# A Reflection on the Use of Systemic Thinking in Software Development

Paolo Ciancarini[1][a], Mirko Farina[2][b], Artem Kruglov[2][c], Giancarlo Succi[1][d] and Ananga Thapaliya[2][e]

[1]*University of Bologna, Bologna, Italy*

[2]*Innopolis University, Universitetskaya st., 1, Innopolis, Russia*

Keywords: Systemic Thinking, Software Engineering, External Factors.

Abstract: The research examines the value and potential usefulness of using systemic thinking, which looks at the interconnectedness of things, to comprehend the complexities of software development projects and the technical and human factors involved. It considers two different aspects of systemic thinking - psychological and sociological - and posits that these can assist in understanding how software teams function and attain their objectives, as well as the goals of the entities for which they work. Our research aims to provide a novel contribution to the field by investigating the use of systemic thinking in software development teams and organizations. We evaluate the reliability and validity of the survey applied to different groups of relevant participants, relate our findings to existing literature, and identify the most representative factors of systemic thinking. Despite the popularity of various factors that fall under the umbrella of 'systems thinking', there is limited understanding of their effectiveness in improving organizational performance or productivity, particularly when it comes to psychological and sociological systemic factors. The relationship between the use of systems thinking and organizational performance is often based on anecdotal evidence, rather than the identification and application of specific factors. Our work emphasizes the importance of understanding and applying such factors in order to build a solid foundation for the effective use of system dynamics and systems thinking tools, which is crucial for software development teams.



## 1 INTRODUCTION

Systemic thinking is a field of research that takes a holistic approach to understanding complex structures, whether physical, virtual, or numerical (Sheridan, 2010). It aims to study systems from a "system of systems" perspective, meaning it looks at how different systems interact and depend on one another (Gallón, 2020). This interdisciplinary field draws on various disciplines including engineering, computer science, cognitive science, management, philosophy, psychology, and biology (Lipson, 2007).

In this paper, we use systemic thinking as a way to understand and improve the design and management of complex systems and their life cycles (Edson, 2008; Boardman and Sauser, 2008). We view teams as systems made up of individuals with different skills working together towards a common goal or plan (Katzenbach and Smith, 2008) and apply this understanding to software development teams specifically. We believe that using this approach can provide valuable insights into how to improve teamwork in software engineering.

While individual contributions are important in software development, it is also important to recognize that teams play a crucial role in the IT industry. Teams can be a key factor in a company's success and are often necessary to meet the demands of a competitive industry. In order to understand how software teams function and the dynamics within them, it is important to use systemic thinking as a tool for insight and understanding (Gogichaty et al., 2023; Zahra and Bogner, 2000). This is why we have chosen to use systemic thinking as a framework in this paper to explore the relationships and dynamics within software development teams.

The study aims to test the hypothesis by addressing the following sub-questions:

1. What elements of systemic thinking can be found in the actions and interactions of individuals,

[a] https://orcid.org/0000-0002-7958-9924
[b] https://orcid.org/0000-0001-8342-6549
[c] https://orcid.org/0000-0003-2038-1392
[d] https://orcid.org/0000-0001-8847-0186
[e] https://orcid.org/0000-0002-4526-1090



Ciancarini, P., Farina, M., Kruglov, A., Succi, G. and Thapaliya, A.
A Reflection on the Use of Systemic Thinking in Software Development.
DOI: 10.5220/0011963300003464
In *Proceedings of the 18th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2023)*, pages 521-529
ISBN: 978-989-758-647-7; ISSN: 2184-4895

teams, and organizations involved in software engineering?

2. Are those in the software engineering field aware of these elements?

Section 2 argues that a wide range of factors, broadly classified as psychological and sociological, play a crucial role in software development and goes on to examine these factors in depth. In an effort to confirm the hypothesis that systemic thinking can aid in understanding how software teams work and achieve their goals, Section 3 explains the research methodology and experimental design used in the study. Section 4 presents a critical analysis of the qualitative data collected and Section 5 summarizes the findings and their significance to the software engineering field. Finally, Section 6 summarizes the study's key takeaways and suggests potential areas for future research.

# 2 SYSTEMIC THINKING FACTORS IN SOFTWARE DEVELOPMENT

The success or failure of software teams is not solely dependent on the technology being used, the skills of team members, or the internal dynamics of a specific project. External factors, such as organizational, psychological, and sociological, can have a significant impact on the performance of software teams. These external factors are often instrumental or constitutive in the functioning of software companies, and in the practice of software engineering and development overall. Therefore, it is important to consider and understand these external factors when analyzing software teams.

Many researchers (Capretz, 2014; Feldt et al., 2008) in the field of software development have argued that psychological and sociological factors have a direct impact on the production of software programs. They have pointed out that external factors such as inspiration (Beecham et al., 2008; Sharp et al., 2009; Hall et al., 2009), character (Cruz et al., 2011), and the environment (Leidner and Kayworth, 2006) in which software is created play a fundamental role in the process. This is an important consideration as it highlights that when discussing the creation of software, it's necessary to take into account not just the technical aspects but also the context and environment in which it is created. This goes beyond what is often referred to as "soft skills," (Ahmed et al., 2012; Matturro, 2013; Matturro et al., 2015) which are often reduced to generic statements about someone's personality. The creation of software is a complex undertaking that requires distributed intelligence across team members (Nerur, 2014), and a deeper understanding of psychology and sociology may provide unique insights into the dynamics of these teams.

There is a risk that software engineers may not fully understand the key factors that contribute to the success or failure of software projects, by focusing too much on formalized procedures and techniques of change control and neglecting the role of individuals and their relationship with the environment (Oreg, 2003). For example, in software development, resistance to change, which is often influenced by environmental factors, maybe a more important factor to consider than formalized change control processes. However, most research in software development has focused on formalized procedures and techniques, rather than the role of individuals and their relationship with the environment (Unterkalmsteiner et al., 2011).

In this section, we will argue that using two branches of systemic thinking, namely psychological systemic thinking and systemic sociological thinking, can help us gain important insights into the nature, scope, and dynamics of collaborative partnerships in software development.

Several authors (Hazzan and Hadar, 2008; Trendowicz and Münch, 2009; Amrit et al., 2014) have demonstrated that software engineering is a human-centered activity that is greatly impacted by people's emotions, memories, and behaviors. Software production is greatly influenced by cognitive factors and external systemic processes (Feldt et al., 2010; Khan et al., 2011; Lenberg et al., 2014). In software firms, software design, especially when using Agile methodologies, must take into account not only the project's objectives but also the interactions among team members in delivering the end product. Therefore, psychological and sociological factors that underlie people's behaviors are extremely important for software developers. Systemic thinking is a fundamental tool for understanding these factors. In this section, we will examine two branches of systemic thinking, psychological and sociological, that are relevant to our work.

## 2.1 Psychological Systemic Thinking

Psychological systemic thinking is a branch of psychology that deals with practices and actions in the environment (Schultz and Schultz, 1998). It has several different applications in psychology (Curtis, 1984), including forensic psychology (Tornhill, 2015) where it is used to predict the future of the codebase, evaluate the refactoring direction, and under-

stand how a team may influence a design. Other applications include studying the effect of moods on developers' debugging performance (Khan et al., 2011) and exploring organizational and technological issues in software engineering as a practice (Lytras et al., 2010).

Software developers are constantly exposed to new technology and techniques, and their profession requires collaboration and cooperation with a variety of actors, stakeholders, and disciplines. Therefore, it is important to understand software engineering from the point of view of software teams within software companies. To build an effective team, it is important to consider the interests, needs, and desires of all team members. When difficulties arise, it is important to find ways to mediate and compromise. In this section, we will discuss a series of psychological factors that directly affect team performance.

Effort measurement, which includes mental and physical effort, is an important psychological factor to consider when evaluating the productivity of a software team and understanding intellectual performance (Mulder, 1986; Farina et al., 2022). This variable can be described as the amount of time (expressed in hours) spent by team members on completing their assigned tasks, which can be divided into technical time (such as design, debugging, coding, and testing) and non-technical time (such as team meetings, seminars, and consultations). Several researchers (Trendowicz et al., 2008; Dlamini et al., 2022) have shown that it is crucial to measure and specify such efforts in order to understand how software teams work. All types of efforts should be included in the total amount of working hours reported and should be correlated with product improvement, both in terms of technical and non-technical efforts.

Team Leader: The behavior of the team leader is another important psychological systemic factor that can greatly impact the performance of software development teams (Acuña et al., 2008). The team leader's behavior, as shown by (Srivastava et al., 2006) can have a major influence on team members' productivity, creating an environment that is conducive to effective communication (Sudhakar et al., 2011), learning, and dedication. Therefore, it is crucial to consider the team leader's behavior when trying to understand the performance of software teams (Srivastava et al., 2006).

Based on the above, psychological systemic thinking can help us understand some of the dynamics that affect the performance of software teams. With this understanding, we can now examine the contributions that sociological systemic thinking can make to software engineering.

## 2.2 Sociological Systemic Thinking

Sociological systemic thinking primarily deals with aspects of software development in the context of technologically and culturally scaffolded artifacts, and with a focus on communication (Stichweh, 2011). This theory emphasizes the importance of understanding the dynamics behind the production of cultural artifacts, particularly within social and collaborative institutions (Parsons, 1977). The origins of this theory can be traced back to the work of philosophers and social scientists (Offer, 2010; Alexander, 1990), but a discussion of the roots of sociological systemic thinking is beyond the scope of this paper. In this section, we will focus on identifying a number of sociological factors that can potentially affect software teams and discuss some that we believe to be the most important.

### 2.2.1 Team Size

The term "team size" has been used loosely in sociology to refer to both small groups of individuals who interact productively and larger groups of individuals who share a common identity (McGrath, 1984). A more restrictive definition of a team, used in team process studies, is a group of people formed specifically for research, without prior relations among its members. According to sociological systemic thinking, an effective and productive team should have a balanced size, not too big and not too small (Cooley, 1992). A small group may not have the variety of skills it needs to operate well, while large teams may lead to decreased participation and dedication to the project. The optimal team size also depends on factors such as management skills, type of project, and personality of the group (Lee et al., 2003). It is generally considered that teams with 5 to 10 people are most effective, but exceptions can exist.

### 2.2.2 Team Cohesion and Collaboration

Team cohesion refers to the quality and degree of the relationships between members of a team. This relationship creates a bond among team members and enables them to work together effectively to achieve team objectives. Cohesion is essential for building productive collaborative activities (Kang et al., 2011). The productivity of a software development team (Bhardwaj and Rana, 2016) is greatly affected by the level of teamwork and cooperation within the team. Therefore, it is important for the program manager to create an environment that promotes teamwork and encourages cooperation (Kang et al., 2011) (Clincy, 2003).

#### 2.2.3 Communication

Communication is crucial for software teams (Ciancarini et al., 2021b; Ciancarini et al., 2021a; Kruglov, 2021), because, without effective communication, team members will not be able to effectively utilize the resources available to them, regardless of how good these resources are (Ciancarini et al., 2023). The importance of communication is often seen in face-to-face meetings (Bonner, 1959; Hare, 1994; Freeman, 1992; Stogdill, 1959; Locke et al., 1981), as without this type of interaction, the sharing of information will not happen.

However, the effectiveness of communication is not limited to human interactions. It is also greatly influenced by external factors such as the workplace and other variables previously discussed, such as cohesion and collaboration.

The use of various technological tools can also greatly impact communicative interactions, by enabling more flexible cycles and designs (Finholt et al., 1990; Davidow, 1992). For example, studies on remote brainstorming have shown that larger web-based brainstorming groups can generate more ideas than smaller groups (Connolly, 1997). Similarly, research suggests that computer-mediated communication can greatly reduce the strategic issues that arise in traditional social settings (Nunamaker et al., 1991; Bugayenko et al., 2022b; Bugayenko et al., 2022a).

After explaining how psychological and sociological systemic thinking can be used to understand the impact of external factors on software engineering teams, we present our experimental protocol, which is designed to test and support our hypothesis that systemic thinking can help us understand how software teams operate, achieve their goals, and meet the goals of the organizations, companies, and institutions for which they work.

## 3 METHODOLOGY

To expand on our research, we conducted an online survey with software developers in three different countries (Nepal, Russia, and Luxembourg). The purpose of the survey was to gather more objective opinions and perspectives on the topic we examined in this paper. The countries for the online survey were selected based on one of the authors' personal backgrounds, as he is from Nepal, studied in Russia, and currently resides in Luxembourg. Next, we will provide an overview of the structure of our survey.

We employed qualitative research for our initial study. This approach is typically used to understand perspectives and opinions. It provides insight into various issues and helps to generate concepts or theories for future quantitative research. The qualitative method allows for a deeper analysis of problems and can reveal new ideas and personal views in our study. This method employs various unstructured or semi-structured methods for data collection such as online surveys. It involves analyzing responses to identify issues and develop solutions. Since our research combines technological and behavioral elements and includes studying the responses and feedback from participants, a qualitative approach was deemed most appropriate.

### 3.1 The Survey

Surveys are a useful tool for collecting data on demographics and short-answer responses from a large number of people. In this study, surveys were used to gather and refine the information that was previously obtained through reviewing relevant literature.

The survey we created in this study aimed to provide answers to our hypothesis; that psychological and sociological systemic thinking can assist us in understanding the workings and operations of developer teams.

The survey consisted of a series of questions (combining multiple choice and numerical responses) that were used to gauge whether the participants agreed or disagreed with the notion that specific systemic factors can significantly impact their performance. It also aimed to identify and measure (using a Likert scale (Allen and Seaman, 2007)) the relative importance of these factors. The survey took no more than 30 minutes to complete (with an average actual time of 25 minutes).

As mentioned, the questions contained in our questionnaire for the survey have been taken and modified from (Lamb, 2009). They have been also inspired, as noted above, by discussions with experts in the field of systemic thinking.

The questionnaire for this study was designed in a way to ensure that the responses received would be valuable in addressing our research questions. This was done by following established best practices (Bentler et al., 1971; Bradburn and Sudman, 2004) within the field and by getting feedback from experts with relevant experience. The expert was the one with experience in using systemic thinking in STEM education.

The survey participants were chosen to bring a practical perspective on performance indicators in their work environment. They were able to complete the survey online for about a month and gave permis-

sion for the author to use and share their responses. 50 software developers from different tech companies in Russia, Nepal, and Luxembourg participated, most of whom provided information about their experience in the field. 12 had over 11 years of experience, 24 had 5-9 years, and 14 had less than 5 years. Additional information about their educational background was also collected, with 17 being graduate students, 16 being undergraduate students, 13 being master's students, and 4 being Ph.D. students.

The respondents were informed of the study's purpose and asked to identify themselves so researchers could relate their responses to information from the company. They were assured that their company would only receive summary results and that their privacy would be protected. Prior to the experiment, each organization involved sent a letter to the participating employee guaranteeing confidentiality. The companies also provided confidential performance ratings of the subjects, allowing for a comparison between an independent assessment of performance and the self-reported measure from the respondents.

# 4 ANALYSIS OF THE COLLECTED DATA

We used basic statistics using ordinal scales and coding procedures to analyze the responses we gathered from the questionnaires. We note here that our research is mostly observational. We also note that finding subjects willing to participate in a study like this is extremely difficult, and it is even more difficult because the emphasis of our work is on software managers and engineers involved in complex and large-scale projects.

From the survey, we analyzed the responses and allocated extracts that characterized the main pattern and divided it into categories and sub-categories as shown in Table 1.

Table 1: Identified categories and sub categories (systemic factors).

| **Categories** | **Sub Categories** (systemic factors) |
|---|---|
| 1) Individual | 1.1) Focus of attention<br>1.2) Interconnection<br>1.3) Adaptability |
| 2) Software Team | 2.1) Communication<br>2.2) Project Complexity<br>2.3) Team Leader |
| 3) Software Firm | 3.1) Administrative Change<br>3.2) Relationship with Clients<br>3.3) Work Environment |

According to the online survey, many of the psychological and sociological factors identified by researchers as being characteristic of systemic thinking in Section 2, are not taken into account by software development firms and teams. These variables though can have a positive effect on the success of software development teams and can assist software companies in better understanding their software teams. It is therefore important for businesses to make sure they understand how these variables affect their developers and improve their performance.

8 of the 50 people in the survey referred to technical aspects of engineering a system: e.g., management, technical depth of teams, producing a product or finding interfaces. The remaining people referred to communication, team interactions, and social leadership. This highlights the fact that systemic thinking is deeply ingrained in the daily activities of software engineers and therefore demonstrates the importance of psychological and sociological factors in developers' teams.

Commonly occurring phrases and ideas were tagged, organized, and combined when appropriate, resulting in a set of commonly cited codes. After a set of unique codes was identified (open coding) the codes were reorganized into categories (axial coding), grouping similar code constructs and providing greater explanatory power. Axial coding resulted in three main categories: individual, software team, and software firms as shown in Table 1. These categories were further developed into subcategories as informed by the critical constructs identified in the literature review discussed above.

# 5 SUMMARY OF THE RESULTS

Our literature review in Section 2 revealed that various psychological and sociological factors associated with systemic thinking are present in the operations of software development individuals, teams, and companies. These factors, as outlined in Section 4 , lead to a holistic approach to problem-solving and success in their work and projects. However, most companies and team members were not aware that they were utilizing systemic thinking in their daily projects and even stated that it was not practiced in their company, yet they recognized the significance of the related psychological and sociological factors such as communication and collaboration, indicating at least a basic level of understanding of its relevance.

Furthermore, the study of interview transcripts revealed three distinct categories of the use of systemic thinking in software development, which can

be roughly linked to individuals, teams, and corporations (as shown in Table 1). These categories are not mutually exclusive, and multiple perspectives on the same event may be included. These categories are examined in greater detail in the subsequent sections, which are loosely organized based on the problem description, context, and identify patterns.

Below, we discuss all the identified patterns from the survey:

*Focus of Attention*: Focus of attention is about the factors that a team leader takes into account when making decisions. When providing feedback or making crucial choices, some individuals concentrate on certain aspects of the problem, while others evaluate the available information holistically. Systemic thinking necessitates a readiness to comprehend the entire task under examination as having various and unique qualities and characteristics (so it requires both a reductionist and a holistic perspective).

*Interconnection:* To make sound decisions, managers must comprehend how the parts within the projects they manage are interrelated. It's also important for members of the project to understand how the decisions made by marketing managers, operation and account departments, or executives interrelate and how these choices affect the company's overall results. This is reflected in the interview transcripts by various statements that evaluate how the respondents perceived the relationship between the whole and the parts. Furthermore, when working on a project or sprint, most of the team members involved reported that they enjoyed seeing how their specific tasks fit into the larger project.

*Adaptability:* Systemic thinking emphasizes that team members should be creative and responsive to changing conditions when addressing challenges. This requires both the ability and the willingness to be adaptable. Skilled leaders tend to frame the situations they encounter in a way that allows for a nimble and flexible response. It's common for experienced decision-makers (leaders) to have a variety of suitable feedback (insights) for a specific issue. These insights are built up over time and can be accessed as needed. This enhances their personal adaptability and their ability to swiftly solve problems. In tackling a difficulty, the leader reflects on various past experiences and works to find an innovative solution. This aspect is reflected in the interview transcripts by various statements that evaluate the level of rigidity or flexibility with which participants completed projects/sprints.

*Communication:* Many software companies implemented distributed teams of software engineers in their daily operations. Some tech firms began implementing distributed teams after our research. By analyzing the responses from the participants in our survey, we can conclude that communication was cited by more than two-thirds of the subjects as an area that could be improved by taking into account psychological and sociological systemic factors.

*Project Intricacy:* Project complexity is determined by the difficulty in understanding how a particular project is being executed and can take many forms. Examples of complexity include coherent complexity, the complexity of the business domain, and computational and systemic complexity. Complexity and productivity have an inverse relationship. Many responses indicated that organizations had to narrow their focus on certain occasions when dealing with complex issues. Examples of project complexity include cost and time evaluation of construction activities. To effectively deal with complexity, it is important to understand the intricacy of the problem and develop solutions for software engineering issues that take into account the individual (e.g."character"), the team (e.g."standards"), and the software company (e.g."community").

*Team leader:* The behavior of the team leader is crucial for productivity. Team leaders can improve team performance by, for example, promoting a mindset that promotes focused communication, commitment, and motivation. Therefore, it is essential to consider team leadership when attempting to understand the performance of software teams. Additionally, the relationship between teamwork and the necessary skills for task recognition, planning, and training should be evaluated as consistently as possible.

*Administrative Change:* Many responses mentioned organizational change as an area where living systems should be taken into account. It was interesting to see how the changes were implemented and why they were made. The participants in the study suggested that administrative changes in software firms are more frequent than in other types of companies and that "staff should figure out how to adapt to changes since they happened constantly." Although the participants couldn't explain the reason for the high rate of change, they stated that "it is in the nature of software engineering" and that software development cycles, such as the agile method of work, "are a way of coping with changes." However, the understanding that "changes can happen at any moment" can put pressure on employees, which can be overwhelming. Many respondents also said that initiatives undergoing administrative change rarely "reach all the way" and get lost along the way for unexplained reasons. Another participant agreed with this statement and said that "teams and individuals are

powerful elements that are normally unaffected by administrative changes." The participants had some ideas about what kind of awareness they would need to help deal with ambiguous administrative change issues. However, the common idea was that the problems could not be solved simply by focusing on practical issues. They required a multidisciplinary approach, the type of approach proposed by systemic thinking.

*Relationship with clients:* The participants agreed that the supplier-customer relationship has a significant impact on the likelihood of a successful outcome. Some believed that the relationship could positively influence team motivation. One issue identified with client relations was that software releases (updating software versions) often resulted in organizational changes for the customer. According to the participants, these changes can threaten the relationship between clients and employees. As a result, the workers may develop a hostile attitude towards the supplier as a psychological defense mechanism. If not handled properly, this can negatively impact collaboration between the parties involved.

*Work Environment:* The physical work environment can have a positive effect on the performance and efficiency of a software team. For example:

1. A work environment that promotes collaboration can boost productivity.
2. Resources such as computer networks and software tools are essential for the workplace.
3. These resources can facilitate more effective collaboration, team service, and individual cohesion, all of which have a significant positive impact on performance.
4. Software industrial facilities are conditions that encourage the standardization of methods and software measurement concepts.

This can also significantly influence teams' efficiency as well as productivity.

## 6 CONCLUSION

This study has revealed the importance of systemic thinking in software development. The psychological and sociological factors associated with systemic thinking, such as communication and collaboration, were found to be present in the operations of software development individuals, teams, and companies. The study also identified three distinct categories of the use of systemic thinking in software development, which can be linked to individuals, teams, and corporations. Additionally, it was found that team leader behavior plays a crucial role in productivity and that organizational change can be a challenging environment. Participants also agreed that the supplier-customer relationship has a significant impact on the outcome and the physical work environment can have a positive effect on the performance and efficiency of a software team. These findings highlight the need for a multidisciplinary approach, as proposed by systemic thinking, to effectively deal with the complexities and challenges in software development.

While this study provides valuable insights into the role of systemic thinking in software development, it has several limitations. One limitation is that the study is based on a small sample of participants and may not be generalizable to the larger population of software development companies. Additionally, the study focused only on the perception of the participants, further research with objective measures could provide a more comprehensive understanding of the topic. Another limitation is that the study only focused on one perspective of systemic thinking which is the psychological and sociological factors, further research could explore other aspects of systemic thinking such as technical and organizational factors.

Future research could also explore the effects of systemic thinking on the performance and efficiency of software development teams in a more controlled setting. Additionally, it would be interesting to investigate how different software development methods, such as Agile and Waterfall, affect the use of systemic thinking. Moreover, future research could also investigate the impact of different team structures and remote working on the use of systemic thinking in software development. These studies could provide a more in-depth understanding of the role of systemic thinking in software development and help organizations to improve their performance and efficiency.

## ACKNOWLEDGEMENTS

This research is supported by the Russian Science Foundation, Grant No. 22-21-00494.

## REFERENCES

Acuña, S. T., Gómez, M., and Juristo, N. (2008). Towards understanding the relationship between team climate and software quality—a quasi-experimental

study. *Empirical software engineering*, 13(4):401–434.

Ahmed, F., Capretz, L. F., and Campbell, P. (2012). Evaluating the demand for soft skills in software development. *IT Professional*, 14(1):44–49.

Alexander, J. C. (1990). *Durkheimian sociology: cultural studies*. Cambridge University Press.

Allen, I. E. and Seaman, C. A. (2007). Likert scales and data analyses. *Quality progress*, 40(7):64–65.

Amrit, C. A., Daneva, M., and Damian, D. (2014). Human factors in software development: On its underlying theories and the value of learning from related disciplines. a guest editorial introduction to the special issue. *Information and software technology*, 56(12):1537–1542.

Beecham, S., Baddoo, N., Hall, T., Robinson, H., and Sharp, H. (2008). Motivation in software engineering: A systematic literature review. *Information and software technology*, 50(9-10):860–878.

Bentler, P. M., Jackson, D. N., and Messick, S. (1971). Identification of content and style: A two-dimensional interpretation of acquiescence. *Psychological Bulletin*, 76(3):186.

Bhardwaj, M. and Rana, A. (2016). Key software metrics and its impact on each other for software development projects. *ACM SIGSOFT Software Engineering Notes*, 41(1):1–4.

Boardman, J. and Sauser, B. (2008). *Systems thinking: Coping with 21st century problems*. CRC Press.

Bonner, H. (1959). *Group dynamics; principles and applications*. New York: Ronald Press Company.

Bradburn, N. M. and Sudman, S. (2004). The current status of questionnaire research. *Measurement errors in surveys*, pages 27–40.

Bugayenko, Y., Bakare, A., Cheverda, A., Farina, M., Kruglov, A., Plaksin, Y., Succi, G., and Pedrycz, W. (2022a). Automatically prioritizing and assigning tasks from code repositories in puzzle driven development. In *Proceedings of the 19th International Conference on Mining Software Repositories*. ACM.

Bugayenko, Y., Daniakin, K., Farina, M., Jolha, F., Kruglov, A., Succi, G., and Pedrycz, W. (2022b). Extracting corrective actions from code repositories. In *Proceedings of the 19th International Conference on Mining Software Repositories*. ACM.

Capretz, L. F. (2014). Bringing the human factor to software engineering. *IEEE software*, 31(2):104–104.

Ciancarini, P., Farina, M., Masyagin, S., Succi, G., Yermolaieva, S., and Zagvozkina, N. (2021a). Non verbal communication in software engineering–an empirical study. *IEEE Access*, 9:71942–71953.

Ciancarini, P., Farina, M., Masyagin, S., Succi, G., Yermolaieva, S., and Zagvozkina, N. (2021b). Root causes of interaction issues in agile software development teams: Status and perspectives. In *Future of Information and Communication Conference*, pages 1017–1036. Springer.

Ciancarini, P., Farina, M., Okonicha, O., Smirnova, M., and Succi, G. (2023). Software as storytelling: A systematic literature review. *Computer Science Review*, 47:100517.

Clincy, V. A. (2003). Software development productivity and cycle time reduction. *Journal of Computing Sciences in Colleges*, 19(2):278.

Connolly, T. (1997). Electronic brainstorming: Science meets technology in the group meeting room. teoksessa culture of the internet. ed. by sara kiesler.

Cooley, C. H. (1992). *Human nature and the social order*. Transaction Publishers.

Cruz, S. S., da Silva, F. Q., Monteiro, C. V., Santos, P., Rossilei, I., and dos Santos, M. (2011). Personality in software engineering: Preliminary findings from a systematic literature review. In *15th annual conference on Evaluation & assessment in software engineering (EASE 2011)*, pages 1–10. IET.

Curtis, B. (1984). Fifteen years of psychology in software engineering: Individual differences and cognitive science. In *Proceedings of the 7th international conference on Software engineering*, pages 97–106.

Davidow, W. H. (1992). The virtual corporation: Structuring and revitalizing the corporation for the 21st century.

Dlamini, G., Ergasheva, S., Kholmatova, Z., Kruglov, A., Sadovykh, A., Succi, G., Timchenko, A., Vasquez, X., and Zouev, E. (2022). Metrics for software process quality assessment in the late phases of SDLC. In *Lecture Notes in Networks and Systems*, pages 639–655. Springer International Publishing.

Edson, R. (2008). Systems thinking. applied. a primer. arlington, va: Analytic services.

Farina, M., Gorb, A., Kruglov, A., and Succi, G. (2022). Technologies for GQM-based metrics recommender systems: A systematic literature review. *IEEE Access*, 10:23098–23111.

Feldt, R., Angelis, L., Torkar, R., and Samuelsson, M. (2010). Links between the personalities, views and attitudes of software engineers. *Information and Software Technology*, 52(6):611–624.

Feldt, R., Torkar, R., Angelis, L., and Samuelsson, M. (2008). Towards individualized software engineering: empirical studies should collect psychometrics. In *Proceedings of the 2008 international workshop on Cooperative and human aspects of software engineering*, pages 49–52.

Finholt, T., Sproull, L., and Kiesler, S. (1990). Communication and performance in ad hoc task groups, intellectual teamwork: social and technological foundations of cooperative work, l.

Freeman, L. C. (1992). The sociological concept of" group": An empirical test of two models. *American journal of sociology*, 98(1):152–166.

Gallón, L. (2020). Systemic thinking. *Quality Education*, pages 830–840.

Gogichaty, M., Ivanov, V., Kruglov, A., Pedrycz, W., Samatova, A., Succi, G., and Valeev, R. (2023). A systemic approach to evaluating the organizational agility in large-scale companies. *IEEE Access*, 11:3307–3323.

Hall, T., Baddoo, N., Beecham, S., Robinson, H., and Sharp, H. (2009). A systematic review of theory use in studies investigating the motivations of software engineers. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 18(3):1–29.

Hare, A. P. (1994). Types of roles in small groups: A bit of history and a current perspective. *Small Group Research*, 25(3):433–448.

Hazzan, O. and Hadar, I. (2008). Why and how can human-related measures support software development processes? *Journal of Systems and Software*, 81(7):1248–1252.

Kang, D., Jung, J., and Bae, D.-H. (2011). Constraint-based human resource allocation in software projects. *Software: Practice and Experience*, 41(5):551–577.

Katzenbach, J. R. and Smith, D. K. (2008). *The discipline of teams*. Harvard Business Press.

Khan, I. A., Brinkman, W.-P., and Hierons, R. M. (2011). Do moods affect programmers' debug performance? *Cognition, Technology & Work*, 13(4):245–258.

Kruglov, A. (2021). Impact of the communication issues: A case study of IT start-up. In *Communications in Computer and Information Science*, pages 113–132. Springer International Publishing.

Lamb, C. M. (2009). *Collaborative Systems Thinking: An exploration of the mechanisms enabling team systems thinking*. PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and ....

Lee, S.-U., Park, J.-Y., and Park, J.-H. (2003). An ideal team size in development and maintenance project. *The KIPS Transactions: PartD*, 10(1):77–84.

Leidner, D. E. and Kayworth, T. (2006). A review of culture in information systems research: Toward a theory of information technology culture conflict. *MIS quarterly*, 30(2):357–399.

Lenberg, P., Feldt, R., and Wallgren, L.-G. (2014). Towards a behavioral software engineering. In *Proceedings of the 7th international workshop on cooperative and human aspects of software engineering*, pages 48–55.

Lipson, H. (2007). Principles of modularity, regularity, and hierarchy for scalable systems. *Journal of Biological Physics and Chemistry*, 7(4):125.

Locke, E. A., Shaw, K. N., Saari, L. M., and Latham, G. P. (1981). Goal setting and task performance: 1969–1980. *Psychological bulletin*, 90(1):125.

Lytras, M. D., Ordonez de Pablos, P., Ziderman, A., Roulstone, A., Maurer, H., and Imber, J. B. (2010). Organizational, business, and technological aspects of the knowledge society. *Organizational*, 112.

Matturro, G. (2013). Soft skills in software engineering: A study of its demand by software companies in Uruguay. In *2013 IEEE/ACM 6th International Workshop on Cooperative and Human Aspects of Software Engineering —CHASE)*, pages 133–136.

Matturro, G., Raschetti, F., and Fontán, C. (2015). Soft Skills in Software Development Teams: A Survey of the Points of View of Team Leaders and Team Members. In *2015 IEEE/ACM 8th International Workshop on Cooperative and Human Aspects of Software Engineering — CHASE*, pages 101–104.

McGrath, J. E. (1984). *Groups: Interaction and performance*, volume 14. Prentice-Hall Englewood Cliffs, NJ.

Mulder, G. (1986). The concept and measurement of mental effort. In *Energetics and human information processing*, pages 175–198. Springer.

Nerur, S. (2014). Distributed cognition in software design: An experimental investigation of the role of design patterns and collaboration. *MIS Quarterly*, 38(1):249–274.

Nunamaker, J. F., Dennis, A. R., Valacich, J. S., Vogel, D., and George, J. F. (1991). Electronic meeting systems. *Communications of the ACM*, 34(7):40–61.

Offer, J. (2010). *Herbert Spencer and social theory*. Springer.

Oreg, S. (2003). Resistance to change: Developing an individual differences measure. *Journal of applied psychology*, 88(4):680.

Parsons, T. (1977). *Social systems and the evolution of action theory*. Free Press.

Schultz, D. and Schultz, S. (1998). An introduction to industrial and organizational psychology.

Sharp, H., Baddoo, N., Beecham, S., Hall, T., and Robinson, H. (2009). Models of motivation in software engineering. *Information and software technology*, 51(1):219–233.

Sheridan, T. B. (2010). The system perspective on human factors in aviation. In *Human factors in aviation*, pages 23–63. Elsevier.

Srivastava, A., Bartol, K. M., and Locke, E. A. (2006). Empowering leadership in management teams: Effects on knowledge sharing, efficacy, and performance. *Academy of management journal*, 49(6):1239–1251.

Stichweh, R. (2011). Systems theory. *International Encyclopedia of Political Science. New York: Sage*.

Stogdill, R. M. (1959). Individual behavior and group achievement: A theory; the experimental evidence.

Sudhakar, G. P., Farooq, A., and Patnaik, S. (2011). Soft factors affecting the performance of software development teams. *Team Performance Management: An International Journal*.

Tornhill, A. (2015). *Your code as a crime scene: use forensic techniques to arrest defects, bottlenecks, and bad design in your programs*. Pragmatic Bookshelf.

Trendowicz, A. and Münch, J. (2009). Factors influencing software development productivity—state-of-the-art and industrial experiences. *Advances in computers*, 77:185–241.

Trendowicz, A., Münch, J., and Jeffery, R. (2008). State of the practice in software effort estimation: a survey and literature review. In *IFIP Central and East European Conference on Software Engineering Techniques*, pages 232–245. Springer.

Unterkalmsteiner, M., Gorschek, T., Islam, A. M., Cheng, C. K., Permadi, R. B., and Feldt, R. (2011). Evaluation and measurement of software process improvement—a systematic literature review. *IEEE Transactions on Software Engineering*, 38(2):398–424.

Zahra, S. A. and Bogner, W. C. (2000). Technology strategy and software new ventures' performance: Exploring the moderating effect of the competitive environment. *Journal of business venturing*, 15(2):135–173.