IT SYSTEMS DELIVERY IN THE DIGITAL AGE:

AGILE, DEVOPS AND BEYOND

Alena Buchalcevova, Michal Doležel

Department of Information Technologies
Faculty of Informatics and Statistics
University of Economics, Prague
alena.buchalcevova@vse.cz, michal.dolezel@vse.cz

Keywords

Agile Development; DevOps; Digital Transformation; Multi-speed IT; Software development methods

Abstract

Today, we live in the digital age. However, the original plan-driven methods of IT and software systems delivery were proposed several decades ago, being coined in an entirely different context. In this position paper we review recent advancements in the IT and software systems delivery method space, and discuss and categorize concepts such as method tailoring, method hybridization, large-scale agile and multi-speed IT. From a method-centered perspective, we outline possible future directions towards making digital transformation happen.

1. Introduction

Many people believe that "software is eating the world" (Andreessen, 2011). This expression highlights the role of software in modern society, which has strongly been influenced by the enormous opportunities that innovative software solutions presently offer. Put differently, software is a major driving force behind digital transformation, a technology-driven continuous change process focused on companies and entire society. In general, digital transformation is about adopting disruptive technologies to increase productivity, value creation, and the social welfare (Ebert & Duarte, 2018). Although digital transformation is already underway, it has not been proceeding at the same pace everywhere. According to the McKinsey Global Institute's Industry Digitization Index (McKinsey, 2016), Europe is currently operating at 12% of its digital potential, whereas the USA at 18%.

CIOs and their IT teams are in a unique position to drive this transformation. However, many of these initiatives have fallen behind due to implementation challenges (Ismail, 2018). As confirmed by the long-standing results of the Standish Group's CHAOS Reports, only one third of all application development projects satisfy the criteria of successfulness (Standish Group, 2015).

While digital transformation does not necessarily require development of radically new software technologies, it has given rise to new software technology applications. As a consequence, complexity and scale of technological solutions have increased substantially along with placing time to market, quality, and affordability at the forefront (Ebert & Duarte, 2018). An effective software development management, reusability, and requirements engineering methods, techniques, and tools are needed to address these issues of IT systems delivery.

The aim of this position paper is to review current approaches to IT systems delivery, categorize them and outline possible future directions towards making digital transformation happen. In this paper, we examine the problem from a dual perspective. Firstly, we begin by categorizing modern software delivery methods in a generic sense. Then, we deal with the impact of these methods on the whole enterprise.

2. From Waterfall to Agile, Lean, DevOps and Hybrid Methods

2.1. Plan-driven Methods

Traditional plan-driven development approaches emphasize predictability and stability in a project (Boehm & Turner, 2003). These approaches are known as "Waterfall" as they use a downward-flowing stage model for developing software requiring substantial upfront design (Mahadevan, Ketinger, & Meservy, 2015). Feedback is limited between stages of the system development lifecycle (SDLC), including specifications, development, testing, and implementation (Boehm, 1988).

While the traditional plan-driven software development methods do not scale to the challenges brought by digital transformation, agile and lean approaches are a major step in that direction. Roots of these approaches as well as their main principles are outlined further.

2.2. Agile Methods

Agile methods have now become the mainstream for software development worldwide. Formally, they were introduced through a set of four core values and 12 principles laid out in the Agile Manifesto (Beck et al., 2001). At their inception, agile methods were coined by a group of "organizational anarchists", who strived to uncover "better ways of developing software". In essence, this group of people believed that software should be developed differently from the then mainstream norms of software engineering (Doležel, 2018). However, many people put much less emphasis on the ideological dimension of the problem nowadays, while prioritizing the pragmatic benefits of agile methods.

Looking at their pragmatic aspects, agile methods have been proposed as a way to avoid project failures (Dybå & Dingsøyr, 2008). The risk of project failure is reduced each time a software increment is delivered, since the highest priority requirements are selected for development during each increment and each increment is used to gather client and user feedback. The increments are delivered regularly and each comprises a carefully defined fragment of the overall development effort. This contrasts with the plan-driven methods in which risks progressively rise until the product is handed over at the end of the project. On these grounds, there is an evidence that agile methods can improve both software development productivity and product quality (Dybå & Dingsøyr, 2009).

2.3. Lean Methods

Lean software development has its roots in Lean manufacturing and Toyota Production System. Lean is about "doing more with less" by ideally producing "the right things, at the right time and in the right place", as stated in the book *The machine that changed the world* (Womack, Jones, & Roos, 1990). Lean software development dates to the early 1990s. At the beginning, Lean software development was predominantly considered as one of Agile methods (Dybå & Dingsøyr, 2008). Later, it has become a method of its own. However, there are several interpretations of Lean in software development, some of them are explained by Rodríguez et al. (2019). The main benefits of Lean software development include: increased customer satisfaction; increased productivity; decreased lead and development times; improved progress transparency; identification of problems' root causes; identification of bottlenecks (Rodríguez et al., 2019). The most popular among Lean techniques is Kanban. In short, Kanban uses a board visualizing each activity in the development cycle in a column, and sticky notes representing user stories or tasks from the product backlog. The number of items associated with every activity is limited by establishing work in progress (WIP) limits. Importantly, there is a clear conceptual link between Lean software development, DevOps and scaled agile methods. When examining the role of Lean software development nowadays, one should therefore attempt to see beyond the strict limits of the individual methods. In that vein, combining different methods and approaches to software development seems to be a natural step today. More formally, this is often labeled as method tailoring.

2.4. Method Tailoring and Hybridization

It has been known for quite a long time that due to the differences in project characteristics, environmental contexts, and developer characteristics no particular software development method will ever be a "silver bullet" (Brooks, 1987). As a result, software development methods are rarely implemented in a "by book" manner (Dittrich, 2016). Instead of rigidly following the method prescriptions, selecting, adapting and combining software practices is a reality.

Commonly, the term "method tailoring" is used as an umbrella concept to label such strategies. Method tailoring comprises two main approaches, contingency-based method selection and method engineering (Bass, 2016). Contingency-based method selection is about selecting an appropriate software development method dependent on the project context. In contrast, using the method engineering approach, development teams construct a bespoke new process using method fragments (Fitzgerald, Hartnett, & Conboy, 2006).

Conceptually, the relationship between tailoring and hybrid software development methods should be clarified. In general, "A hybrid software development approach is any combination of agile and/or traditional (plan-driven or approaches that an organizational unit adopts and customizes to its own context needs)" (Marco Kuhrmann et al., 2018). However, we believe that two very different classes of hybridization efforts deserve one' attention: (i) hybridization occurring within the tents of the Agile camp (e.g. ScrumXP), and (ii) hybridization occurring across the Agile vs. Plan-driven divide (Doležel, 2018). These two classes of hybridization are described further.

2.4.1. Hybridization within the Agile Method Space

Although early agile adopters stated a very strict and orthodox approach to the method usage, at present some agile methodologists predominantly view the agile practices as a "toolkit" to be applied as needed in a variety of project environments (Tripp & Armstrong, 2014). A recent systematic literature review of empirical agile tailoring research papers suggests that the method engineering approach is more popular with project teams (Campanelli, Camilo, & Parreiras, 2015)

and can be related to stakeholders, project life cycle, project organization and knowledge building (Kalus & Kuhrmann, 2013). According to the last State of Agile Development survey (VersionOne, 2018), combined agile methods together account for 28% of the total usage. One of the first combined agile methods was the union of Scrum and Extreme programming (XP), which makes use of the Scrum's focus on project management and XP's focus on software engineering (Fitzgerald et al., 2006). This combined method still keeps a relatively high share (6%) in method usage (VersionOne, 2018). Being another example, ScrumBan combines Scrum and Kanban with its share recently still increasing and reaching 8% (VersionOne, 2018). The examples of an agile method tailoring usage in practice can be found in (Conboy & Fitzgerald, 2010; Fitzgerald et al., 2006).

2.4.2. Hybridization among Agile and Plan-driven Methods

According to West (2011), hybrid agile and plan-driven methods are the reality in most agile implementations. Scrum adoption is limited to the development-team level, whereas compliance requirements are another factor driving hybrid approaches, as they call for strong governance processes before and after the development. The term “Water-Scrum-Fall” has been coined and West (2011) hypothesized that hybrid development methods would become the standard. Based on this idea, the HELENA study has been conducted aimed at determining the current state of practice in using and combining the multitude of available software development approaches - be it agile and traditional ones (Kuhrmann et al., 2017). The results of this study conducted in more than 55 countries confirmed that the combination of different software and system development approaches has become the reality and is found independently of the company size, the respective industry sector or the actual region (Kuhrmann et al., 2018).

2.5. DevOps

The software development methods described above represent only one dimension of the problem how IT systems are implemented within the industry. The remaining issue is how the systems are deployed, supported, monitored, and later decommissioned. An important trend in this domain is DevOps.

The concept of DevOps emerged about a decade ago, still attaining an increasing interest both from practitioners and researchers. DevOps is an abbreviation of Dev (development – software development) and Ops (operations – software operations). Many practitioners view DevOps as a logical and natural extension of agile software development ideas (Jabbari, Nauman, & Tanveer, 2016). In essence, DevOps promotes those practices that are making software development and operations closely integrated with each other, emphasizing a frequent feedback from both sides. In that sense, DevOps is commonly associated with a shift in work responsibilities and with a change in work patterns related to IT professionals working in the IT systems delivery domain.

Pragmatically speaking, DevOps can be viewed as “a development methodology aimed at bridging the gap between Development (Dev) and Operations, emphasizing communication and collaboration, continuous integration, quality assurance and delivery with automated deployment utilizing a set of development practices" (Jabbari et al., 2016). Taken more broadly, however, DevOps can also be associated with four basic elements, providing a sort of loose prescription on how DevOps principles can be put into reality. That is, by introducing specific Culture elements, Automation practices, Measurement principles, and by supporting information and knowledge Sharing (altogether abbreviated as CAMS). Taking such an approach, one is to build a foundation where modern software development and operations approaches can thrive alongside each other, supporting both current and future business needs. The above CAMS principles can then serve as a

conceptual guideline in a sense of highlighting the foremost priorities for the DevOps implementation programs, whether they are focused on particular development teams or on the corporate level.

Being not always apparent, an important part of the DevOps efforts is building a specific DevOps culture. In that sense, Sánchez-Gordón & Colomo-Palacios (2018) remind that "beyond the tool chain, DevOps is [predominantly] a culture shift". In addition, many practitioners call for making the business component in DevOps more explicit, which leads to promoting the term BizDevOps (Erich, Amrit, & Daneva, 2017). Others would like to see software testing and quality assurance at the same place, resulting in the idea of DevTestOps (Scheaffer, Ravichandran, & Martins, 2018).

3. Which Way to the Digital Age?

An illustrative overview of current approaches to IT systems delivery is pictured in Figure 1. The root paths of current approaches, e.g waterfall, agile and lean are depicted at the bottom. These methods are tailored in a way that results in a hybrid nature of the formerly autonomous methods; development is widened to operations, together forming DevOps. All these approaches are primarily followed at the IT organization level – within particular development teams.

Looking at Figure 1, the crucial question to ask is: Which direction should one take today? Nevertheless, we are hesitant to give a universal answer, because this would largely depend on the particular set of context-dependent factors of one's enterprise. In any case, we can acknowledge the current trend, which seems to be to scale agile and lean approaches to larger projects (and to the whole enterprise). This type of scaling is described in the next section.

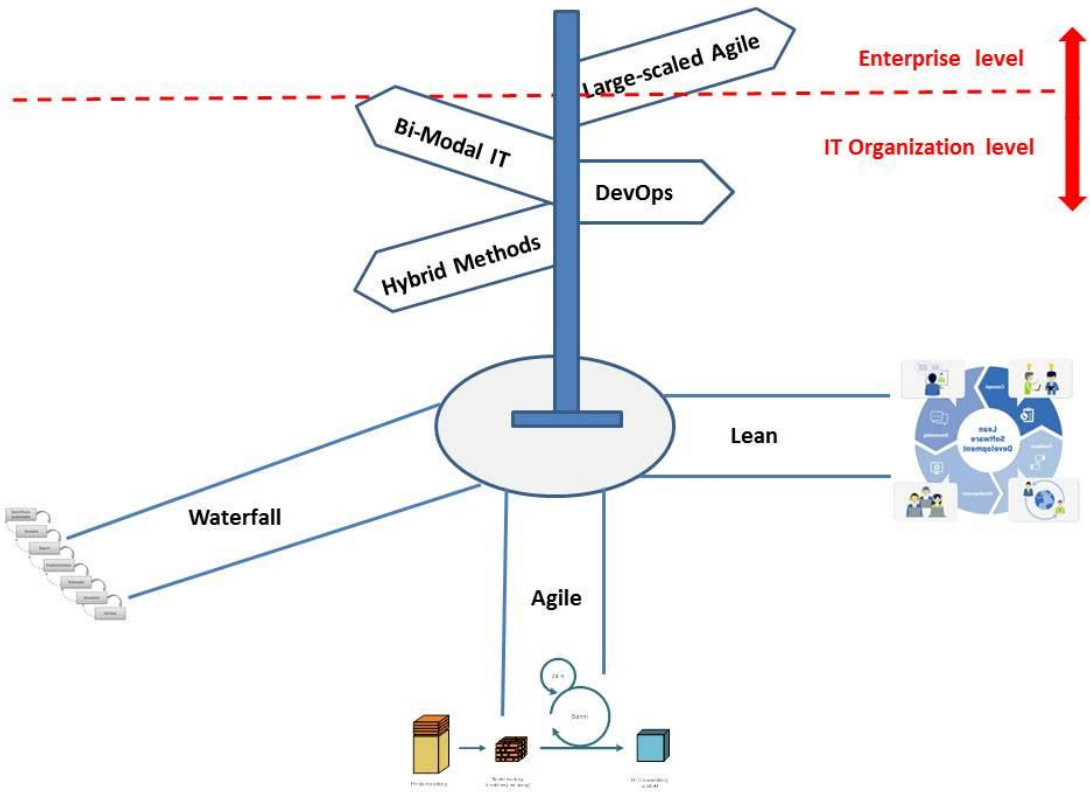

Figure 1. Overview of IT Systems Development Methods and Organizational Structures

4. Scaling Methods to Enterprise Level and New IT Organizational Structures

4.1. Large-scale Agile Methods

Although agile methods were originally designed to be used in small, single team projects (Boehm & Turner, 2005), their benefits have made them attractive also for larger projects and larger companies (Dikert, Paasivaara, & Lassenius, 2016). Compared to small projects, larger ones are characterized by the need for an additional coordination, which makes agile method implementation more difficult (Bick, Spohrer, Hoda, Scheerer, & Heinzl, 2018; Dybå & Dingsøyr, 2009). Large-scale agile involves additional concerns in handling an inter-team coordination and interfacing with other organizational units, such as human resources, marketing and sales, and product management. In addition, large scale may cause users and other stakeholders to become distant from the development teams (Dikert et al., 2016). Despite such known problems related to large-scale agile, there is an industry trend towards adopting agile methods in-the-large (Dingsøyr & Moe, 2014; VersionOne, 2018). A number of scaled agile methods and frameworks are in place like the Discipline Agile Delivery (DAD), Large-scale Scrum (LeSS), Scaled Agile Framework (SAFe), Scrum@Scale, and Nexus (Kalenda, Hyna, & Rossi, 2018).

4.2. Two-speed/multi-speed IT and New IT Organizational Structures

Given that "DevOps presents challenges for the existing IT function and organizational structure" (Wiedemann, Wiesche, Gewald, & Krcmar, 2018), it is important to understand how the DevOps principles can be combined with traditional IT organizations and departments. In fact, due to complex and rigid IT infrastructures and inflexible hierarchical organizational silos in business and IT, companies are often unable to achieve the level of agility and flexibility needed for conducting digital transformation. In some cases, digital transformation in traditional organizations results in two different modes of IT operations, i.e. "two-speed IT" or "bi-modal IT" (Haffke, Kalgovas, & Benlian, 2017). This model consists of two components, a fast customer-facing and slow business-oriented IT organization. The first IT component is established in order to react to rapidly changing customer needs. The goal of this mode is to explore new IT capabilities and to innovate. The second IT component is established to respond to the need of companies keeping or gradually decoupling the 'legacy IT' within the established IT infrastructure and IT organization. This part of the IT organization works in longer cycles (i.e. at a "lower speed"), as it commonly runs large legacy systems, which cannot be changed or turned into a new digital architecture easily. Alternatively, such a turn would pose a risk. Therefore, the goal of this latter mode is to provide stability for existing IT operations (Horlach, Drews, & Schirmer, 2016).

Apart from the different speed modes, both parts operate with different organizational structures and methods. Hence, many companies implement a "bimodal IT" organization with different governance mechanisms, processes and organizational structures to respond to this duopoly of speed (Horlach et al., 2016). Broadly speaking, there is an increasing interest in the changing nature of IT departments and IT organizations. Companies seem to transform their formally defined IT organizational entities into less formal and more pervasive ones, responding to the needs of digital transformation (Peppard, 2018). In so doing, however, companies also face the danger of losing control over their IT landscapes. Such an unwanted situation is commonly referred to as the problem of "Shadow IT" (Huber, Zimmermann, Rentrop, & Felden, 2017).

5. Conclusion

Today, we live in exciting times when IT-driven solutions transform our ways of living and working. Looking on the bright side of this trend primarily, we argue that IT and software professionals should be ready to accept their pivotal role in all these changes. To support digital transformation, a number of important concepts, tools and techniques have been introduced in the recent years. Their common denominator is the need to promptly respond to the changing business needs, fulfilling the vision of the world being "eaten" by modern software.

In this position paper, we covered mostly innovative software development methods and new organizational models of IT operations.

As the traditional methodologies do not scale to the challenges brought by digital transformation, agile methods have become the mainstream software development methods in the world. Thanks to their pragmatic benefits and ability to avoid project failures, they have become the innovative means of digital transformation in practice. Getting closer to the reality, a method tailoring approaches have emerged following the trend of adapting and combining software practices within the agile space, scaling agile methods or heading towards the traditional methods.

Responding to the other side of digital transformation, new organizational models of IT operations have emerged. The concept of DevOps bridges the gap between Development and Operations and makes the business component more explicit. Whereas the two-speed IT model makes it possible for a typically rigid company to react fast to rapidly changing customer needs and at the same time give the organization a possibility to preserve its stability and take time to carry out large digital transformation changes. Overall, companies head towards transforming their formally defined IT organizational entities into less formal ones, responding to the needs of digital transformation.

Clearly, there are many other important topics that are related to innovative models of software and systems delivery and were not discussed here. Among these we include, for example, the software development and operations specifics of SMACIT (social, mobile, analytics, cloud and Internet of things) (Moloney et al., 2017), the role that crowdsourcing starts to play in software development and testing, or the need to understand how successful engineering managers carry out their job duties in modern enterprises (Kalliamvakou et al., 2017). In our view, especially Internet of things has a potential to truly revolutionize a number of industry sectors. However, the need of selecting optimal technology from quite a diverse set (Maryska, Doucek, Nedomova, & Sladek, 2018) also clearly foresees the need of deploying "fitting" IT systems delivery models.

We conclude by reiterating the words of George Westerman from MIT: "When digital transformation is done right, it's like a caterpillar turning into a butterfly, but when done wrong, all you have is a really fast caterpillar" (MIT Sloan, 2014). Hence, it does matter whether and how we support the transformational processes by providing and managing innovative IT means.

6. References

Andreessen, M. (2011). Why Software Is Eating the World. *Wall Street Journal*, *20*.

Bass, J. M. (2016). Artefacts and agile method tailoring in large-scale offshore software development programmes. *Information and Software Technology*, *75*, 1–16.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001). Manifesto for Agile Software Development. Retrieved March 10, 2019, from https://agilemanifesto.org/

Bick, S., Spohrer, K., Hoda, R., Scheerer, A., & Heinzl, A. (2018). Coordination Challenges in Large-Scale Software Development: A Case Study of Planning Misalignment in Hybrid Settings. *IEEE Transactions on Software*

Engineering, *44*(10), 932–950.

Boehm, B. (1988). A spiral model of software development and enhancement. *Computer*, *5*, 61–72.

Boehm, B., & Turner, R. (2003). Using risk to balance agile and plan-driven methods. *Computer*, *36*(6), 57–66.

Boehm, B., & Turner, R. (2005). Management Challenges to Implementing Agile Processes in Traditional Development Organizations. *IEEE Software*, *22*(5), 30–39.

Brooks, F. P. (1987). No Silver Bullet - Essence and Accident in Software Engineering. *IEEE Computer*, *4*, 10–19.

Campanelli, A. S., Camilo, R. D., & Parreiras, F. S. (2015). Agile methods tailoring - A systematic literature review. *Journal of Systems and Software*, *110*, 85–100.

Conboy, K., & Fitzgerald, B. (2010). Method and developer characteristics for effective agile method tailoring. *ACM Transactions on Software Engineering and Methodology*, *20*(1), 1–30.

Dikert, K., Paasivaara, M., & Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations : A systematic literature review. *The Journal of Systems & Software*, *119*, 87–108.

Dingsøyr, T., & Moe, N. B. (2014). Towards Principles of Large-Scale Agile Development. In *International Conference on Agile Software Development*. Springer, Cham.

Dittrich, Y. (2016). What does it mean to use a method? Towards a practice theory for software engineering. *Information and Software Technology*, *70*, 220–231.

Doležel, M. (2018). Possibilities of applying institutional theory in the study of hybrid software development concepts and practices. In *International Conference on Product-Focused Software Process Improvement* (pp. 441–448).

Dybå, T., & Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology*, *50*(9–10), 833–859.

Dybå, T., & Dingsøyr, T. (2009). What do we know about agile software development? *IEEE Software*, *26*(5), 6–9.

Ebert, C., & Duarte, C. H. C. (2018). Digital Transformation. *IEEE Software*, *35*(4), 16–21.

Erich, F., Amrit, C., & Daneva, M. (2017). A qualitative study of DevOps usage in practice. *Journal of Software: Evolution and Process*, *29*(6), e1885.

Fitzgerald, B., Hartnett, G., & Conboy, K. (2006). Customising agile methods to software practices at Intel Shannon. *European Journal of Information Systems*, *15*(2), 200–213.

Haffke, I., Kalgovas, B., & Benlian, A. (2017). The Transformative Role of Bimodal IT in an Era of Digital Business. In *Proceedings of the 50th Hawaii International Conference on System Sciences (2017)* (pp. 5460–5469).

Horlach, B., Drews, P., & Schirmer, I. (2016). Bimodal IT : Business-IT alignment in the age of digital transformation. In *Multikonferenz Wirtschaftsinformatik (MKWI)*.

Huber, M., Zimmermann, S., Rentrop, C., & Felden, C. (2017). Integration of Shadow IT Systems with Enterprise Systems - A Literature Review. In *PACIS* (pp. 1–13).

Ismail, N. (2018). Why IT projects continue to fail at an alarming rate. Retrieved March 10, 2019, from https://www.information-age.com/projects-continue-fail-alarming-rate-123470803/

Jabbari, R., Nauman, B. A., & Tanveer, B. (2016). What is devops? A systematic mapping study on definitions and practices. In *Proceedings of XP2016*.

Kalenda, M., Hyna, P., & Rossi, B. (2018). Scaling agile in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process*, *30*(10), e1954.

Kalliamvakou, E., Bird, C., Zimmermann, T., Begel, A., DeLine, R., & German, D. M. (2017). What Makes a Great Manager of Software Engineers? *IEEE Transactions on Software Engineering*, *45*(1), 87–106.

Kalus, G., & Kuhrmann, M. (2013). Criteria for software process tailoring: a systematic review. In *International Conference on Software and Systems Process* (pp. 171–180).

Kuhrmann, M., Diebold, P., Münch, J., Tell, P., Garousi, V., Felderer, M., … Prause, C. R. (2017). Hybrid Software and System Development in Practice:Waterfall, Scrum, and Beyond. In *Proceedings of International Conference on Software System Process*.

Kuhrmann, M., Tell, P., Klünder, J., Hebig, R., Licorish, S., & Macdonell, S. (2018). HELENA Stage 2 Results.

Retrieved May 24, 2019, from https://www.researchgate.net/publication/329246439_HELENA_Stage_2_Results/stats

Mahadevan, L., Ketinger, W. J., & Meservy, T. O. (2015). Running on hybrid: Control changes when introducing an agile methodology in a traditional "waterfall" system development environment. *Communications of the Association for Information Systems*, *36*.

Maryska, M., Doucek, P., Nedomova, L., & Sladek, P. (2018). The Energy Industry in the Czech Republic: On the Way to the Internet of Things. *Economies*, *6*(2), 36.

McKinsey. (2016). Digital Europe: Realizing the continent's potential. Retrieved March 15, 2019, from https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/digital-europe-realizing-the-continents-potential

MIT Sloan. (2014). The digital business transformation imperative. Retrieved from https://executive.mit.edu/blog/the-digital-business-transformation-imperative

Moloney, I. S., Ross, J., Beath, C., Mocker, M., Moloney, K. G., & Fonstad, N. O. (2017). How Big Old Companies Navigate Digital Transformation. *MIS Quarterly Executive*, *16*(3), 197–213.

Peppard, J. (2018). Rethinking the concept of the IS organization. *Information Systems Journal*, *28*(1), 76–103.

Sánchez-Gordón, M., & Colomo-Palacios, R. (2018). Characterizing DevOps Culture: A Systematic Literature Review. In *SPICE* (pp. 3–15).

Scheaffer, J., Ravichandran, A., & Martins, A. (2018). *The Kitty Hawk Venture: A Novel about Continuous Testing in DevOps to Support Continuous Delivery and Business Success*. San Francisco, CA: CA Press.

Standish Group. (2015). Chaos Report 2015. Retrieved March 10, 2019, from https://www.infoq.com/articles/standish-chaos-2015

Tripp, J. F., & Armstrong, D. J. (2014). Exploring the relationship between organizational adoption motives and the tailoring of agile methods. *Proceedings of the Annual Hawaii International Conference on System Sciences*, 4799–4806.

VersionOne. (2018). 12 th Annual State of Agile Development Survey. Retrieved May 24, 2019, from https://www.versionone.com/about/press-releases/12th-annual-state-of-agile-survey-open/

West, D. (2011). Water-Scrum-Fall Is The Reality Of Agile For Most Organizations Today. Retrieved May 24, 2019, from http://www.storycology.com/uploads/1/1/4/9/11495720/water-scrum-fall.pdf

Wiedemann, A., Wiesche, M., Gewald, H., & Krcmar, H. (2018). Integrating DevOps within IT Organizations – Key Pattern of a Case Study. In *Projektmanagement und Vorgehensmodelle* (pp. 157–166).

Womack, J. P., Jones, D. T., & Roos, D. (1990). *The Machine That Changed the World*. NY: Rawson Associates.