Adapting the (Big) Data Science Engineering Process to the Application of Test Driven Development

Daniel Staegemann[a], Matthias Volk[b] and Klaus Turowski
Magdeburg Research and Competence Cluster VLBA, Otto-von-Guericke University Magdeburg, Magdeburg, Germany

Keywords: Big Data, Data Science, Software Engineering, Big Data Engineering, Test Driven Development, TDD, Process, BDSEP.

Abstract: Knowledge, information, and modern technologies have become some of the most influential drivers of today's society, consequently leading to a high popularity of the concepts of big data (BD). However, their actual harnessing is a demanding task that is accompanied by many barriers and challenges. To facilitate the realization of the corresponding projects, the (big) data science engineering process (BDSEP) has been devised to support researchers and practitioners in the planning and implementation of data intensive projects by outlining the relevant steps. However, the BDSEP is only geared towards a test last development approach. With recent works suggesting the application of test driven development (TDD) in the big data domain, it appears reasonable to also provide a corresponding TDD focused equivalent to the BDSEP. Therefore, in the publication at hand, using the BDSEP as a foundation, the test driven big data science engineering process (TDBDSEP) is proposed, facilitating the application of TDD in the big data domain and further enriching the discourse on BD quality assurance.

1 INTRODUCTION

Knowledge, information, and modern technologies have become some of the most influential drivers of today's society (Levin and Mamlok 2021). Consequently, the concepts of big data (BD) and big data analytics (BDA) are extremely relevant and promising for many organizations across varying domains and sizes. The potential applications and desired benefits are manyfold (Poleto et al. 2017; van der Aalst and Damiani 2015). This includes, for instance, customer relation management, marketing, managerial decision support, improvements to maintenance and supply chain management, or the generation of ideas and insights for the exploitation of new markets and products. However, the actual harnessing is a demanding task that is accompanied by many barriers and challenges. The main factors influencing the obtained results are the quality of the used data, the competence and willingness of the responsible users, and the quality of the application's implementation (Janssen et al. 2017; Staegemann et al. 2019a). While all those aspects are highly important, the focus of the publication at hand is on the latter. Despite the popularity of BD, the corresponding quality assurance is not yet mature and new approaches, methods and tools are still being actively explored. One example of this is the adaptation of the test driven development (TDD) approach to the BD domain (Staegemann et al. 2020b). This promises to bring several benefits, such as an improvement to the developed systems' quality, a subsequent increase of trust by the users, and also more flexibility when it comes to the adaptation of the applications to new requirements and changes to the relevant environment. However, to our knowledge, there is no guideline on how to structure the corresponding activities for the test driven implementation of a BD project. Yet, in the form of the *(big) data science engineering process* (BDSEP), as proposed by Volk et al. (2020a), there is one for general BD endeavours. Therefore, it appears reasonable to adapt it to the application of TDD. For this reason, within this work, the following research question (RQ) shall be answered:

[a] https://orcid.org/0000-0001-9957-1003
[b] https://orcid.org/0000-0002-4835-919X

Staegemann, D., Volk, M. and Turowski, K.
Adapting the (Big) Data Science Engineering Process to the Application of Test Driven Development.
DOI: 10.5220/0011289200003280
In *Proceedings of the 19th International Conference on Smart Business Technologies (ICSBT 2022)*, pages 120-129
ISBN: 978-989-758-587-6; ISSN: 2184-772X

RQ: How can the (big) data science engineering process be adapted to the application of test driven development?
To answer the RQ, the publication at hand is structured as follows. After this introduction, the most relevant terms and concepts are outlined in the background section. Afterwards, the BDSEP is presented in a separate section to account for its significance in the course of this work. This is followed by the development of the adapted process that supports the application of TDD. Finally, in the concluding remarks, the proposed artifact is further discussed, the presented work is recapitulated, and avenues for future research are outlined.

2 BACKGROUND

To facilitate a common understanding of the relevant terms and concepts, those are in the following briefly outlined to establish a solid foundation for the remainder of the publication at hand.

2.1 Big Data

Despite big data being one of today's big trends (Ghasemaghaei and Calic 2020; Volk et al. 2020b), and consequently also intense scientific discourse (Staegemann et al. 2019b), there is still no universally used definition for the term itself. In fact, not even the origins of the term are completely clear (Diebold 2012).

However, the definition that is provided by the National Institute of Standards and Technology (NIST), is widely acknowledged, and therefore also relied upon for the publication at hand. It states that big data *"consists of extensive datasets primarily in the characteristics of volume, velocity, variety, and/or variability that require a scalable architecture for efficient storage, manipulation, and analysis"* (Chang and Grady 2019).

Here, volume indicates the amount of data, regarding the number and/or size of files, that have to be processed by the corresponding applications (Russom 2011). Velocity refers to two aspects, the speed with which the data are incoming and the timeliness that is expected for the application's results (Gandomi and Haider 2015). Variety addresses the data's heterogeneity, which is, inter alia, expressed through it being differently structured (structured, semi-structured, unstructured), the use of varying units of measurement and formats as well as different contexts it originates from (Gani et al. 2016). Finally, by variability it is expressed that the aforementioned characteristics, but also the questions that shall be answered through the use of BD, as well as the data's content can change over time (Katal et al. 2013; Staegemann et al. 2020a; Wu et al. 2014).

Besides those four characteristics, there are, however, further aspects that are relevant in the BD context. The quality of the used data is, for example, extremely important and has huge impact on the analysis results (Hazen et al. 2014). Moreover, besides the data, BDA combines organizational, human, and further technical aspects (Alharthi et al. 2017). The latter is emphasized through a plethora of available tools and techniques (Turck and Obayomi 2019), which renders it hard to make the right choice, when it comes to the technology selection (Volk et al. 2021). Finally, due to the potentially high impact of the BDA applications on the success of the applying organizations (Müller et al. 2018), and the resulting need for trust and appreciation by the responsible decision makers to assure correct use (Günther et al. 2017), comprehensive quality assurance is of utmost importance for the corresponding endeavors (Gao et al. 2016; Ji et al. 2020; Staegemann et al. 2021b).

2.2 Big Data Engineering

As a consequence of the aforementioned big data characteristics, the implementation of the corresponding systems significantly differs from conventional IT projects, since there needs to be a huge focus on the handling and interpretation of data. This often increases the development's complexity. The term "big data engineering" (BDE) describes the entirety of the activities that are associated with the creation of those BD systems (Volk et al. 2019). This field that is in the intersection of big data, data science, and systems engineering includes numerous tasks in several phases. In the beginning, there is the project planning with steps like the requirements engineering (Altarturi et al. 2017). This is followed by the actual design and implementation, including aspects like the technology selection (Lehmann et al. 2016). Finally, the solution's deployment ensues. Additionally, the aspect of quality assurance has to be considered.

To facilitate the BDE process and support practitioners as well as researchers in the realization of their BD endeavors, Volk et al. (2020a) have developed the (big) data science engineering process (BDSEP) that outlines the sequence of activities when creating such a BD application.

2.3 Test Driven Development

As shown by the literature, the application of TDD is a way of increasing a developed application's quality (Staegemann et al. 2021a). This is mainly based on two aspects. By the corresponding increase of the test coverage, the detection of errors is facilitated. Further, the design of the developed system is also influenced. The latter effect is caused by TDD heavily relying on the decomposition of the developed application into possibly small pieces. Due to the correspondingly decreased complexity, it is easier to avoid errors and, additionally, the maintainability is also increased (Crispin 2006; Shull et al. 2010).

While usually features are planned, implemented and then tested, this order is changed when applying TDD. After the first step, which now also puts emphasis on breaking down the envisioned functionality into small, capsulated parts (Fucci et al. 2017), the writing of the tests follows. To assure that they indeed test new aspects, they are subsequently run, with the expectation to fail, since the actual implementation has not yet happened (Beck 2015). Consequently, based on that premise, in case they pass, they have to be reworked. Once the tests are set up, the real implementation happens, enabling the new functionality. Here, aspects like the elegance of the code or the adherence to conventions can be ignored, as long as the tests pass (Crispin 2006). Only afterwards the codes overall quality is improved through refactoring (Beck 2015). This is supported by the previously written tests that help to detect if new errors were introduced during this procedure. As stated previously, this overall process with its focus on incremental changes and small tasks (Williams et al. 2003) not only impacts the test coverage and provides the developers with faster feedback, due to shorter test cycles (Janzen and Saiedian 2005), but also heavily influences the developed solution's design (Janzen and Saiedian 2008).

Usually, unit tests are the backbone of TDD. However, those are supposed to be complemented by other types of tests such as integration or system tests (Sangwan and Laplante 2006), with especially the former being seen as essential (Kum and Law 2006). Moreover, it is common to use continuous integration (CI) pipelines when applying TDD to enable test automation and, therefore, assure a high test frequency without the need for the developers to cumbersomely run the tests manually (Karlesky et al. 2007; Shahin et al. 2017). In doing so, once a change to the code is made, the existing tests are run by a CI server to check if any new errors have been introduced.

2.4 Microservices

The idea behind the microservice concept is to partition the developed application into multiple smaller services, which subsequently cooperate to solve the given task (Nadareishvili et al. 2016). Oftentimes, those services are constructed to provide a certain business functionality. This allows for a high degree of specialization in the implementation.

Each microservice runs in its own process. As a consequence of their independent nature, their implementation can also be heterogeneous (Freymann et al. 2020). Therefore, the responsible developers of each microservice can autonomously decide on the utilized technology stack and programming languages. To enable the communication among the services, only lightweight solutions are used. Due to their properties, microservices can be separately deployed and used. To automate the former, it is common to use continuous deployment tools and pipelines.

While, in software engineering, achieving a high degree of modularity is not only considered desirable, but also challenging (Faitelson et al. 2018), the use of microservices facilitates this task, since it is achieved by design. Moreover, when changes are implemented, it is often sufficient to only redeploy the respective microservice instead of the entire system. As a result, the effort for maintenance as well as for modifications is reduced. This, in turn, promotes an evolutionary design with frequent and controlled changes (Krylovskiy et al. 2015).

2.5 Test Driven Development in Big Data

Since BD applications are highly complex and also extremely quality sensitive, while TDD is capable of improving a developed application's quality, its application in the BD domain appears obvious. As the technical foundation for the concrete realisation, the use of microservices has been proposed (Staegemann et al. 2020b). This is based on the strong synergy that exists between the concept of microservices and the breaking down of the desired applications into possibly small parts as it is core of the TDD methodology (Shakir et al. 2021). By utilizing microservices, each business functionality can be designed as a separate service that can also be independently scaled to correspond to the arising workloads. This also allows to distribute the development across different teams that can act mostly independent of each other and are further free to use the technologies and tools of their choice

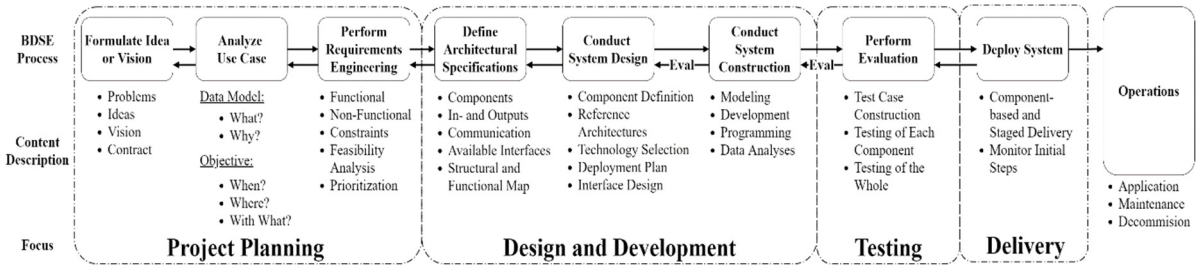

Figure 1: The (Big) Data Science Engineering Process (BDSEP) (Volk et al. 2020a).

instead of having to find an overarching consensus as it would be needed for a monolithic solution.

Since the created tests enable the developers to easily and immediately validate the functionality of any changes to the system, TDD also increases the flexibility of BD applications, since it is easier to implement changes to adapt to new needs and changes in the application environment. However, due to the inherent complexity, the application of TDD in the BD domain is a challenging task with the research on it being not yet very mature. To somewhat reduce the complexity and support researchers and practitioners in realizing their own endeavours, the use of a corresponding process model that helps to structure the necessary activities appears to be sensible.

3 THE (BIG) DATA SCIENCE ENGINEERING PROCESS (BDSEP)

To facilitate the introduction of BD applications and overcome the challenges of BDE, Volk et al. (2020a) have proposed the BDSEP. By combining knowledge and practices from information systems engineering as well as insights into data science processes, they crafted the BDSEP to support researchers and practitioners in the planning and implementation of data intensive projects by outlining the relevant steps, needed for the corresponding endeavours.

On a high level, the BDSEP comprises four main phases, namely *project planning*, *design and development*, *testing*, and *delivery*. While those as well as the steps described in the following, are generally performed in the given order, it is always possible to go back to previous activities if deemed necessary.

The first phase begins with the need to formulate a general idea or vision what shall be achieved by introducing a new system. This is followed by a more in-depth analysis of the concrete use case, including considerations regarding the necessary data and a clear definition of the objectives. Subsequently, the requirements engineering is performed, determining the functional and non-functional requirements as well as possible constraints and the respective priorities.

In the second phase, the architectural specifications are defined. This includes aspects such as the system's components with their in- and outputs, the intended communication, and the available interfaces. Then, the system design is conducted. The previously determined components are further specified, the most suitable technologies are chosen, and the deployment plan is crafted. For those tasks, the harnessing of reference architectures (Ataei and Litchfield 2020), best practices (Pääkkönen and Pakkala 2015), and decision support systems (Volk et al. 2019) is explicitly highlighted as advisable. Once the design is finished, the system's construction can take place. Apart from its development, the applications running on it are programmed and the necessary algorithms are developed or integrated.

The testing of the created solution constitutes the third phase of the process. Here, it is identified, what should be tested, the corresponding test cases are constructed, subsequently run and the results are evaluated. This applies to each component individually as well as to the system as a whole.

Once all the tests are passed, the delivery as the fourth phase succeeds. For this distribution of the solution to the target environment it is highlighted, that, due to its complexity, a staged process should be chosen (Chen et al. 2015; Mobus and Kalton 2015) to detect unforeseen issues. Therefore, this procedure should also be comprehensively monitored

Finally, those four main phases of the BDSEP are followed by the system's actual operation, including the necessary maintenance and at the end of its lifetime also its decommissioning. While it is not strictly a part of the engineering and is, therefore, also not seen as part of the main phases, it is evidently highly relevant with respect to the success of the developed application.

An overview of the process in its entirety is given in Figure 1, which is heavily based on the original depiction in (Volk et al. 2020a).

While the BDSEP in its current form fits to the needs of many BD endeavours, it is clearly geared towards a test last development (TLD) approach, where the testing only follows the implementation. For the application of TDD, there is, to our knowledge, currently no similar proposition. However, while there are significant differences between TLD and TDD, major parts of the BDSEP appear to be still applicable, which makes it reasonable to use it as a foundation for the development of this work's contribution, the test driven big data science engineering process (TDBDSEP).

4 ADAPTING THE BDSEP TO TDD (TDBDSEP)

To create the TDBDSEP, two pillars are built upon. Those are the BDSEP (Volk et al. 2020a), which is used as the foundation, as well as the concept and terminology for using TDD in the BD domain (Staegemann et al. 2020b). One important aspect of the latter is the consideration of different levels when regarding the developed solution. Besides the system level, there are the component level, the sub-component or microservice level, and the method level. The latter deals, according to its name, with the separate methods and functions, that are implemented in the course of the project, without considering how their role in the bigger picture. In the microservice level, the services in their entirety are regarded. The services, in turn, are the building blocks of components. Those are (virtual) units that are contentually connected due to their functionality. Examples for such components could be the import of data when it is realized by multiple services that are specialized to get data from one specific (type of) source or the utilized data's pre-processing, if it comprises various steps that are implemented as discrete microservices. However, there are no clear rules for the definition of the components. It depends on the respective developers and their evaluation of the developed system. Furthermore, a microservice can be part of multiple components, but always at least belongs to one and each component consists of one or many sub-components. Finally, on the system level, the developed solution is regarded as a whole, which could be seen as the equivalent of a monolithic implementation (Shakir et al. 2021).

To create a process that is geared towards the application of TDD, it is necessary to account for those levels, since having only one generic test activity as in the BDSEP is no longer sufficient.

However, the initial considerations regarding a BD project remain the same, independently of the decision if a TLD or a TDD approach is chosen, since the respective particularities only come into play once a rough concept for the desired product is devised.

Therefore, the first phase of the BDSEP, the *project planning*, can be carried over to the TDBDSEP without the need for modifications. This means, that, again, at first the rough idea or vision for the project is formulated, based on the perceived problem or need that caused its inception. This is followed by a more in-depth analysis of the use case. Here it is clarified, which objective should be fulfilled, and the corresponding specifics (e.g., time, location, or stakeholders) are discussed. Moreover, it is determined which data should be used for which purpose, where they come from, what their characteristics are, and which implications come from this (e.g., if orchestration or harmonization of different data sources is necessary). Afterwards, the requirements engineering is performed, comprising functional and non-functional ones, including the corresponding prioritization, but also aspects such as the incorporation of constraints and a feasibility analysis.

Following the project planning, an entirely new second phase is introduced, which deals with the *success definition*. For this purpose, the criteria to evaluate if the aspired goals of the implementation have been achieved are determined. This entails, for instance, which inputs should lead to which outputs, but also the general system behavior as well as any other aspects that are deemed relevant and can be evaluated. In the subsequent activity, the corresponding test cases for the system as a whole are constructed. Those might be automated tests, but also manually conducted ones. Since this activity is primarily geared towards the actual implementation in daily production and the intended users' perspective, relevant business stakeholders, such as managers, domain experts, and targeted decision makers should be heavily involved.

The third phase is heavily leaning on the second phase of the BDSEP, yet some adjustments come into play. Because the term component in the BDSEP has not exactly the same meaning as the term has in the context of the above introduced terminology, it is replaced with the word "element". Yet, the definition of the components is also newly introduced. Further, since one of the big advantages of microservice

architectures is the option to conduct the actual development in a distributed fashion, once the underlying architecture and design are known, design and development are detached from each other. For this reason, the *design* is a separate phase that contains two activities, namely the definition of architectural specifications and the system design. Those are mostly identical to the corresponding activities from the BDSEP. Yet, the preparation of the implementation plan is explicitly introduced because of the additional complexity due to the distributed nature. Further the technology selection no longer happens during the system design and is postponed instead, because this decision is up to the developers of the respective microservices. This way, following the idea behind the microservice concept, each team can make the most sensible choice with respect to the task, the members' skills, preferences, or other factors that are considered relevant. As during the project planning and success definition, it is again possible to go back to the prior activity if an issue or an oversight becomes apparent.

The TDBDSEP's fourth phase, *development and testing*, constitutes the biggest deviation from the foundational BDSEP. Even though it is somewhat the counterpart to the second aspect of its design and development phase as well as the testing phase, the TDD approach causes significant changes. Following its concept, the first task is to prepare the evaluation of the parts that shall be developed next. This is done in two activities, one on the component level and, thereafter, one for the microservices. Once those are set up, the actual implementation of the chosen service can take place. In contrast to the BDSEP, the technology selection only happens now, allowing for more autonomy in the construction process. Further, the service is created in a test driven fashion, which makes the unit testing of its internal functions a key aspect. Again, for all the described activities, it is possible to go back to the previous one if it is deemed sensible. After the construction is completed, the execution of the prepared tests ensues. This comprises three activities. In the first one, the tests for the microservice are run. If they don't pass, the process goes back to the construction activity. Otherwise, there are two options. Either there are still more services to be constructed in the component, then the corresponding tests for the next one are written and it is subsequently constructed, or this was the last service in the component, which leads to the next activity. There, the test cases that were created for the component level are run. If they fail, the next step would be to go back to the test creation for the microservice that is identified as responsible, since apparently some aspects have not been sufficiently reflected by the existing tests for it. In case of success,

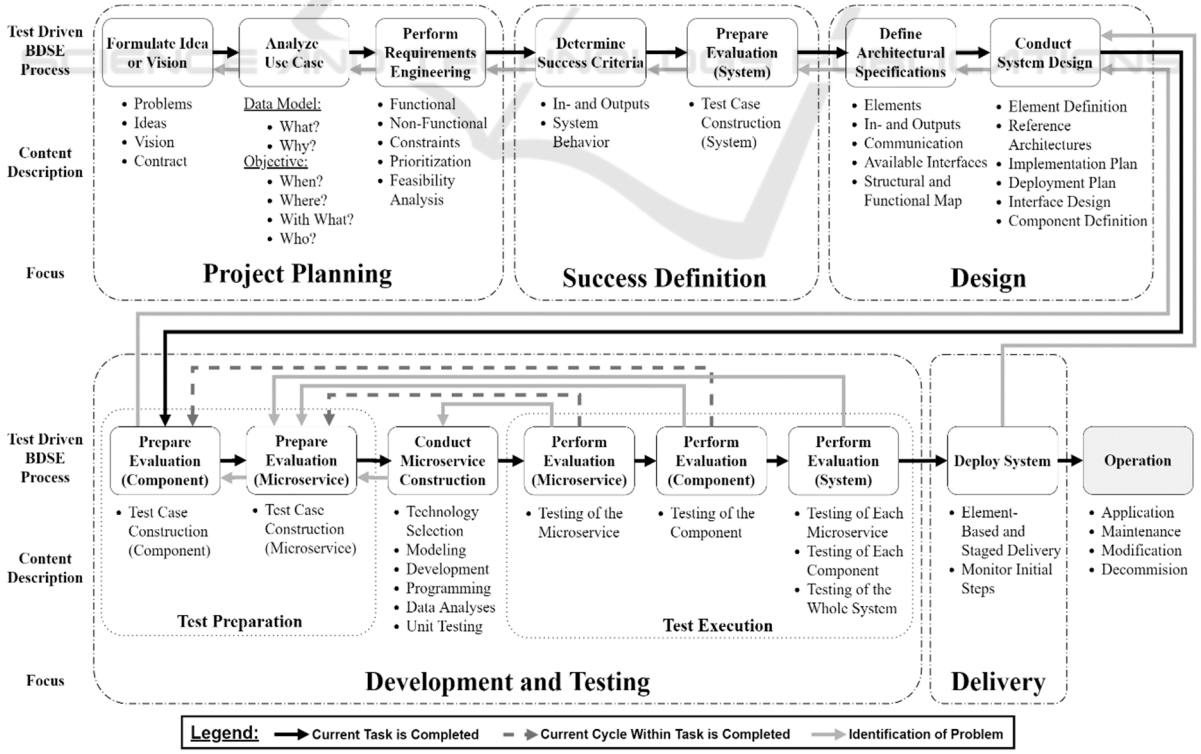

Figure 2: The Test Driven Big Data Science Engineering Process (TDBDSEP).

there are again two options. If there are more components that need to be implemented, the tests for the next one are written, which is followed by the subsequent steps. Should this have been the last missing piece for the system, the final evaluation can take place as the third activity of the test execution. There, the available tests for all the components and microservices are repeated. Further, also the tests that were created in the success definition phase are performed. Therefore, this activity gives the most comprehensive assessment of the developed system and covers all aspects that have been deemed relevant by the developers. If there are any issues occurring, the process is continued from the test creation for the service that is identified as the cause, following the same logic as in the previous step.

However, when the final testing procedure is successfully concluded, the *delivery* as the fifth phase can follow. Similar to the project planning, it can be carried over from the BDSEP as it is, since it is not majorly affected by the TDD approach. Therefore, it is, again, a closely monitored staged process (Chen et al. 2015; Mobus and Kalton 2015). In case of identified problems, the process should be traversed again from the system design activity, since errors during the implementation would have been likely identified through the created tests, which hints towards an issue with the design.

Finally, the five main phases of the TDBDSEP are followed by the system's actual *operation*. This includes, besides the productive utilization, again, the necessary maintenance as well as the decommissioning. However, this time, the former is facilitated by the strong modularization and the availability of comprehensive tests, which makes it easier to modify or replace elements without risking the introduction of new issues.

An illustration of the TDBDSEP to facilitate the comprehensibility of its structure and contents is depicted in Figure 2.

Even though the described process is rather comprehensive, some aspects have been simplified to increase clarity and readability. While it is generally possible for a microservice to be assigned to multiple components, as it was stated in the beginning of this section, the prior descriptions assume that each service is part of only one component. In situations where this is not the case, corresponding modifications to the process have to be factored in. The same applies to the fact that the process describes a setting in which the development is conducted in a linear fashion, whereas in reality, a parallelization during the development and testing phase is not only feasible, but possibly also advisable.

5 CONCLUDING REMARKS

With big data becoming more and more important regarding both, the prevalence of its application as well as the importance within the utilizing organizations, the related scientific discourse is very active. This applies, for instance, to the exploration of its practical use in different scenarios, organizational aspects, and questions regarding the technical realization. An important facet of the latter is the facilitation of the corresponding quality assurance, since the quality of the provided solutions is highly important when striving to maximize the benefits offered by the use of BD. One rather recent proposition in that regard is the application of TDD, based on microservices, in the BD domain. However, while there is guidance on the realization of BD projects through the BDSEP, it is not suited for TDD and, to our knowledge, there was also no other comparable process model that is. Yet, to reduce (similarly to the BDSEP) the complexity, and support researchers and practitioners in realizing their own test driven BD endeavours, the creation of a corresponding process model that helps to structure the necessary activities appears to be desirable. To bridge this gap, in the publication at hand, it was explored how the BDSEP can be adapted to the application of TDD. Thereby, the BDSEP was taken as a foundation that was then modified to reflect the specificities of the TDD approach, resulting in the TDBDSEP as this work's contribution.

While some aspects remained the same, compared to the BDSEP, the strong connection between the design and testing also led to major changes regarding the process' phases and activities. It now comprises five phases, namely *project planning*, *success definition*, *design*, *development and testing*, and *delivery*, which are followed by the actual *operation*. Even though the proposed process is generally comprehensive, for the sake of clarity, there had to be made some compromises that lead to certain limitations. Despite the possibility of a microservice belonging to several (virtual) components at once, this is not reflected in the description, to avoid complicating it for the reader and therefore hampering its application and dissemination. Yet, in situations where this option becomes relevant, it must be accounted for by the TDBDSEP's applicants. Further, while it is generally possible and oftentimes advisable to conduct the implementation of the separate microservices in a parallelized fashion through multiple teams, for the TDBDSEP, this is also simplified to a linear sequence of singular activities, making it easier for the reader to follow.

With respect to future research, there are two main avenues that should be pursued. The first one is to further explore and outline the details of the described phases and activities, providing prospective applicants with additional insights on how to shape their projects to obtain the best possible results. Moreover, the TDBDSEP should be evaluated in and possibly refined through the application in varying settings and domains, amending the theoretical considerations with ancillary inputs from practice.

REFERENCES

Alharthi, A., Krotov, V., and Bowman, M. (2017). "Addressing barriers to big data," *Business Horizons* (60:3), pp. 285-292 (doi: 10.1016/j.bushor.2017.01.002).

Altarturi, H. H., Ng, K.-Y., Ninggal, M. I. H., Nazri, A. S. A., and Ghani, A. A. A. (2017). "A requirement engineering model for big data software," in *Proceedings of the IEEE 2017 Conference on Big Data and Analytics (ICBDA),* Kuching, Malaysia. 16.11.2017 - 17.11.2017, pp. 111-117 (doi: 10.1109/ICBDAA.2017.8284116).

Ataei, P., and Litchfield, A. (2020). "Big Data Reference Architectures, a systematic literature review," in *Australasian Conference on Information Systems (ACIS) 2020,* Wellington, New Zealand, AIS.

Beck, K. (2015). *Test-Driven Development*: *By Example*, Boston: Addison-Wesley.

Chang, W. L., and Grady, N. (2019). "NIST Big Data Interoperability Framework: Volume 1, Definitions," *Special Publication (NIST SP),* Gaithersburg, MD: National Institute of Standards and Technology.

Chen, H.-M., Kazman, R., Haziyev, S., and Hrytsay, O. (2015). "Big Data System Development: An Embedded Case Study with a Global Outsourcing Firm," in *First International Workshop on Big Data Software Engineering - BIGDSE 2015*, IEEE, pp. 44-50 (doi: 10.1109/BIGDSE.2015.15).

Crispin, L. (2006). "Driving Software Quality: How Test-Driven Development Impacts Software Quality," *IEEE Software* (23:6), pp. 70-71 (doi: 10.1109/MS.2006.157).

Diebold, F. X. (2012). "On the Origin(s) and Development of the Term 'Big Data'," *SSRN Electronic Journal* (doi: 10.2139/ssrn.2152421).

Faitelson, D., Heinrich, R., and Tyszberowicz, S. (2018). "Functional Decomposition for Software Architecture Evolution," in *Model-Driven Engineering and Software Development*, L. F. Pires, S. Hammoudi and B. Selic (eds.), Cham: Springer International Publishing, pp. 377-400 (doi: 10.1007/978-3-319-94764-8_16).

Freymann, A., Maier, F., Schaefer, K., and Böhnel, T. (2020). "Tackling the Six Fundamental Challenges of Big Data in Research Projects by Utilizing a Scalable and Modular Architecture," in *Proceedings of the 5th International Conference on Internet of Things, Big Data and Security,* Prague, Czech Republic. 07.05.2020 - 09.05.2020, SCITEPRESS - Science and Technology Publications, pp. 249-256 (doi: 10.5220/0009388602490256).

Fucci, D., Erdogmus, H., Turhan, B., Oivo, M., and Juristo, N. (2017). "A Dissection of the Test-Driven Development Process: Does It Really Matter to Test-First or to Test-Last?" *IEEE Transactions on Software Engineering* (43:7), pp. 597-614 (doi: 10.1109/tse.2016.2616877).

Gandomi, A., and Haider, M. (2015). "Beyond the hype: Big data concepts, methods, and analytics," *International Journal of Information Management* (35:2), pp. 137-144 (doi: 10.1016/j.ijinfomgt.2014.10.007).

Gani, A., Siddiqa, A., Shamshirband, S., and Hanum, F. (2016). "A survey on indexing techniques for big data: taxonomy and performance evaluation," *Knowledge and Information Systems* (46:2), pp. 241-284 (doi: 10.1007/s10115-015-0830-y).

Gao, J., Xie, C., and Tao, C. (2016). "Big Data Validation and Quality Assurance -- Issuses, Challenges, and Needs," in *Proceedings of the 2016 IEEE Symposium on Service-Oriented System Engineering (SOSE),* Oxford, United Kingdom. 29.03.2016 - 02.04.2016, IEEE, pp. 433-441 (doi: 10.1109/SOSE.2016.63).

Ghasemaghaei, M., and Calic, G. (2020). "Assessing the impact of big data on firm innovation performance: Big data is not always better data," *Journal of Business Research* (108:2), pp. 147-162 (doi: 10.1016/j.jbusres.2019.09.062).

Günther, W. A., Rezazade Mehrizi, M. H., Huysman, M., and Feldberg, F. (2017). "Debating big data: A literature review on realizing value from big data," *The Journal of Strategic Information Systems* (26:3), pp. 191-209 (doi: 10.1016/j.jsis.2017.07.003).

Hazen, B. T., Boone, C. A., Ezell, J. D., and Jones-Farmer, L. A. (2014). "Data quality for data science, predictive analytics, and big data in supply chain management: An introduction to the problem and suggestions for research and applications," *International Journal of Production Economics* (154), pp. 72-80 (doi: 10.1016/j.ijpe.2014.04.018).

Janssen, M., van der Voort, H., and Wahyudi, A. (2017). "Factors influencing big data decision-making quality," *Journal of Business Research* (70:3), pp. 338-345 (doi: 10.1016/j.jbusres.2016.08.007).

Janzen, D., and Saiedian, H. (2005). "Test-driven development concepts, taxonomy, and future direction," *Computer* (38:9), pp. 43-50 (doi: 10.1109/MC.2005.314).

Janzen, D., and Saiedian, H. (2008). "Does Test-Driven Development Really Improve Software Design Quality?" *IEEE Software* (25:2), pp. 77-84 (doi: 10.1109/MS.2008.34).

Ji, S., Li, Q., Cao, W., Zhang, P., and Muccini, H. (2020). "Quality Assurance Technologies of Big Data Applications: A Systematic Literature Review," *Applied Sciences* (10:22), p. 8052 (doi: 10.3390/app10228052).

Karlesky, M., Williams, G., Bereza, W., and Fletcher, M. (2007). "Mocking the Embedded World: Test-Driven Development, Continuous Integration, and Design Patterns," in *Embedded Systems Conference,* San Jose, California, USA. 01.04.2007 - 05.04.2007, UBM Electronics.

Katal, A., Wazid, M., and Goudar, R. H. (2013). "Big data: Issues, challenges, tools and Good practices," in *Sixth International Conference on Contemporary Computing*, Parashar (ed.), Noida, India. 08.08.2013 - 10.08.2013, IEEE, pp. 404-409 (doi: 10.1109/IC3.2013.6612229).

Krylovskiy, A., Jahn, M., and Patti, E. (2015). "Designing a Smart City Internet of Things Platform with Microservice Architecture," in *2015 3rd International Conference on Future Internet of Things and Cloud (FiCloud 2015)*, I. Awan (ed.), Rome, Italy. 24.08.2015 - 26.08.2015, Piscataway, NJ: IEEE, pp. 25-30 (doi: 10.1109/FiCloud.2015.55).

Kum, W., and Law, A. (2006). "Learning Effective Test Driven Development - Software Development Projects in an Energy Company," in *Proceedings of the First International Conference on Software and Data Technologies,* Setúbal, Portugal. 11.09.2006 - 14.09.2006, SciTePress - Science and and Technology Publications, pp. 159-164 (doi: 10.5220/0001316101590164).

Lehmann, D., Fekete, D., and Vossen, G. (2016). "Technology selection for big data and analytical applications," Working Papers, ERCIS - European Research Center for Information Systems 27, Münster.

Levin, I., and Mamlok, D. (2021). "Culture and Society in the Digital Age," *Information* (12:2), p. 68 (doi: 10.3390/info12020068).

Mobus, G. E., and Kalton, M. C. (2015). *Principles of Systems Science*, New York, NY: Springer.

Müller, O., Fay, M., and Vom Brocke, J. (2018). "The Effect of Big Data and Analytics on Firm Performance: An Econometric Analysis Considering Industry Characteristics," *Journal of management information systems* (35:2), pp. 488-509 (doi: 10.1080/07421222.2018.1451955).

Nadareishvili, I., Mitra, R., McLarty, M., and Amundsen, M. (2016). *Microservice architecture*: *Aligning principles, practices, and culture*, Beijing, Boston, Farnham, Sebastopol, Tokyo: O´Reilly.

Pääkkönen, P., and Pakkala, D. (2015). "Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems," *Big Data Research* (2:4), pp. 166-186 (doi: 10.1016/j.bdr.2015.01.001).

Poleto, T., Heuer de Carvalho, V. D., and Costa, A. P. C. S. (2017). "The Full Knowledge of Big Data in the Integration of Inter-Organizational Information," *International Journal of Decision Support System Technology* (9:1), pp. 16-31 (doi: 10.4018/IJDSST.2017010102).

Russom, P. (2011). "Big Data Analytics: TDWI Best Practices Report Fourth Quarter 2011,"

Sangwan, R. S., and Laplante, P. A. (2006). "Test-Driven Development in Large Projects," *IT Professional* (8:5), pp. 25-29 (doi: 10.1109/MITP.2006.122).

Shahin, M., Ali Babar, M., and Zhu, L. (2017). "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices," *IEEE Access* (5), pp. 3909-3943 (doi: 10.1109/ACCESS.2017.2685629).

Shakir, A., Staegemann, D., Volk, M., Jamous, N., and Turowski, K. (2021). "Towards a Concept for Building a Big Data Architecture with Microservices," in *Proceedings of the 24th International Conference on Business Information Systems,* Hannover, Germany/virtual. 14.06.2021 - 17.06.2021, pp. 83-94 (doi: 10.52825/bis.v1i.67).

Shull, F., Melnik, G., Turhan, B., Layman, L., Diep, M., and Erdogmus, H. (2010). "What Do We Know about Test-Driven Development?" *IEEE Software* (27:6), pp. 16-19 (doi: 10.1109/MS.2010.152).

Staegemann, D., Volk, M., Daase, C., and Turowski, K. (2020a). "Discussing Relations Between Dynamic Business Environments and Big Data Analytics," *Complex Systems Informatics and Modeling Quarterly* (23), pp. 58-82 (doi: 10.7250/csimq.2020-23.05).

Staegemann, D., Volk, M., Jamous, N., and Turowski, K. (2019a). "Understanding Issues in Big Data Applications - A Multidimensional Endeavor," in *Proceedings of the Twenty-fifth Americas Conference on Information Systems,* Cancun, Mexico. 15.08.2019 - 17.08.2019.

Staegemann, D., Volk, M., Jamous, N., and Turowski, K. (2020b). "Exploring the Applicability of Test Driven Development in the Big Data Domain," in *Proceedings of the ACIS 2020,* Wellington, New Zealand. 01.12.2020 - 04.12.2020.

Staegemann, D., Volk, M., Lautenschlager, E., Pohl, M., Abdallah, M., and Turowski, K. (2021a). "Applying Test Driven Development in the Big Data Domain – Lessons From the Literature," in *2021 International Conference on Information Technology (ICIT),* Amman, Jordan. 14.07.2021 - 15.07.2021, IEEE, pp. 511-516 (doi: 10.1109/ICIT52682.2021.9491728).

Staegemann, D., Volk, M., Nahhas, A., Abdallah, M., and Turowski, K. (2019b). "Exploring the Specificities and Challenges of Testing Big Data Systems," in *Proceedings of the 15th International Conference on Signal Image Technology & Internet based Systems,* Sorrento.

Staegemann, D., Volk, M., and Turowski, K. (2021b). "Quality Assurance in Big Data Engineering - A Metareview," *Complex Systems Informatics and Modeling Quarterly* (28), pp. 1-14 (doi: 10.7250/csimq.2021-28.01).

Turck, M., and Obayomi, D. (2019). "The Big Data Landscape," available at http://dfkoz.com/big-data-landscape/, accessed on Jan 13 2020.

van der Aalst, W., and Damiani, E. (2015). "Processes Meet Big Data: Connecting Data Science with Process Science," *IEEE Transactions on Services Computing* (8:6), pp. 810-819 (doi: 10.1109/TSC.2015.2493732).

Volk, M., Staegemann, D., Bischoff, D., and Turowski, K. (2021). “Applying Multi-Criteria Decision-Making for the Selection of Big Data Technologies,” in *Proceedings of the Twenty-seventh Americas Conference on Information Systems,* Montreal, Canada/Virtual. 09.08.2021 - 13.08.2021.

Volk, M., Staegemann, D., Bosse, S., Häusler, R., and Turowski, K. (2020a). “Approaching the (Big) Data Science Engineering Process,” in *Proceedings of the 5th International Conference on Internet of Things, Big Data and Security,* Prague, Czech Republic. 07.05.2020 - 09.05.2020, SCITEPRESS - Science and Technology Publications, pp. 428-435 (doi: 10.5220/0009569804280435).

Volk, M., Staegemann, D., Pohl, M., and Turowski, K. (2019). “Challenging Big Data Engineering: Positioning of Current and Future Development,” in *Proceedings of the IoTBDS 2019*, SCITEPRESS - Science and Technology Publications, pp. 351-358 (doi: 10.5220/0007748803510358).

Volk, M., Staegemann, D., and Turowski, K. (2020b). “Big Data,” in *Handbuch Digitale Wirtschaft*, T. Kollmann (ed.), Wiesbaden: Springer Fachmedien Wiesbaden, pp. 1-18 (doi: 10.1007/978-3-658-17345-6_71-1).

Williams, L., Maximilien, E. M., and Vouk, M. (2003). “Test-driven development as a defect-reduction practice,” in *Proceedings of the 14th ISSRE,* Denver, Colorado, USA. 17.11.2003 - 20.11.2003, IEEE, pp. 34-45 (doi: 10.1109/ISSRE.2003.1251029).

Wu, X., Zhu, X., Wu, G.-Q., and Ding, W. (2014). “Data mining with big data,” *IEEE Transactions on Knowledge and Data Engineering* (26:1), pp. 97-107 (doi: 10.1109/TKDE.2013.109).