WWW.IJRPR.COM

International Journal of Research Publication and Reviews

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

Cost Potentials of Platforms for Software Engineering

David Kuhlen*

**IU International University of Applied Sciences, Waterloohain 9, 22769 Hamburg, Germany david.kuhlen@iu.org*
DOI : https://doi.org/10.55248/gengpi.5.0724.1815

A B S T R A C T

The demand on software is actually high. Therefore, companies need to focus on the art of programming. The idea of treating the development of programs as an act of engineering is currently interesting, even if it is a bit unpopular today. In this context, platforms play a central role. In the past multiple best practices became formulated which aim facilitate the collaboration between software developers. This paper aims on reviewing the best practices and challenges today in order to elaborate the potentials of platforms to decrease costs in software development.

Keywords: Software Engineering, Platform

Introduction

In order to evaluate the potentials that could be offered by platforms on modern software engineering, the evolution of software engineering has to be analyzed. The current art of software development is a critical success factor in multiple companies. Especially the speed of the software development determines its success Herbsleb, Mockus, Finholt, and Grinter (2001). Obviously, the popularity of the software industry increased (Kumar, Ji, Sethi, & Yeh, 2007, p. 133) together with an increased demand for software solutions Ruiz, Ramos, and Toro (2006). Multiple IT-Solutions will be produced, not only in the software industry. IT-Solutions been developed are programs as well as scripts, websites, workflows, configurations and so on. All these IT-Solutions are the outcome of a production process.

The production of software is, as explained by Saleem and Burney, “one of the profitable and demanding best ventures of business globally since the last five decades” (Saleem & Burney, 2019, p. 145). In order to develop software in an productive manner, the production became treated as an engineering discipline, called software engineering Sommerville (2016). By using the practices of software engineering, companies seek to improve the correctness and efficiency of the development (Heaton & Carver, 2015, p. 2). Especially the reduction of the delivery time is important as requirements frequently change (Schmidt, Lyytinen, Keil, &Cule, 2001, p. 19). In the last few years, the way, how software is developed, has been changed dramatically. However, looking on the way of building IT-Solutions as an act of production is already not attractive (Herzwurm, Mellis, &Schmolling, 1994, p. 18). Herzwurm, Mellis and Schmolling argued in 1994 that viewing on the development of software as a production does not lead to further insights on how the productivity could be increased (Herzwurm et al., 1994, p. 18). Heaton and Carver argued in 2015 that there is still a potential for increased use of the practices of software engineering in order to improve its productivity (Heaton & Carver, 2015, p. 2). Often the development process is seen as a creative act. Of course, development requires creativity. However, in order to develop IT-Solutions successfully, the producer needs to work efficiently and planned. Like the construction of a building or a machinery, the construction of software requires a smart plan too. Different parts of the software need to be developed in a defined order. Furthermore, within the construction of components, the developers need to think about possible axis of change (Fayad& Cline, 1996, p. 59) . Customers demanding a solution for their specific problem (Balzert, 2008, p. 157). Compared to other sectors, in the software development low thresholds (Porter (1985)) make it easy for competitors to enter the market (Balzert, 2008, p. 185). Even complex software solutions could be recreated by low-cost competitors (Balzert, 2008, p. 185). The objective of this paper is to elaborate the potential benefits, offered by platforms on software engineering.

Related Work

A manager for software development needs to make sure that its production is economically (Balzert, 2008, p. 148). The attractiveness of a software product will increase if more customers use the product von Engelhardt (2006). The ability of software to be recombined is an advantage in the production (von Engelhardt, 2006, p. 17). Software development has the image to be too slow, creating too little quality and to be too expensive (Armour, 2014, p. 42). The risk to exceed the scheduled budget and time frame arises in practice regularly (Cusumano, 1989, p. 2). Project managers struggle to assess the progression of a software development (Balzert, 2008, p. 37). Developers risk enriching their developed functions (Balzert, 2008, p. 17). Sometimes, software developers being in enamored to their source code, spending too much time to bring their outcome to perfection, even if it is subsidiary (Balzert, 2008, p. 207). Limitations, like high costs, intellectual property rights or standards give competitive advantages for companies

(Carr, 2003, p. 6). If there exists a software solution which fulfils a specific need, it is often cheaper to buy it rather then developing a new solution (Carr, 2003, p. 8). The optimism of programmers has an impact on their estimation (Brooks, 1995, p. 17). To employ unproductive staff is unfair towards the productive staff, because they need to rescue where the unproductive failed (Balzert, 2008, p. 93). Plewan and Poensgen criticize the software industry to lack on general accepted methods and best practices (Plewan&Poensgen, 2011, p. 17). In order to be a professional discipline, the software industry (1.) needs to have uniform best practices and methods, (2.) needs to obtain reliable results in a good quality, (3.) have admitted values (code of ethics) and (4.) needs to have standards that define the skills, required by professionals (Plewan&Poensgen, 2011, p. 17). The software industry struggles on multiple software developers, setting up their own standards Putnam (1978). The increased demand of users for software functionality pressurizes software producers (Boehm &Papaccio, 1988, p. 1465 - 1466). In order to successful realize software projects, software developers and business experts needs to understand each other Heise, Strecker, Frank, and Jung (2008). Business process reengineering cause obstacles, that has to be managed, in order to be successful (Hammer &Champy, 1993, p. 212). As Putnam explained, the development could be views on the basis of a life cycle pattern Putnam (1977). Each software manager has to manage an area of conflict that consists of time, quality, costs and complexity (Balzert, 2008, p. 212). Like classical manufacturing seeks to improve the productivity, viewing software development as an engineering discipline leads the focus on the question, how to increase the productivity with regard to input and output Lu06]. The competition in the software market forces software producers to deliver more functionality faster to the customers Kumar et al. (2007). If this pressure leads to less profit, e.g. because the more functionality was delivered in an unproductive manner, the software producer would fail Kumar et al. (2007).

Method

In order to determine the potentials of platforms in software-engineering, we first need a collection of today's challenges. These challenges come from new requirements, that has to be solved by software development project. The next step is to determine the current setting of best practices, used in software engineering. Next to the programming, software producers have to analyze and design software solutions (Balzert, 2008, p. 21). Beyond the development of new software solutions, software producers have to maintain the existing systems. Within this maintenance, software systems become changed, in order to solve bugs or to fulfill change requests (Sommerville, 2016, p. 270). In order to be useful, the specification documents need to have a clear structure, to be precise and unambiguous in their description, to be complete and to be adaptable (Pohl & Rupp, 2015, p. 45 f.). Within the development, the software producer has to use best practices in programming. These best practices seek to improve the maintainability of a program.

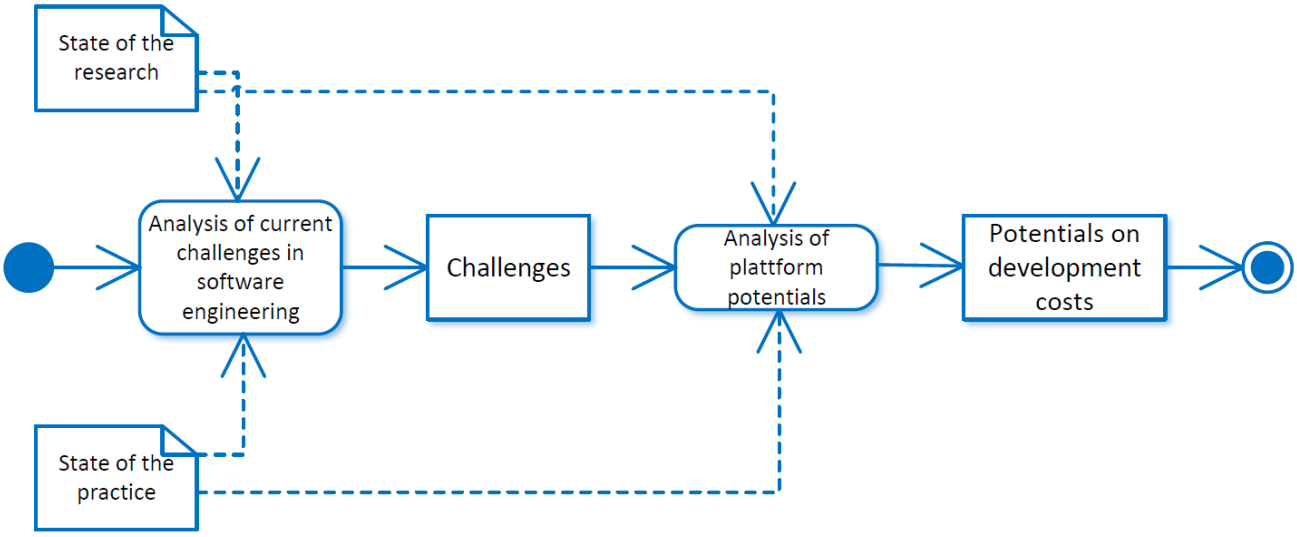

Figure 1: Illustration of the method, own illustration created with Microsoft Visio (2021)

Figure 1 illustrates the procedure which will be used in this work. The derivation of cost potentials based on logical assumptions. The basis of these assumptions is an analysis of the state of the research and experience from practice.

Results

In economy, multiple companies use platforms in order to decrease their costs. Decreasing costs is always a powerful motivation for using new technologies. The potentials offered by platforms to decrease the costs result mainly from reasons displayed in figure 2.

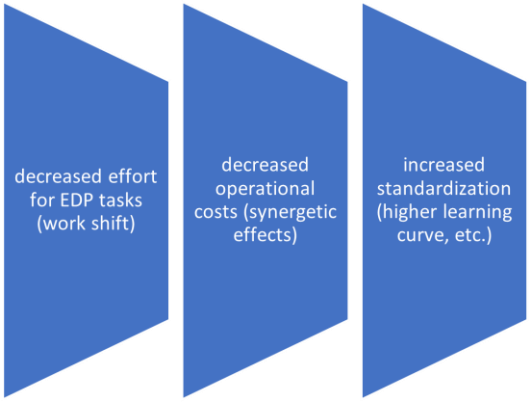

Figure 2: Illustration of cost effects, own illustration created with Microsoft Word (2019)

In economy, collaborative platforms are implemented in order to simplify the communication between a supplier and its client Lindquist, Berglund, and Johannesson (2008). As displayed in figure 3 this effect plays an important role in software engineering too. Collaborative platforms, described by Lindquist et al. (2008), allow to shift the effort, which is necessary to enter, transform and to submit data between different partners in the supply chain. In the context of software engineering, they offer the potential to simplify the communication in requirements engineering. For example, companies build platforms, that allow their suppliers to enter detailed information about products, tasks, orders or requests Lindquist et al. (2008). This allows companies, which have a strong market position, to shift their internal effort to their suppliers.

Furthermore, the use of platforms decreases the operational costs. By using a standard platform, multiple companies save capital (invested working hours, hardware capacity etc.). In general, Stefanou confirms the potential of decreasing costs by the use of a common IT-Architecture (Stefanou, 2001, p. 210). This is a powerful reason for using cloud services. As shown in figure 3, the integration of cloud native aspects may facilitate the construction of software architectures. By using a standard platform, the user does not have to set up his own platform. As McKeen explained, the costs of maintenance could consume the whole budget (McKeen & Smith, 2012, p. 73). Synergetic effects allow the reusage of standard software components, which are a part of the platform and share the invested labor on a wide range of clients / users. This simplifies maintenance, too. Additionally, in the field of software engineering, where 80 percent of the costs come from human resources (Hu, Plant, & Hertz, 1998, p. 149), less effort directly decreases the costs.

Platforms offer a strong potential to standardize procedures. Figure 3 shows the benefits of platforms on the development and the deployment of software products. The usage of a common software interface aligns all users to follow the same process. For example, the use of Microsoft DevOps (n.d.), as a standard platform in software development, facilitates all developers to follow the same development procedure. Software producers have the possibility to customize their platform, in order to make sure that individual processes will be followed. A unique platform simplifies the onboarding of people. This leads to a decrease of training costs.

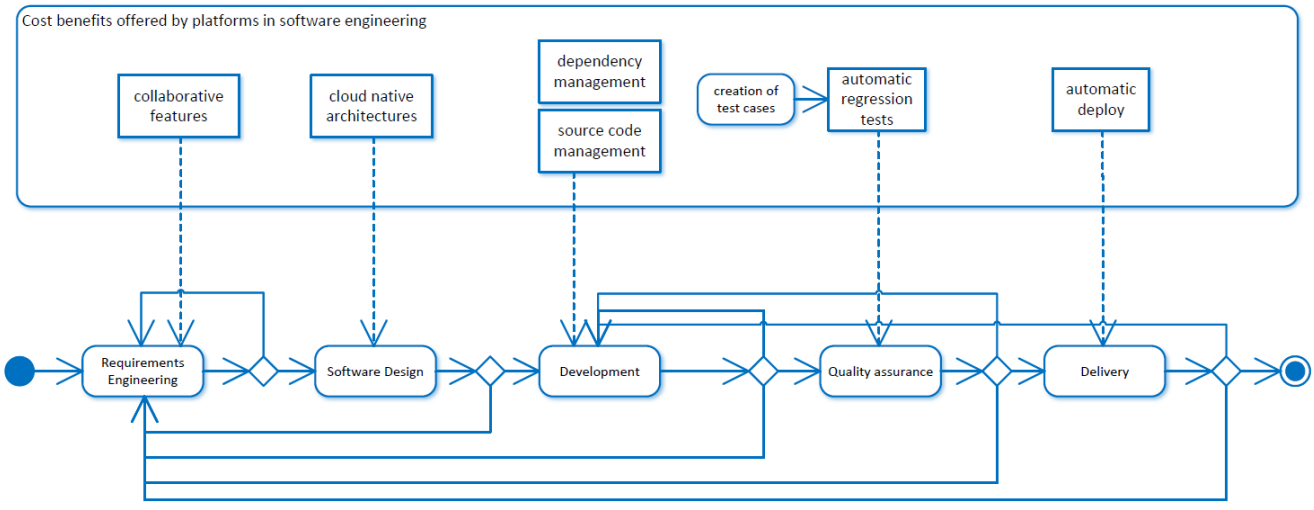

Figure 3: Contribution of platforms to achieve cost benefits in software production, own illustration created with Microsoft Visio (2021)

In total, platforms have an impact on the cost situation of companies. By using platforms, companies have a possibility to decrease their costs. However, it is important that the architecture of the platform is valid, in order to realize sustainable benefits Lindquist et al. (2008).

Conclusion

The discipline of software engineering requires the collaboration of different stakeholders like software developers, the exchange of information between these stakeholders and the management of the production process. platforms offer a strong potential to decrease costs, especially for information interchange. In the context of software engineering, the usage of platforms could be beneficial to decrease the development costs.

In further investigations, the potentials of platforms for software-engineering on the process controlling might be an interesting topic. In order to optimize processes and to its outcome, business uses and analyzes data. Platforms offer a central possibility, to collect and manage date on the software development and to simplify the collaboration. Data analytic projects seek to answer questions on the basis of existing data material, to improve decision making. Decision making is driven by data analytics. Modern business needs to get the right information at the right moment to choose between multiple alternative decision options. However, before the management needs to decide, the circumstances of a decision are unknown, in some part.

References

Armour, P. G. (2014, January). Estimation Is Not Evil. Communications of the ACM, 57(1), 42–43. (DOI: 10.1145/2542505)

Balzert, H. (2008). Lehrbuch der Softwaretechnik. Softwaremanagement (2. Auflage). Heidelberg: Spektrum Akademischer Verlag. (ISBN: 978-3-8274-1161-7)

Boehm, B. W., & Papaccio, P. N. (1988, Oktober). Understanding and Controlling Software Cost. IEEE Transactions on Software Engineering, 14(10), 1462 - 1477. (DOI: 10.1109/32.6191)

Brooks, F. P. J. (1995). The Mythical Man-Month. Essay on Software Engineering. Anniversary edition with four new chapters (No. 20th Anniversary Edition). Boston, San Francisco, New York and others: AddisonWesley Longman, Inc. (ISBN: 0-201-83595-9)

Carr, N. G. (2003, May). IT Doesn´t Matter. Harvard Business Review, 81(5), 41–49. (Reprint R0305B; HBR OnPoint 3566.)

Microsoft Corporation, Microsoft DevOps (n.d.). Azure DevOps. (Information available at microsoft.com: https://azure.microsoft

.com/de-de/products/devops/)

Microsoft Corporation, Microsoft Word (2019). Microsoft© Word. Office Home Business 2019. (Version 2212)

Microsoft Corporation, Microsoft Visio (2021). Microsoft© Visio© 2021 MSO. Visio Standard. (Version 2308 Build 16.0.16731.20052, 64 Bit)

Cusumano, M. A. (1989, December). Factory Concepts and Practices in Software Development: An Historical Overview (Working Paper No. 3095-89 BPS). Cambridge, Massachusetts: Massachusetts Institute of Technology. Center for Computational Research in Economics and Management Science.

Fayad, M., & Cline, M. P. (1996, October). Aspects of Software Adaptability. Communications of the ACM, 39(10), 58–59. (DOI: 10.1145/236156.236170)

Hammer, M., &Champy, J. (1993). Reengineering the Corporation. A Manifesto for Business Revolution. New York, USA: HarperBusiness A Division of HarperCollins Publishers. (ISBN: 0-88730-687-X)

Heaton, D., & Carver, J. C. (2015, November). Claims About the Use of Software Engineering Practices in Science: A Systematic Literature Review. Information and Software Technology, 67, 207–219. (Elsevier. DOI: 10.1016/j.infsof.2015.07.011)

Heise, D., Strecker, S., Frank, U., & Jung, J. (2008, February). Erweiterung einer Unternehmensmodellierungsmethode zur Unterstützung des IT-Controllings. In M. Bichler et al. (Eds.), Multikonferenz Wirtschaftsinformatik 2008. GITO.

Herbsleb, J. D., Mockus, A., Finholt, T. A., & Grinter, R. E. (2001, May). 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). Los Alamitos, Washington, Brussels, Tokyo: IEEE Computer Society. (DOI: 10.1109/ICSE.2001.919083)

Herzwurm, G., Mellis, W., & Schmolling, K. (1994). Software Factory – Ein Statusbericht. HMD Praxis der Wirtschaftsinformatik (HMD 180), 8 – 19.

Hu, Q., Plant, R. T., & Hertz, D. B. (1998, December). Software Cost Estimation Using Economic Production Models. Journal of Management Information Systems, 15(1), 143–163. (Taylor & Francis. DOI: 10.1080/07421222.1998.11518200)

Kumar, S., Ji, Y., Sethi, S. P., & Yeh, D. H. (2007, December 2006). Dynamic Optimization of Software Enhancement Effort. In A. P. Sinha & R. Venkataraman (Eds.), Proceedings of 16th Annual Workshop on Information Technologies & Systems (WITS). Milwaukee, Wisconsin: SSRN. (DOI: 10.2139/ssrn.1026842)

Lindquist, A., Berglund, F., &Johannesson, H. (2008, March). Supplier Integration and Communication Strategies in Collaborative Platform Development. SAGE Publications, 16(1). (DOI: 10.1177/1063293X07084639)

McKeen, J. D., & Smith, H. A. (2012, March). Effective Application Maintenance. Communication of the Association for Information Systems (CAIS), 30(5), 73–82. (DOI: 10.17705/1CAIS.03005)

Plewan, H.-J., & Poensgen, B. (2011). Produktive Softwareentwicklung. Bewertung und Verbesserung von Produktivität und Qualität in der Praxis (1. Auflage). Heidelberg: dpunkt.verlag GmbH. (ISBN: 978-3-89864-686-4)

Pohl, K., & Rupp, C. (2015). Basiswissen Requirements Engineering. Aus- und Weiterbildung nach IREB-Standard zum Certified Professional for Requirements Engineering Foundation Level (4. überarbeitete Auflage). Heidelberg: dpunkt.verlag GmbH. (ISBN: 978-3-86490-283-3)

Porter, M. E. (1985). Competitive advantage. Creating and sustaining superior performance. THE FREE PRESS.

Putnam, L. H. (1977). The Influence Of The Time - Difficulty Factor In Large Scale Software Development. In Proceedings of the 15th IEEE Computer Society International Conference. COMPCON Fall'77 (pp. 348–353). Long Beach, California: IEEE Computer Society. (DOI: 10.1109/CMPCON.1977.680858)

Putnam, L. H. (1978, July). A General Empirical Solution to the Macro Software Sizing and Estimating Problem. IEEE Transactions on Software Engineering, SE-4(4), 345–361. (DOI: 10.1109/TSE.1978.231521)

Ruiz, M., Ramos, I., & Toro, M. (2006, February). Software Process Dynamics: Modeling, Simulation and Improvement. In S. T. Acuña& M. I. Sánchez-Segura (Eds.), New Trends in Software Process Modeling (pp. 21–56). World Scientific Publishing Co Pte Ltd. (DOI: 10.1142/9789812774460_0002)

Saleem, H., & Burney, S. M. A. (2019, January). Imposing Software Traceability and Configuration Management for Change Tolerance in Software Production. IJCSNS International Journal of Computer Science and Network Security, 19(1), 145–154.

Schmidt, R., Lyytinen, K., Keil, M., &Cule, P. (2001). Identifying Software Project Risks: An International Delphi Study. Journal of Management Information Systems, 17(4), 5 – 36. (Taylor & Francis. DOI: 10.1080/07421222.2001.11045662)

Sommerville, I. (2016). Software Engineering (Tenth Edition). Harlow, England: Pearson Education Limited. (ISBN: 978-1-292-09613-1)

Stefanou, C. J. (2001). A framework for the ex-ante evaluation of ERP software. European Journal of Information Systems, 10(4), 204 – 215. (Taylor & Francis. DOI: 10.1057/palgrave.ejis.3000407)

von Engelhardt, S. (2006). Die ökonomischen Eigenschaften von Software. Jenaer Schriften zur Wirtschaftswissenschaft, Arbeits- und Diskussionspapier (14). (Friedrich-Schiller-Universität Jena, Wirtschaftswissenschaftliche Fakultät. ISSN: 1611-1311