software development: A literature review. In Ē. Nazaruka, K. Sandkuhl, & U. Seigerroth (Eds.), Perspectives in Business Informatics Research, 462, 3–17. *Springer International Publishing*.

Sadowski, C. and Zimmermann, T. (Eds.) (2019), Rethinking Productivity in Software Engineering, *Springer Nature*, Berkeley, CA.

Sahay, A., Indamutsa, A., Di Ruscio, D., & Pierantonio, A. (2020). Supporting the understanding and comparison of low-code development platforms. *46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 171-178.

Saldaña, J. (2021). The coding manual for qualitative researchers. Thousand Oaks, CA: *Sage*.

Schad, J., & Bansal, P. (2018). Seeing the forest and the trees: How a systems perspective informs paradox research. *Journal of Management Studies*, 55(8), 1490–1506.

Schad, J., Lewis, M. W., Raisch, S., & Smith, W. K. (2016). Paradox research in management science: Looking back to move forward. *Academy of Management Annals*, 10(1), 5-64.

Smith, W. K., & Lewis, M. W. (2011). Toward a theory of paradox: A dynamic equilibrium model of organizing. *Academy of Management Review*, 36(2), 381-403.

Statista. (2022). Organizations experiencing skills shortage worldwide 2015-2022. Retrieved from https://www.statista.com/statistics/1269776/worldwide-organizations-talent-shortage-skills-tech/

Strauss, A. L., & Corbin, J. M. (1998). Basics of qualitative research: Techniques and procedures for developing grounded theory (2nd ed.). *Sage Publications*.

Teece, D. & Pisano, G. P. (1994). The dynamic capabilities of firms: An introduction. In *Industrial and Corporate Change*. 3 (3), 537-556. doi: 10.1093/icc/3.3.537-a

Tisi, M., Mottu, J.-M., Kolovos, D. S., de Lara, J., Guerra, E., Ruscio, D. D., Pierantonio, A., & Wimmer, M. (2021). Lowcomote: Training the next generation of experts in scalable low-code engineering platforms.

Tiwana, A. (2010) Systems Development Ambidexterity: Explaining the Complementary and Substitutive Roles of Formal and Informal Controls, In *Journal of Management Information Systems*, 27:2, 87-126, DOI: 10.2753/MIS0742-1222270203

Ullrich, C., Lata, T., & Geyer-Klingeberg, J. (2021). Celonis Studio-A Low-Code Development Platform for Citizen Developers. In *BPM (PhD/Demos)*, 102-105.

Wang, Y., Feng, Y., Zhang, M., & Sun, P. (2021). The necessity of low-code engineering for industrial software development: A case study and reflections. In *2021 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, 415–420.

Waszkowski, R. (2019). Low-code platform for automating business processes in manufacturing. *IFAC-PapersOnLine*, 52(10), 376–381.

Weick, K. E., & Quinn, R. E. (1999). Organizational change and development. In *Annual Review of Psychology*. 50, 361–386. https://doi.org/10.1146/annurev.psych.50.1.361

Ziegler, A., Kalliamvakou, E., Li, X.A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G. and Aftandilian, E. (2022). Productivity assessment of neural code completion. In *Proceedings of the 6th ACM SIGPLAN International Symposium*, 21–29