WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Analysis of Challenges in Software Engineering and Recommendations for Project Management

David Kuhlen

IU International University of Applied Sciences, Waterloohain 9, 22769 Hamburg, Germany. Mail: david.kuhlen@iu.org

A B S T R A C T

The development of software is associated with numerous challenges that require a wide range of skills from both software developers and project managers. Various works in the field of research describe these challenges. Project managers can prevent the occurrence of challenges by knowing, monitoring, and effectively countering them when necessary. However, the sheer volume of challenges makes it difficult for project managers to maintain an overview. This paper analyzes these challenges through empirical surveys, which have been conducted and evaluated. A categorization enables a connection to the software development process and provides actionable recommendations for project managers. The provided recommendations can offer project managers initial guidance. However, the results from Q2 indicate that the most significant challenges in practice could not be captured by the proposed options (22.6%), meaning that the recommendations must be continuously and situationally assessed for their applicability in practice and potentially expanded. Although the considerable diversity of software development projects, which is also rooted in the high level of innovation within the industry, hinders the formulation of universally applicable recommendations, the recommendation presented here, based on practical experiences and discussions with students, appears to be valid for many projects in an initial context.

Keywords: software engineering, challenges, software project management

Introduction

The development of software is considered a challenging task and requires well-trained software developers. In many companies, software development is becoming a critical success factor [HMF+01]. However, the economic quality of the software development process is sometimes critically evaluated [Ar14, p. 42]. To achieve optimal results, it seems reasonable to examine the challenges and skill requirements of developers.

The body of research in the field of software engineering includes a wide range of works that either specifically or incidentally address the challenges of software development and offer solutions. This article highlights selected challenges. However, capturing all possible challenges seems difficult due to the diversity of available information. This poses a challenge for software development projects, as challenges are generally easier to avoid when they are known. Particularly in projects where software is developed, there is a risk that challenges could lead to problems, as the development process might not be standardized or repeatable due to the project-specific nature.

To contribute to improvements, this paper examines the greatest challenges in software development. The attempt is made to reduce the multitude of possible challenges to a few groups. Subsequently, the connection of these meta-challenges to the software development process is elaborated. Based on this, actionable recommendations can be formulated.

This paper addresses the following research questions: RQ1: What are the biggest challenges that software developers must overcome? RQ2: What key competencies do software developers require? RQ3: What recommendations can be given to project managers at the start of a project to address these challenges?

In the following section, selected challenges are explored based on the literature. The subsequent section outlines the methodology. Then, the results of the empirical investigations and the recommendations for action are presented and discussed. The paper concludes with a summary and an outlook on future research.

Related Work

Over the past decades, core tasks and challenges of software engineering have been identified. The associated discourse covers various topics, often quite diverse, such as challenges for software architectures due to different development styles [Br95], the consideration of axes of change [FC96, p. 59], or the discussion of market entry barriers for software manufacturers [Ba08, p. 185]. A focus on the risks of software projects can be found in works such

as [SL01]. Gumm and Sommer highlight challenges such as error-free development, reusability, modifiability, extensibility, applicability, and methodological organization [GS13, p. 828-830].

Balzert outlines the competency requirements for various participants in the software development process, differentiated by roles [Ba08, p. 102 f.]. According to Balzert, software developers need the ability for abstraction, mathematical understanding, creativity, and precision [Ba08, p. 102].

Sommerville emphasizes that the competencies required of software developers go beyond technical skills [So16, p. 28 f.]. For example, software developers are expected to act morally, adhering to ethical principles, ensuring the protection of rights, and acting in a socially responsible manner [So16, p. 28 f.]. In this regard, Sommerville [So16] refers to the ACM/IEEE Code of Ethics [ACM/IEEE 99]. According to Plewan and Poensgen, the software industry lacks generally accepted methods necessary for establishing a professional discipline [PP11, p. 17].

Various challenges affect the organization of the software development process, understanding requirements, developing a valid software structure, and team collaboration. Partial aspects of these issues have been addressed in various works. For example, in project management, there are challenges in measuring development progress [Ba08, p. 37]. The quality of emerging software structures is reduced when software developers spend too much time perfecting a software structure, without it being economically justified [Ba08, p. 17 & 207]. Developing a valid software structure becomes a challenge within the team when different developers solve tasks in various ways [Br95, p. 48 f. & 257] [Pu78]. Communication plays a crucial role in successful team collaboration [CNM97]. However, determining the right amount of communication is a demanding task [DH07, p. 29; HC93].

Method

To answer the research questions RQ1 and RQ2, a survey was conducted using [Mentimeter]. [Mentimeter] is an online solution that allows the creation of interactive presentations. These presentations can, for example, include questions that the audience can answer anonymously and in real-time during the presentation. The survey was conducted alongside lectures. Participants in the survey are dual bachelor students from various semesters. Dual students spend a portion of their studies at a company, where they gain practical experience in the profession they are studying. For this reason, participants were asked to answer certain questions related to their company. The participants were asked the following questions:

Q1	**Question**: What do you believe is the most difficult aspect of programming? **Answer options:** a) Planning the approach organizationally b) Properly understanding the requirements c) Building a valid software structure d) Mastering the programming language e) Reaching a consensus on a solution within the team f) Something else **Response**: Participants were asked to choose one of the provided answer options. The focus was on the participants' personal assessment. Due to the nature of the question, multiple selections were not allowed. **Evaluation**: To evaluate the responses, the frequency with which each of the provided answer options was chosen will be determined. These frequencies will be compared to the total number of responses. This will reveal a majority sentiment, showing which challenge was selected by the greatest number of participants as the most significant challenge.
Q2	**Question**: What are the biggest problems in software development, and how are these solved in your company? **Answer options**: a) Planning the approach organizationally b) Properly understanding the requirements c) Building a valid software structure

	d) Mastering the programming language e) Reaching a consensus on a solution within the team f) Something else **Response**: Participants were asked to discuss question Q2 with their training company and provide an answer reflecting the perspective of their company. Due to the nature of the question, multiple selections were not allowed. **Evaluation**: To evaluate the responses, the frequency with which each of the provided answer options was chosen will be determined. These frequencies will be compared to the total number of responses. This will reveal a majority sentiment, showing which challenge was selected by the greatest number of participants as the most significant challenge in their company.
Q3	**Question**: Assuming mastery of the programming language as a fundamental core competence, which additional skills does a good developer need? **Answer options**: a) Understanding customer requirements b) Planning a sustainable software structure c) Ability to anticipate future change needs d) Ensuring software quality e) Planning a development approach f) Communication skills for interaction with customers and colleagues **Response**: Participants were asked to provide their personal assessment. For each of the provided answer options, participants rated them on a scale from 0 (= Irrelevant) to 10 (= Very important). **Evaluation**: To evaluate the responses, the mean scores for each of the answer options will be calculated and compared. Additionally, further statistical analysis will be conducted by calculating measures of central tendency such as the median, minimum, first quartile, third quartile, and the upper whisker, which will be displayed in a box plot diagram.

The questions Q1 and Q2 allow for directly comparable results, showing the different evaluations of challenges from the participants' personal perspective as well as from the perspective of their training companies. For this reason, the responses to Q1 and Q2 are compared with each other. To facilitate this comparison, Q1 and Q2 use the same answer options. These answer options are limited to five technical challenges (a to e). Additionally, participants have the option to select option f if none of the listed challenges, in their view, represent the biggest challenge in software development. Questions Q1 and Q2 contribute to answering RQ1. In Q3, the relevance of skills related to typical tasks of software developers is examined. The participants' assessments help contribute to answering RQ2.

Based on the insights gained through the review of literature and empirical research, a set of recommendations for project management will be formulated. These recommendations are also based on experiences and discussions with students regarding the handling of typical challenges.

Results

The results of Q1 reflect the participants' perspective on the greatest challenges in software development. Question Q1 was answered by 84 participants. From their viewpoint, understanding customer requirements (option b) is perceived as the greatest challenge, with 33.3% of participants selecting option b as the biggest challenge in software development. Among the technical options a) to e), option e) (Teamwork) was selected the least as the greatest challenge (4.8%).

The results of Q2 show the evaluation of possible challenges in software development from the perspective of the training companies. It should be noted that the perspective of the training companies refers to the viewpoints of selected employees from the companies, whose identities are not known. Due

to the anonymous nature of the data, it is not possible to verify whether these responses truly represent the perspective of the companies. There are 25 responses to question Q2.

The evaluation of the results from Q2 reveals that in most cases (22.6%), none of the listed options represent the greatest challenge as perceived by the companies. Among the technical options a) to e), understanding requirements (option b) was again selected as the greatest challenge, with 22.6% of responses. The least frequently selected option (3.2%) was option c) as the greatest challenge in software development.

Figure 1 summarizes the results of Q1 and Q2 and contrasts the respective responses. Among the technical options a) to e), the greatest differences are observed primarily in the selection of options c) and d). While participants personally chose mastering the programming language (d) and building a valid software structure (d) more frequently as challenges, these options were selected less frequently as the greatest challenges from the company perspective in the Q2 data. This is understandable for option d). While mastering the programming language may still be a significant challenge from the perspective of the participating students, it is assumed that this challenge has been overcome in companies, and therefore it is no longer selected as frequently as the greatest challenge.

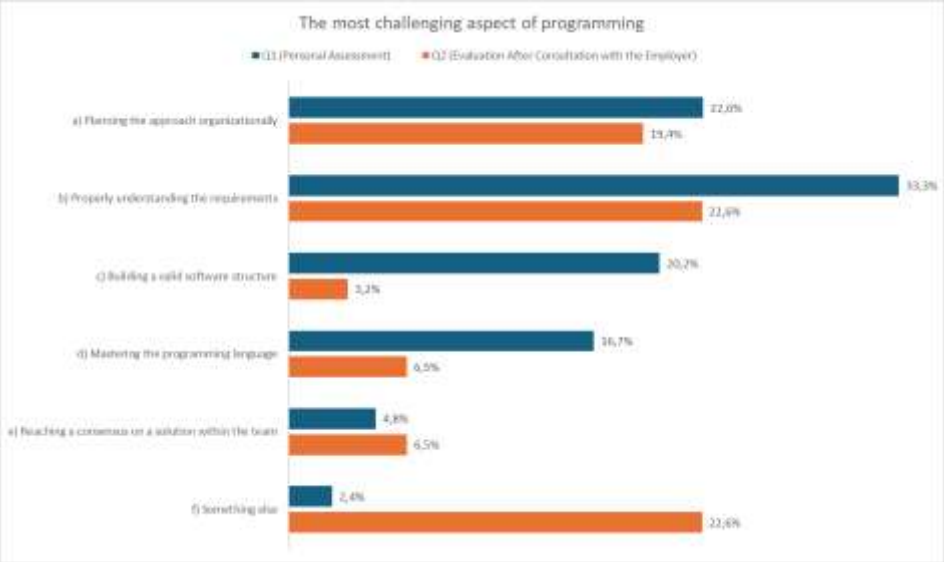

Figure 1 - Results of Q1. Own illustration, created with [Microsoft® Excel® 2019].

In the context of Q3, participants were asked to provide their personal assessment of important competencies for software developers that go beyond mastery of the programming language (see option d in Q1). In accordance with the previous section's description, the data from Q1 are evaluated by calculating the mean score and analyzing additional statistical measures of central tendency. Figure 2 presents the results of Q3, comparing the mean ratings of the various categories.

Similar to the findings from Q1, the evaluation of the data from Q3 highlights that understanding customer requirements is considered the most important competency for a software developer (option a with a mean of 7.9). Communication skills (option f with a mean of 7.7) were rated as the second most important competency. The least important competency was option e) (with a mean of 5.9).

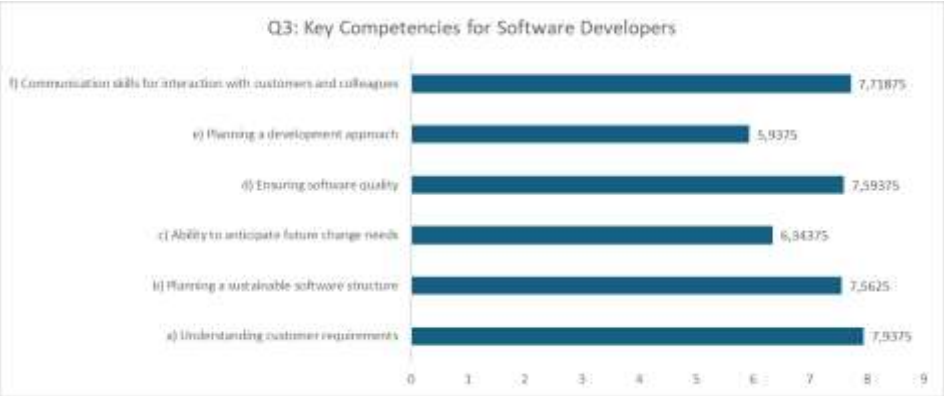

Figure 2 - Results of Q3. Own illustration, created with [Microsoft® Excel® 2019].

Figure 3 shows the results of Q3 in the form of a boxplot diagram. Notably, there is a high level of confidence among participants regarding the relevance of competency options a), b), and d). Participants were relatively certain about the importance of understanding customer requirements, the ability to plan

a software structure, and the ability to ensure software quality. In contrast, the ratings for competency options e) and f) show significant variability. The opinions on the importance of skills related to planning the development approach and communication were highly diverse among the participants.

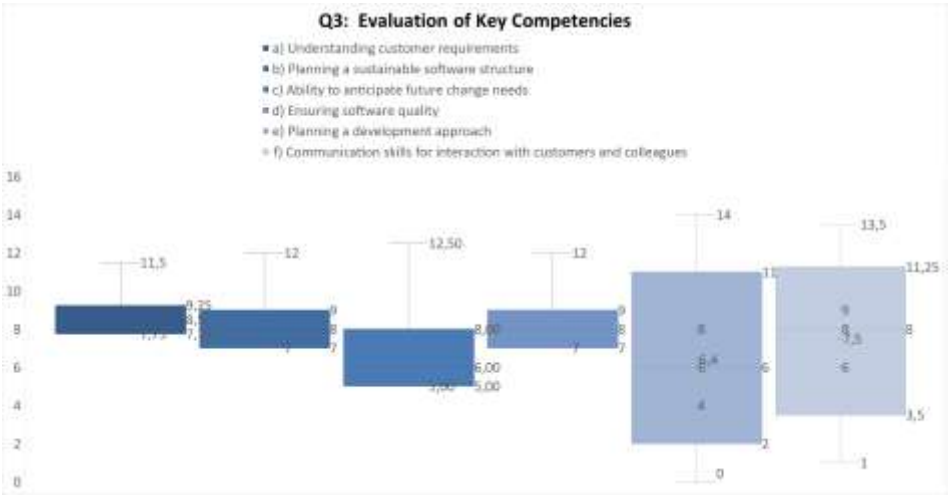

Figure 3 - Results of Q3 in form of a boxplot. Own illustration, created with [Microsoft® Excel® 2019].

The data collected during the empirical studies on challenges, as well as insights from the state of research, suggest that three categories of challenges should be considered: requirement-related challenges, software structure-related challenges, and organizational challenges. For better clarity, these categories can be related to steps in the software development process.

Requirement-related challenges include, for example, high requirement volatility [SL01, p. 19] or high innovation pressure [KJS+06; GS13]. Software-related challenges involve issues such as an unproductive drive for perfectionism [Ba08, p. 17, 207] or the accumulation of technical debt [Cu92; MS12, p. 75; Al12]. Organizational challenges include, for instance, planning difficulties due to the need for creativity [Ba08, p. 19, 32] as well as the correct application of the methodology [PP11].

The diagram in Figure 4 shows the software development process in connection with challenges that can occur during development. Similar to the representation in [KS15, p. 160], challenges cause setbacks in the process when they arise. Figure 4 represents the process as a UML activity diagram and incorporates challenges by associating them as use cases. It should be noted that this representation deviates from the UML standard and its associated intention. The depiction serves as a guide.

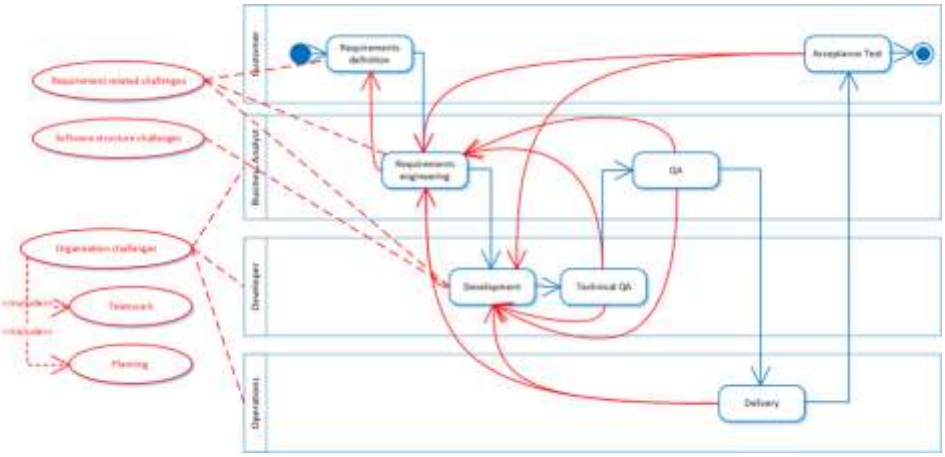

Figure 4 - Process of software development in connection with challenges. Own illustration, created with [Microsoft® Visio® 2021].

The illustration in Figure 4 clearly shows that setbacks naturally affect the steps of development as well as requirements engineering. Challenges related to requirements or software structure are directly associated with the activities of requirements engineering and development. It is important to note that requirement-related challenges can also lead to additional efforts on the customer's side, for instance, when details about requirements are missing. Organizational challenges have an overarching relationship with the software development process and cannot be assigned to individual steps in the process.

As a recommendation for avoiding challenges, it is essential to establish risk management and maintain it throughout the software development process. A description of how to approach risk management can be found, for example, in [PR18]. Such risk management assumes that the project manager regularly coordinates with the project participants and notices early when the likelihood or impact of a risk changes [PR18]. This requires the project manager to have technical knowledge about the diverse challenges that can arise during software development. Additionally, the project manager must

be willing to engage with the details of the software development process, even if they involve technical aspects. To prevent the emergence of challenges, the steps of requirements engineering and software development should be carried out with the utmost care and in accordance with industry standards. This includes, for instance, employing good programming techniques during software development [GS13; Ba09; Ma03; Fo19].

To ensure compliance with industry standards, attention should be given to ensuring uniform qualifications of all involved participants [Ba08]. Developers who engage in test-driven development achieve structural advantages for the software architecture [Ma03] and facilitate later adaptations to new or changed customer requirements [Fo19]. Furthermore, the process model used for a software development project should be carefully selected and correctly applied. If necessary, further training should be conducted before the process model is applied. The working methods should be regularly reviewed and adjusted if necessary to achieve the best possible outcome.

Discussion

The results from Q1 and Q3 may appear contradictory when compared. While the participants indicated on average in Q1 that teamwork (Option d) rarely represents the greatest challenge in software development, Q3 clearly showed that software developers particularly need communication skills (Option f was rated with 7.7 points). However, it should be noted that the need for good communication skills, as indicated in Q3, does not necessarily mean that these skills are insufficient in practice, leading to problems. Additionally, the results from Q1 showed that participants frequently selected the planning of the software development process (Option a) as the greatest difficulty in software development. This major challenge does indeed require communication skills to be overcome.

Furthermore, the significance of planning the development process, as indicated by Q1, is surprising. With a probability of 22.6%, participants in Q1 selected the corresponding Option a) as the greatest challenge in software development. In contrast, in Q3, participants indicated that planning the development process was of lower importance compared to other competencies for software developers (Option e in Q3). It should be noted that the planning of the development process is not necessarily carried out by software developers. Often, the planning of the development process is the responsibility of project managers or individual software developers in leadership positions. Therefore, the competence to overcome the planning challenge may not be required across all software developers.

Regarding the processual representation, mixtures of challenges are not considered in detail. For example, language barriers between customers and software developers [HSF+08, p. 1017] lead to both requirement-related challenges and organizational challenges. It should be noted that organizational challenges, due to their overarching nature, are related to many other challenges. For instance, one could argue whether technical debt [Cu92] could be avoided if the software development process were planned differently. Therefore, when considering challenges, priority should be given to categorizing the more specific challenges before the more general ones.

The given recommendations are relatively general and refer to established project management methodologies, such as those described in [PR18]. To enable the initial setup of a software development project that avoids challenges, the recommendation connects selected challenges with countermeasures. As mentioned before, it should be emphasized that neither the lists of challenges nor countermeasures can claim to be exhaustive. In particular, software development projects that are highly innovative will face many unique challenges that are not yet known. The study conducted here highlights the importance of requirements engineering activities, software development, and overarching organizational management. Based on practical experiences in various software development projects and the results of discussions with students, it seems especially important that the project manager regularly monitors the occurrence of challenges. For this, they must be informed and, if necessary, also engage with the details. It should be noted that not every project manager possesses the necessary technical qualification for this. In these cases, expanding the project management by involving subject matter experts, who can manage and monitor sub-areas of a project, may be a viable solution that should be evaluated for its applicability.

Conclusion

The consideration of software engineering challenges reveals that various risks can influence the success of software development. Five specific technical risks were examined and further analyzed through a survey (Q1/Q2). The survey shows that, from a practical perspective (Q2), this compilation does not reflect the greatest challenge in most cases (22.6%). Furthermore, the results from Q1 and Q2 indicate that planning and understanding requirements have a high probability of being the greatest challenges. An investigation into the key competencies (Q3) that software developers need reveals that many skills were rated with nearly equal importance. Notably, the competencies identified by the participants in Q3 as most essential were understanding requirements (a with 7.9 points) and communication skills (f with 7.7 points).

Given the large variety of potential challenges, they were categorized after the survey into "Requirement-related challenges," "Software structure challenges," and "Organizational challenges." These categories were mapped to the software development process. A set of recommendations was provided for project managers, which includes establishing risk management, regular risk monitoring, adherence to industry standards, and routine checks of the working process (see the Results section).

For future research following this study, it is important to explore whether there are additional main categories of challenges beyond requirements, software, and organizational challenges. These findings could be used to further systematize the given recommendations and augment them. Since the list of recommendations provided here cannot be exhaustive and serves only as a preliminary orientation for project managers, conducting an experiment

or case study might be beneficial to test and complement these recommendations. A separate, additional research area would focus on the extent to which universal recommendations and challenges can be identified for the fast-paced and innovative software development industry.

References

[ACM/IEEEE 99] Association for Computing Machinery, Inc. (ACM) / the Institute for Electrical and Electronics Engineers, Inc. (IEEE) (1999): "The Software Engineering Code of Ethics and Professional Practice". URL: https://ethics.acm.org/code-of-ethics/software-engineering-code/ accessed 2025-03-19. [based on GMR97].

[Al12] Allman, Eric (2012): "Managing Technical Debt." In: COMMUNICATIONS OF THE ACM March 2012 Vol. 55, Nr. 5. pp. 50-55. [DOI: 10.1145/2160718.2160733].

[Ar14] Armour, Phillip G. (2014): “The Business of Software Estimation Is Not Evil.” In: COMMUNICATIONS OF THE ACM 2014 Nr. 1 Vol. 57. pp. 42-43. [DOI: 10.1145/2542505].

[Ba08] Balzert, Helmut (2008): Lehrbuch der Softwaretechnik. Softwaremanagement. 2. Auflage. Heidelberg: Spektrum Akademischer Verlag (=Lehrbücher der Informatik). [ISBN 978-3-8274-1161-7]

[Ba09] Balzert, Helmut (2009): Lehrbuch der Softwaretechnik. Basiskonzepte und Requirements Engineering. 3. Auflage. Heidelberg: Spektrum Akademischer Verlag (=Lehrbücher der Informatik). Springer. [ISBN 978-3-8274-1705-3].

[Br95] Brooks, Jr., Frederick P. (1995): The Mythical Man-Month. Essay on Software-Engineering. Anniversary edition with four new chapters. Boston u. a.: Addison-Wesley Longman, Inc. (=Pearson Education). [ISBN 0-201-83595-9].

[CNM97] Coad, Peter / North, David / Mayfield, Mark (1997): Object Models. Strategies, Patterns, & Applications. Second Edition. Upper Saddle River, NJ, USA: YOURDON PRESS, Prentice Hall. [ISBN: 0-13-840117-9].

[Cu92] Cunningham, Ward (1992): "Experience Report - The WyCash Portfolio Management System". In: ACM SIGPLAN OOPS Messenger, Vol. 4, Nr. 2. October 1992. OOPSLA '92 Addendum to the Proceedings. pp. 29 - 30. [DOI: 10.1145/157710.157715].

[DH07] Drews, Günter / Hillebrand, Norbert (2007): Lexikon der Projektmanagement-Methoden. 1. Auflage. München: Rudolf Haufe Verlag GmbH & Co. KG. 2007. [ISBN 978-3-448-08052-0].

[FC96] Fayad, Mohamed / Cline, Marshall P. (1996): „Aspects of software adaptability.“ In: Communications of the ACM. Vol. 39, Nr. 10, October 1996. pp. 58-59. [DOI: 10.1145/236156.236170].

[Fo19] Fowler, Martin (2019): Refactoring. Improving the Design of Existing Code. Second Edition. Boston u. a.: Addison-Wesley, Pearson Education, Inc. [ISBN-13: 978-0-13-475759-9].

[GMR97] Gotterbarn, Don / Miller, Keith / Rogerson, Simon (1997): "Software Engineering Code of Ethics". Communications of the ACM, Vol. 40, Nr. 1, November 1997. pp. 110 - 118. [DOI: 10.1145/265684.265699].

[GS13] Gumm, Heinz-Peter / Sommer, Manfred (2013): Einführung in die Informatik. 10. Auflage. München: Oldenbourg Verlag. [ISBN: 978-3-486-70641-3]

[HC93] Hammer, Michael / Champy, James (1993): Reengineering the Corporation. A Manifesto for Business Revolution. New York: HarperBusiness a division of HarperCollinsPublishers. 1993. [ISBN 0-88730-687-X].

[HMF+01] Herbsleb, James D. / Mockus, Audris / Finholt, Thomas A. / Grinter, Rebecca E. (2001): „An Empirical Study of Global Software Development: Distance and Speed.“ In: Proceedings of the 23rd International Conference on Software Engineering. ICSE 2001. pp. 81-90. 2001. IEEE Computer Society. [DOI: 10.1109/ICSE.2001.919083].

[HSF+08] Heise, David / Strecker, Stefan / Frank, Ulrich / Jung, Jürgen (2008): „Erweiterung einer Unternehmensmodellierungsmethode zur Unterstützung des IT-Controllings.“ In: Multikonferenz Wirtschaftsinformatik 2008. Berlin: GITO-Verlag. Universität Duisburg-Essen, Campus Essen, Lehrstuhl für Wirtschaftsinformatik und Unternehmensmodellierung Institut für Informatik und Wirtschaftsinformatik (ICB). pp. 1017 - 1028.

[KJS+06] Kumar, Subodha / Ji, Yonghua / Sethi, Suresh / Yeh, Denny (2006): „Dynamic Optimization of Software Enhancement Effort.“ In: Proceedings of 16th Workshop on Information Technologies and Systems (WITS). Milwaukee:. December 2006. [DOI: 10.2139/ssrn.1026842].

[KS15] Kuhlen, David / Speck, Andreas (2015): „Business process analysis by model checking.“ In: Ceravolo, Paolo (Hg.) / Rinderle-Ma, Stefanie (Hg.): Proceedings of the 5th International Symposium on Data-Driven Process Discovery and Analysis SIMPDA December 2015. Vienna, Austria. pp. 154-170.

[Ma03] Martin, Robert C. (2003): Agile Software Development. Principles, Patterns, and Practices. Upper Saddle River, NJ USA: Prentice Hall, Pearson Education, Inc. [ISBN: 0-13-597444-5. Robert C. Martin with contributions by James W. Newkirk and Robert S. Koss].

[Mentimeter] Mentimeter AB Tulegatan 11 in SE-113 86 Stockholm Sweden. Verfügbar unter URL: https://www.mentimeter.com/ (Stand: 07.10.2021).

[Microsoft® Excel® 2019] Microsoft Corporation (2019): "Microsoft® Excel®. Microsoft Office Home & Business 2019."

[Microsoft® Visio® 2021] Microsoft Corporation (2021): "Microsoft® Visio® 2021 MSO. Visio Standard. Version 2308 Build 16.0.16731.20052, 64 Bit.

[MS12] McKeen, James D. / Smith, Heather A. (2012): "Effective Application Maintenance." In: Communication of the Association for Information Systems 2012 Nr. 5 Vol. 30. pp. 73-82. [DOI: 10.17705/1CAIS.03005].

[PP11] Plewan, Hans-Jürgen / Poensgen, Benjamin (2011): Produktive Softwareentwicklung. Bewertung und Verbesserung von Produktivität und Qualität in der Praxis. 1. Auflage . Heidelberg, Germany: dpunkt. verlag. Juli 2011. [ISBN: 978-3-89864-686-4].

[PR18] Patzak, Gerold / Rattay, Günter (2018): PROJEKTMANAGEMENT. Projekte, Projektportfolios, Programme und projektorientierte Unternehmen. 7., Auflage. Wien: Linde Verlag Ges.m.b.H. [ISBN 978-3-7143-0321-6].

[Pu78] Putnam, Lawrence H. (1978): „A General Empirical Solution to the Macro Software Sizing and Estimating Problem.“ In: IEEE Transactions on Software Engineering, Vol. SE-4, Nr. 4, July 1978 . pp. 345-361. [DOI: 10.1109/TSE.1978.231521].

[SL01] Schmidt, Roy / Lyytinen, Kalle / Keil, Mark / Cule, Paul (2001): „Identifying Software Project Risks: An International Delphi Study.“ In: Journal of Management Information Systems 2001 Nr. 4 Vol. 17. pp. 5 - 36.

[So16] Sommerville, Ian (2016): Software Engineering. Tenth Edition. Boston, Columbus, Indianapolis u. a.: PEARSON. Harlow: Pearson Education Limited. 2016. [ISBN-13: 978-1-292-09613-1].