2020

Exploring the Applicability of Test Driven Development in the Big Data Domain

Daniel Staegemann
Otto-von-Guericke University Magdeburg, daniel.staegemann@ovgu.de

Matthias Volk
Otto von Guericke Universitat Magdeburg, matthias.volk@ovgu.de

Naoum Jamous
Otto von Guericke Universitat, Magdeburg, naoum.jamous@ovgu.de

Klaus Turowski
Otto von Guericke Universitat, Magdeburg, klaus.turowski@ovgu.de

Exploring the Applicability of Test Driven Development in the Big Data Domain

Completed research paper

Daniel Staegemann
Magdeburg Research and Competence Cluster Very Large Business Applications
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
Email: daniel.staegemann@ovgu.de

Matthias Volk
Magdeburg Research and Competence Cluster Very Large Business Applications
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
Email: matthias.volk@ovgu.de

Naoum Jamous
Magdeburg Research and Competence Cluster Very Large Business Applications
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
Email: naoum.jamous@ovgu.de

Klaus Turowski
Magdeburg Research and Competence Cluster Very Large Business Applications
Otto-von-Guericke University Magdeburg
Magdeburg, Germany
Email: klaus.turowski@ovgu.de

Abstract

Big data analytics and the according applications have gained huge importance in daily life. This results on the one hand from their versatility and on the other hand from their capability to greatly improve an organization's performance when utilized appropriately. However, despite their prevalence and the corresponding attention through practitioners as well as the scientific world, the actual implementation still remains a challenging task. Therefore, without the adequate testing, the reliability of the systems and thus the obtained outputs is uncertain. This might reduce their utilization, or even worse, lead to a diminished decision-making quality. The publication at hand explores the adoption of test driven development as a potential approach for addressing this issue. Subsequently, using the design science research methodology, a microservice-based test driven development concept for big data (MBTDD-BD) is proposed. In the end, possible avenues for future research endeavours are indicated.

Keywords Big data, test driven development, big data engineering, microservices, design science research.

1 Introduction

Big data is a topic that has found its way into many areas of life. No matter if it is agriculture (Bronson and Knezevic 2016), tourism (Alaei et al. 2019), governance (Al-Sai and Abualigah 2017), education (Häusler et al. 2020), healthcare (Safa et al. 2019), manufacturing (Nagorny et al. 2017) or transportation (Fiore et al. 2019), practically everywhere, organizations are trying to harness the power of data, while scientists are making efforts to expand the corresponding knowledge. This also reflects in the annual spending (Lee 2017; Raguseo 2018) and the growing number of scientific contributions dealing with the topic that are publicized each year (Parlina et al. 2020). Besides the versatility of its application, big data also prospers due to the ever-increasing amount of data produced by today's society (Yin and Kaynak 2015). Even though, there is no universally used definition, the one provided by the national institute of standards and technology (NIST) is widely accepted. 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" (NIST 2019). While those characteristics lead to challenges that exceed the ones arising in traditional software engineering and the socio-technical nature of the corresponding systems can cause additional issues (Bonesso et al. 2020; Günther et al. 2017; Lunde et al. 2019; Staegemann et al. 2019b) it was shown, that the implementation of big data also promises noticeable advantages (Bughin 2016; Fay and Kazantsev 2018; Fosso Wamba et al. 2015; Müller et al. 2018; Popovič et al. 2018), which are especially valuable when facing high competition. However, since big data applications are highly complex, the impacts of even comparatively small malfunctions or rounding errors can build up to have a huge impact on the final result (Yang et al. 2018). This in turn reduces the confidence of decision makers and also the quality of the decisions made (Hazen et al. 2014; Redman 2004; Staegemann et al. 2019b). Yet, in contrast to traditional software engineering, where it has a long history (Lewis et al. 2009, p. 4), the testing of those systems is still often somewhat neglected (Staegemann et al. 2019b). One of the approaches applied in software engineering is the concept of test driven development (TDD) that arose from the agile domain and describes rather a philosophy instead of a mere testing technique (Munir et al. 2014). Since big data applications often require a certain degree of flexibility and also adaptability, while necessitating extensive testing, on the first glance, the adoption of TDD appears to be a promising solution for meeting those needs. Therefore, the publication at hand is committed to the following research question:

RQ: Is the test driven development approach suitable for the big data domain and which implications would be tied to its implementation?

In the pursuit of providing an answer to the RQ, at first, the applied research methodology is outlined. Afterward, the topics of big data and TDD are presented, constituting the background of the publication at hand. Subsequently, the applicability of TDD in big data engineering is examined and finally, a conclusion is drawn, highlighting possible future research avenues.

2 Methodology

In order to find a suitable answer on the aforementioned RQ the design science research (DSR) according to Hevner et al. (2004) is followed. This constructive scientific methodology seeks to develop and evaluate IT artefacts in context of information systems research. Generally, those artefacts shall be used to solve organizational problems and can be "*constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices), and instantiations (implemented and prototype systems*)" (Hevner et al. 2004, p. 77). To increase not only the general comprehensibility of the made steps but also the structure of this work, additionally, the workflow provided by the design science research methodology (DSRM) by Peffers et al. (2007) is used. The procedure, which is depicted in Figure 1, decomposes the DSR in six consecutive steps.

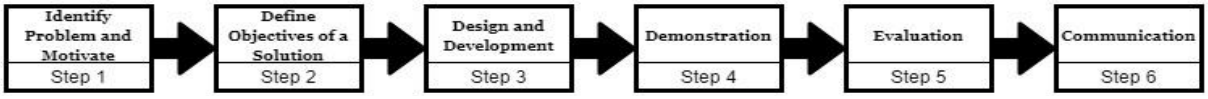

Figure 1: Nominal Process Sequence of the DSRM According to (Peffers et al. 2007)

The procedure starts with the initial identification and motivation of the problem. Afterward, in step two, the objectives of the solution are formulated. Both of them were already conducted within the first section, in the contribution at hand. For the third step, the design and development of the solution, grounded theory as well as an overview about the research background is needed. As a transition to the main part, this is described within the adhering section. Apart from the sole description of the domains big data and test driven development, furthermore, their intersection is presented in conjunction with

the current state of the art. Under consideration of this theory, in the following, the construction of the main artefact takes place, with the underlying explanations as an implicit, preliminary evaluation. However, while the communication is realized through the publication at hand, due to the nature of the artefact as a conceptual model, the demonstration will be postponed to subsequent research steps.

3 Research Background

To assess the viability of TDD in the domain of big data, it is at first necessary to establish an understanding of those terms and the underlying concepts. Therefore, in the following, both topics will be regarded separately, before the prospects of their coupling are explored in the subsequent section.

3.1 Big Data

While a few years ago the volume of data was considered as the most significant data characteristic of *big data*, today the term covers a broad range of characteristics in combination with complex, highly scalable systems that are used for the efficient storage, processing and analysis of this data (NIST 2019). These characteristics and the necessity to harness techniques and technologies, capable to overcome those, origins from a contribution provided by Doug Laney, a former business analyst of Gartner (Laney 2001). In here, the first time the characteristics volume, velocity and variety were introduced in context of the data management. Until today, many practitioners as well as researchers attempted to provide novel perspectives on those and newly identified characteristics, such as the veracity, variability or even valence of the data, as the *connectedness* of the data (Saggi and Jain 2018).

Volume is generally known as the amount of data that have to be processed. Due to the lack of a clear definition of the *amount* itself, authors referred this characteristics either to the number of records that have to be managed or the sheer size of the data (Russom 2011). Independent from its description, in recent years, a tremendous increase is noticeable. Eventually the "*time and expense required to process massive datasets was one of the original drivers for distributed processing*" (NIST 2019). A similar lack of a clear definition exists regarding the velocity. It either describes the speed of the incoming data that has to be handled by the system or the speed to fulfil a processing request (Gandomi and Haider 2015; Saggi and Jain 2018). Variety describes the heterogeneity of the handled data. This includes the general structure, the origin of the used data as well as the units (Gani et al. 2016). Complementary to the characteristics mentioned above, the variability refers to the possible variance of the dataset and, thus, changes of the volume, variety and velocity (NIST 2019). The same dependency applies on other characteristics, such as the veracity that targets the reliability and trustworthiness of the data (Pappas et al. 2019; Schroeck et al. 2012).

In the context of big data, the diversity of these data characteristics has led to the fact that established technologies can no longer solely be used. Instead, approaches that are capable to overcome those challenges, such as non-relational storage mechanisms (NoSQL) or distributed computation (MapReduce) have become the de facto standard in this domain. At the same time, the complexity of the development and testing of modern systems increased (Staegemann et al. 2019c). For now, a profound understanding of individual technologies, their interaction and the corresponding testing is required when it comes to the construction of the big data systems. This comprises the phases of project planning, design and development, testing and delivery, which are usually also conducted in the given order. Together, those activities constitute the discipline of big data engineering (Volk et al. 2020), starting with an identified necessity for a big data application and resulting in an operational solution, as depicted in Figure 2. Along with these complex tasks, significant assistance is facilitated by numerous guidelines, preliminary approaches that support a technology decision, as well as reference architectures (Volk et al. 2019). The latter can be observed as a kind of best practices in the area of big data, for the conceptual design of systems in which a certain combination of data characteristics has to be mastered. They combine the "general architecture knowledge and general experience with specific requirements for a coherent architectural solution for a specific problem domain. They document the structures of the system, the main system building blocks, their responsibilities, and their interactions" (Vogel et al. 2011, p. 232). Similar to the individual technologies themselves, a multitude of possible approaches exists that ranges from very general to domain specific deployments.

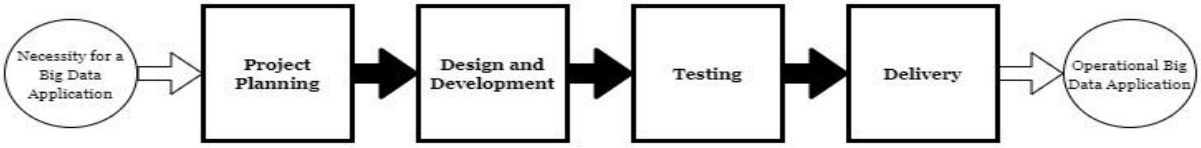

Figure 2: The Conventional Big Data Engineering Process With its Input and Output

3.2 Test Driven Development

As already mentioned earlier, TDD is not just a testing technique, but rather a philosophy that affects the whole process of creating software. The general idea is to swap the often prevalent order of writing code and afterward assuring its functionality. Instead, the test is written first, defining the desired behaviour of the software and only then the function itself is realized. Furthermore, while in the traditional approach some parts of the program might remain untested due to oversight, time constraints, self-deception, laziness or various other reasons, in TDD the existence of the corresponding test is a prerequisite for the implementation of every function. Therefore, the whole development is also carried out in small, incremental steps, each somewhat expanding the functionality and thus leading to an organic growth of the program (Williams et al. 2003). With this aspiration in mind, TDD comprises three major phases that are repeatedly traversed (Beck 2015, 9f.). As a result of this pattern, which is depicted in Figure 3, the creation of the test cases also falls into the responsibility of each developer himself instead of external specialists, since otherwise the idle time between the iterations would be too high.

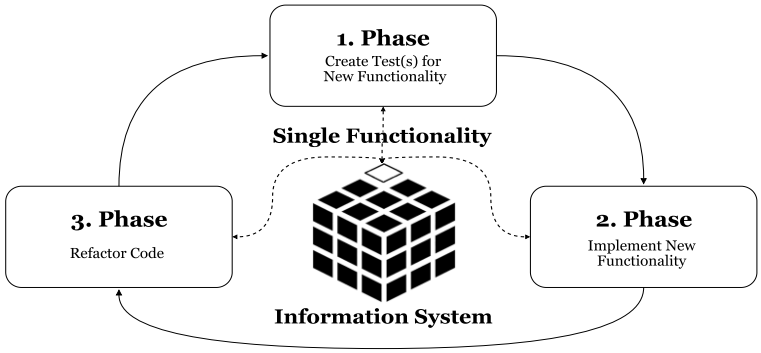

Figure 3: The Pattern of Test Driven Development

At the start of each cycle, the developer decides, which new functionality he wants to implement next. Here, the addition should be planned as small as possible. Once the task is defined, he implements tests for the envisioned behavior. During this task, he benefits from the small scope that has to be covered, as well as his insights into the purpose of the change that external testers possibly would not possess (Astels 2003, p. 27). This might for example prove to be helpful in defining edge-cases or critical inputs. Subsequently, the tests are run and since the functional code is not written yet, they are supposed to fail. Now, the actual implementation takes place, aiming for a preferably fast solution, since the task is only to provide the new functionality, not to create elegant code. Afterward, the tests (existing and new ones) are run. If all tests pass, this indicates that the new functionality is integrated and also that no previously working code was negatively affected. Otherwise, the implementation has to be adjusted, since it either did not deliver as expected or broke previously functioning parts. Once all tests are passed, the code can be refactored. This means to improve on the solution, clean up the code, eliminate duplicates and therefore improve its conciseness. While doing so, repeatedly running the tests ensures that everything is still working. When it comes to the interaction with other classes and objects that are not yet implemented, mock objects are used as a placeholder, simulating their expected behavior and therefore facilitating the regarded codes functionality (Kim et al. 2006). Since the additions are supposed to be small, the creation of the corresponding test cases will be usually relatively fast with the same applying to the initial implementation. However, the refactoring might take a considerable amount of the totally spent time, especially when there is already a lot of existing code.

As a result of this procedure, the developer is forced to put more thought into the decomposition of the big system into smaller components, leading to a better and less monolithic structure. Furthermore, due to the structure and the refactoring, the comprehensibility and maintainability of the code increases. The arguably biggest advantage, however, is the oftentimes more comprehensive test coverage together with the ability to repeatedly run those tests (in a usually automated fashion). This allows to assure the quality of the finished product or parts of it, as well as to quickly check the actual state during the development (Janzen and Saiedian 2005), helping to avoid the need to rewrite large portions of the code due to undetected but far-reaching errors. In context of principles, such as continuous integration, this circumstance facilitates a rapid decrease of the time spent for the actual deployment as well as the maintenance (Shahin et al. 2017). However, in turn for those benefits, there are also some drawbacks. For once, it means a severe change in the style of work for many developers. Especially when starting with the use of TDD, it might feel unfamiliar and cumbersome to work in small steps and think about

tests first, instead of focusing the actual task (Beck 2015, p. 10). Furthermore, much of the benefit comes from the ability to run the tests often. When there are frequent changes regarding the structure and used procedures, which results in the deletion and substitution of parts of the code, also the work for the creation of the corresponding tests is lost. Despite those challenges and a lack of definite outcomes, several studies indicate potential positive effects of the application of TDD on the quality of produced code respectively its testing (Bissi et al. 2016; George and Williams 2004; Marchenko et al. 2009; Maximilien and Williams 2003; Tosun et al. 2017; Tosun et al. 2018).

4 Applying Test Driven Development to Big Data Engineering

While TDD is mostly adopted in agile software development, it is not necessarily limited to that application domain, as the examples of test driven ontology development (Davies et al. 2019; Keet and Ławrynowicz 2016) or process modeling (Slaats et al. 2018) illustrate. Thus, the consideration of TDD as a possible approach in big data engineering seems not too far-fetched. Especially in application scenarios, where the occurrence of changes with regard to the asked questions or the used sources is frequent, and therefore, a once validated system cannot just be used unmodified for a long period of time (Staegemann et al. 2020), the confidence in its validity can decrease. Since, a high quality of the system as well as the used data is a prerequisite for the reliable acquisition of valuable insights, a lack of confidence in either one of them can prevent decision makers from incorporating the obtained insights in their considerations (Staegemann et al. 2019b). The ability to repeatedly and automatically run the prepared tests, whenever changes have occurred, can therefore constitute a valuable asset, not only potentially increasing the quality of the big data analysis, but possibly also its usage. Furthermore, given the outlined characteristics and some of the frequent use cases, not only the mere correctness of the processing is important, but also its speed and the ability to handle huge volumes of data as well as shifting compositions of the same (NIST 2019). Due to the fact that there is no point in a complete self-development, which ignores previous insights and accomplishments and a multitude of big data related tools (Grover and Kar 2017; Turck and Obayomi 2019) and proven reference architectures (Kreps 2014; Martínez-Prieto et al. 2015; Pääkkönen and Pakkala 2015) already exists, a certain structure is usually given from the beginning. This configuration however, comprising multiple different components, implicitly also specifies part of the structure that is used as a foundation for the TDD. In general, four kinds of units, that have to be tested, can be identified. Those represent varied scales and subsequently, they also have differing considerations regarding the testing. There is, for one thing, the *system* as a whole, as the biggest unit. It includes everything from start to finish and is the relevant object, with everything else just being auxiliary. On the opposing side of the scale, the *method* is the smallest entity, determining a minimal portion of the behaviour. Despite the choice of the term, which is borrowed from the software development domain, it does not necessarily have to be an actual method. While this will often be the case, configurations of third-party tools, like for example the specification of rows or columns and their content in a database could also fall under this category. The third group is on the level of tools and applications and is therefore situated between the methods and the system. However, since the combination of those parts can sometimes result in logical entities, it appears in some use cases, with the testing being one of those, to be reasonable to differentiate between *components*, which are said entities and *sub-components*, being the distinct applications (Mobus and Kalton 2015, pp. 108-110). Yet, since they are virtual constructs, derived from the present sub-components, the definition of components is rather flexible, meaning that for different contexts and purposes, different sub-components could be defined. A generalized example of such a structure is depicted in Figure 4.

When intending to implement a completely new big data application, or to make major architectural changes, in a first preparatory step, the general compatibility of the intended components should be assured by using mocks and trying to ensure an uninterrupted communication from start to end. While doing so, metrics like the communications speed or possible limitations regarding the maximum data volume should be explored (Chen et al. 2016). That way, unsuitable concepts can be identified at an early stage. Alternatively, if corresponding experiences are available or appropriate means, like for example a corresponding decision support system (Volk et al. 2019) are used, this step can be foregone. Once the feasibility of the intended architecture is proven, the actual development can start. Based on their structural differences, the aforementioned four categories naturally also have different requirements regarding their examination. On the method-level, analogously to the common software development, unit-tests (Runeson 2006) can be utilized to assert the correctness of the created algorithms. If it comes to the structure or operations of databases, this could be accomplished by issuing commands, like write and read, and comparing the outcomes with the expected behavior. The same applies to other external applications. Once all methods of a sub-component are step-by-step realized and therefore also successfully tested, its functionality is inherently too. However, to assure its compliance to the requirements with regard to processing speed or capacity, it has to be benchmarked and since later on

the used hardware is likely to be heterogeneous (Staegemann et al. 2019b), several repetitions might be required to avoid unrepresentative outliers.

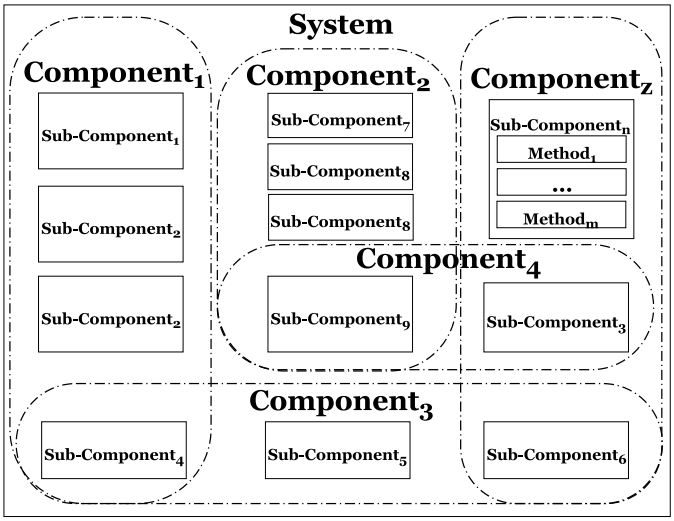

Figure 4: Exemplary Structure of a Big Data Architecture

For the benchmarking, either existing data or a generator for the creation of new ones, as exemplarily shown in (Alexandrov et al. 2012; Faltín et al. 2017; Ming et al. 2014), could be used (Han et al. 2014). While the tasks up until now are well-understood, the bigger challenges start on the component-level. Here, it is necessary to define, which functionality should be provided by each component and which requirements regarding the performance apply. Furthermore, tests for the assurance of those goals have to be created. Subsequently, it is determined, which sub-components form a component and, depending on the use case, sometimes also how many of each are needed to satisfy the demands. There, especially the transmission, respectively integration of data between two sub-components, has to be scrutinized, since those might for example use different formats, languages or datatypes, making according transformations mandatory. By sensibly defining those components, a certain degree of modularity is facilitated, which allows the addition of new sub-components or the modification of existing ones, without the necessity to re-test the whole system. Instead, only the parts that are directly affected by the changes have to be checked again. If, during the creation, some of the needed sub-components are not yet implemented, they can be temporarily replaced by the corresponding mock. For the system as a whole, the functional correctness can be assumed, when all the necessary sub-components, as well as their connections, factored in through the components, are working as intended. However, even though for the testing of the methods and sub-components, there are established approaches that cover most tasks, except the verification of configurations, the same does not apply to the component- and system-level. While previous research was often either very high level or the described solution was confronted with issues like the oracle problem (Gudipati et al. 2013; Staegemann et al. 2019a; Tao and Gao 2016), the approach proposed in the publication at hand promises a continuous testing on all levels. It reaches from the method to the system and at the same time, offers a great deal of flexibility. To put this concept into practice, the utilization of microservices, which are small, independent (in terms of deployment and scaling) and highly specialized applications (Thones 2015), seems to be a promising solution, especially since microservices have already succesfully been utilized in the big data domain (Freymann et al. 2020). An overview and association of the four testing units as well as the proposed testing approaches is given in Table 1.

Unit of Scale	Testing Facilitated Through
Method	Unit-Tests; Microservices
Sub-Component	Inherently through the verification of all the comprised methods; Benchmarking
Component	Microservices; Benchmarking
System	Inherently through the verification of all the comprised components

Table 1. Overview of the Proposed Testing Approaches

With test routines being implemented as microservices that can be flexibly combined when required, a multitude of different test configurations can be composed. Therefore, it is possible to either run every single test, covering every aspect of the system, to just examine a single (sub-) component, or everything in between. This in turn avoids a waste of time and resources compared to constant full-scale tests. Furthermore, the whole arrangement could be deployed on demand, allowing to rent external capacities when needed for larger tests instead of having to build and hold ready own hardware for those probably rare situations. Despite being specifically mentioned in Table 1, the benchmarking could also be realized via microservices. A schematic depiction of the flexibility provided by the proposed concept is shown in Figure 5.

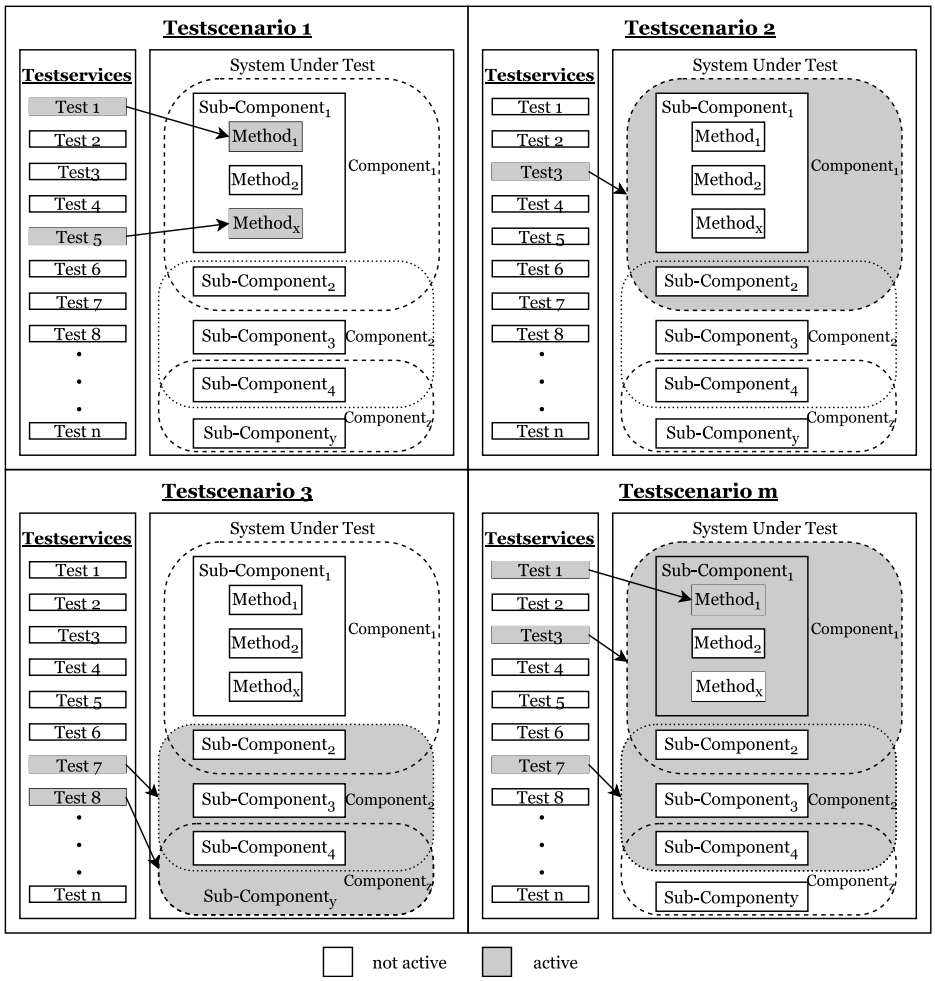

Figure 5: An Exemplary Depiction of the MBTDD-BD's Provided Flexibility

Though, for the sake of clarity, unit-tests that are incorporated into the method themselves are not depicted, with the same applying for a separate mentioning of intended benchmarks. Although the general idea of a distributed big data testing approach is not novel (Staegemann et al. 2019a) and some of those even take the need for frequently repeating existing tests into consideration (Sudsee and Kaewkasi 2019), to our knowledge, this is the first concept for the application of microservice-based TDD in the big data domain (MBTDD-BD). Consequently, this is also the DSR artefact, an answer to the RQ, as well as the main contribution of the publication at hand. Due to the independent nature of the microservices, which allows a discretionary composition, the previously highlighted modularity is achieved. This also enables the parallel implementation of different parts of the system by multiple teams, since each one of them contributes its own necessary tests without requiring extensive amounts of coordination. At the same time, the severe focusing of the specialized tests in conjunction with the developer's deep insights in the particularities of the task he is working on, promises reliable results. By splitting the sometimes extremely complex big data architectures in numerous small parts and assuring their quality individually, as well as in conjunction, the oracle problem might not be solved in its entirety, but the extend of the individual tasks gets reduced and becomes therefore potentially more manageable.

5 Conclusion

For almost ten years, the topic of big data with its technologies, application areas, promises and risks has attracted numerous researchers and practitioners, leading to a variety of corresponding contributions. Yet, despite great efforts, the implementation of those systems, the big data engineering, still remains a highly sophisticated task that entails many challenges. Since the testing is an important part of this process, but there is still a gap regarding appropriate measures and approaches, the publication at hand makes two contributions. At first, according to the RQ, the general applicability off test driven development in a big data context is demonstrated and building on this, the design science methodology is used to propose a novel microservice-based test driven development concept for big data (MBTDD-BD), auguring a high test coverage, conjoined with the possibility of a mostly autonomous cooperation of distributed teams and a maximum of flexibility. Since a lackluster quality assurance of big data applications leads to a lack of trust in the obtained results, this can reduce their usage. Even worse, wrong decisions based on their flawed outputs, can have detrimental impact. Therefore, through the avoidance of those pitfalls, the application of the MBTDD-BD can lead to the realization of competitive advantages by increasing the decision-making quality of an organization. However, even though the test coverage as a common weak point is eliminated, as in every testing approach, the quality of the obtained results highly depends on the correctness and meaningfulness of the implemented test cases. For the future, it is intended to enhance the evaluation by implementing prototypes to show the feasibility of the proposed approach. This also allows to identify best practices and potential challenges. Additionally, once a certain degree of maturity in the process of creating prototypes is reached, it is intended to interview industry experts. That way, on the one hand, their technical expertise can be incorporated and on the other hand, they represent potential real world users who can give insights regarding especially promising application areas and the corresponding business perspective. An overview of the application of the DSR methodology, including steps that are intended for the future, is depicted in Table 2. Besides those core activities, the investigation of ways to automatically calculate the optimal test configuration when changes to a previously verified big data system occur, as well as the exploration of solutions for the convenient deployment and steering of test configurations constitute ancillary research avenues.

Research Step	Status	Description
Identify Problem and Motivate	Finished	The importance of big data for today's society but also its susceptibility to flaws and their impact are discussed. Furthermore, the necessity for comprehensive quality assurance techniques is highlighted.
Define Objectives of a Solution	Finished	The solution shall increase the flexibility of big data applications while not compromising on their quality.
Design and Development	Finished	To increase modularity and allow for the repeated and automated testing when modifications occur, a microservice-based test driven development concept is proposed.
Demonstration	In Progress	The feasibility of the proposed approach will be demonstrated through prototypical implementations that act as a proof of concept and also allow to gain additional insights.
Evaluation	In Progress	Initial evaluative considerations have already been conducted during the previous research steps. For the future, it is planned to analyze objective metrics, such as test coverage in exemplary implementations, but also to interview industry experts to gauge the technical feasibility as well as possible real-world application areas and the corresponding potentials and expectations.
Communication	In Progress	The communication of the research is realized through publications and presentations in scientific conferences and journals.

Table 2. The Application of the DSR Methodology (Peffers et al. 2007)

6 References

Alaei, A. R., Becken, S., and Stantic, B. 2019. "Sentiment Analysis in Tourism: Capitalizing on Big Data," *Journal of Travel Research* (58:2), pp. 175-191 (doi: 10.1177/0047287517747753).

Alexandrov, A., Tzoumas, K., and Markl, V. 2012. "Myriad," *Proceedings of the VLDB Endowment* (5:12), pp. 1890-1893 (doi: 10.14778/2367502.2367530).

Al-Sai, Z. A., and Abualigah, L. M. 2017. "Big data and E-government: A review," in *Proceedings of the 8th International Conference on Information Technology,* Amman, Jordan. 17.05.2017 - 18.05.2017, IEEE, pp. 580-587.

Astels, D. 2003. *Test-driven development*: *A practical guide*, Prentice Hall PTR.

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

Bissi, W., Serra Seca Neto, A. G., and Emer, M. C. F. P. 2016. "The effects of test driven development on internal quality, external quality and productivity: A systematic review," *Information and Software Technology* (74), pp. 45-54 (doi: 10.1016/j.infsof.2016.02.004).

Bonesso, S., Bruni, E., and Gerli, F. 2020. "The Organizational Challenges of Big Data," in *Behavioral Competencies of Digital Professionals*, S. Bonesso, E. Bruni and F. Gerli (eds.), Cham: Springer International Publishing, pp. 1-19.

Bronson, K., and Knezevic, I. 2016. "Big Data in food and agriculture," *Big Data & Society* (3:1) (doi: 10.1177/2053951716648174).

Bughin, J. 2016. "Big data, Big bang?" *Journal of Big Data* (3:1), pp. 1-14 (doi: 10.1186/s40537-015-0014-3).

Chen, H.-M., Kazman, R., and Haziyev, S. 2016. "Strategic Prototyping for Developing Big Data Systems," *IEEE Software*, pp. 1-9 (doi: 10.1109/MS.2016.65).

Davies, K., Keet, C. M., and Lawrynowicz, A. 2019. "More Effective Ontology Authoring with Test-Driven Development and the TDDonto2 Tool," *International Journal on Artificial Intelligence Tools* (28:7) (doi: 10.1142/S0218213019500234).

Faltín, T., Hanzeli, M., Šípek, V., Škvařil, J., Variš, D., and Mlýnková, I. H. 2017. "BDgen," in *Proceedings of the 21st International Database Engineering & Applications Symposium*, B. C. Desai, J. Hong and R. McClatchey (eds.), Bristol, United Kingdom. 12.07.2017-14.07.2017, New York, New York, USA: ACM Press, pp. 200-208.

Fay, M., and Kazantsev, N. 2018. "When Smart Gets Smarter: How Big Data Analytics Creates Business Value in Smart Manufacturing," in *Proceedings of the International Conference on Information Systems*, J. Pries-Heje, S. Ram and M. Rosemann (eds.), San Francisco, Association for Information Systems.

Fiore, S., Elia, D., Pires, C. E., Mestre, D. G., Cappiello, C., Vitali, M., Andrade, N., Braz, T., Lezzi, D., Moraes, R., Basso, T., Kozievitch, N. P., Fonseca, K. V. O., Antunes, N., Vieira, M., Palazzo, C., Blanquer, I., Meira, W., and Aloisio, G. 2019. "An Integrated Big and Fast Data Analytics Platform for Smart Urban Transportation Management," *IEEE Access* (7), pp. 117652-117677 (doi: 10.1109/ACCESS.2019.2936941).

Fosso Wamba, S., Akter, S., Edwards, A., Chopin, G., and Gnanzou, D. 2015. "How 'big data' can make big impact: Findings from a systematic review and a longitudinal case study," *International Journal of Production Economics* (165), pp. 234-246 (doi: 10.1016/j.ijpe.2014.12.031).

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.

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).

George, B., and Williams, L. 2004. "A structured experiment of test-driven development," *Information and Software Technology* (46:5), pp. 337-342 (doi: 10.1016/j.infsof.2003.09.011).

Grover, P., and Kar, A. K. 2017. "Big Data Analytics: A Review on Theoretical Contributions and Tools Used in Literature," *Global Journal of Flexible Systems Management* (18:3), pp. 203-229 (doi: 10.1007/s40171-017-0159-3).

Gudipati, M., Rao, S., Mohan, N. D., and Gajja, N. K. 2013. "Big Data : Testing Approach to Overcome Quality Challenges," *Infosys Labs Briefings* (11:1), pp. 65-73.

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).

Han, R., Lu, X., and Xu, J. 2014. "On Big Data Benchmarking," in *Big Data Benchmarks, Performance Optimization, and Emerging Hardware*, J. Zhan, R. Han and C. Weng (eds.), Springer, pp. 3-18.

Häusler, R., Staegemann, D., Volk, M., Bosse, S., Bekel, C., and Turowski, K. 2020. "Generating Content-Compliant Training Data in Big Data Education," in *Proceedings of the 12th International Conference on Computer Supported Education,* Prague, Czech Republic. 02.05.2020 - 04.05.2020, SCITEPRESS - Science and Technology Publications, pp. 104-110.

Hazen, B., Boone, C., Ezell, J., 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).

Hevner, A. R., March, S. T., Park, J., and Ram, S. 2004. "Design science in information systems research," *MIS quarterly*, pp. 75-105.

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).

Keet, C. M., and Ławrynowicz, A. 2016. "Test-Driven Development of Ontologies," in *The Semantic Web. Latest Advances and New Domains*, H. Sack, E. Blomqvist, M. d'Aquin, C. Ghidini, S. P. Ponzetto and C. Lange (eds.), Cham: Springer International Publishing, pp. 642-657.

Kim, T., Park, C., and Wu, C. 2006. "Mock Object Models for Test Driven Development," in *Proceedings of the SERA 2006,* Seattle, WA, USA. 09.08.2006-11.08.2006, IEEE, pp. 221-228.

Kreps, J. 2014. *Questioning the Lambda Architecture*: *The Lambda Architecture has its merits, but alternatives are worth exploring*. https://www.oreilly.com/ideas/questioning-the-lambda-architecture. Accessed 6 November 2020.

Laney, D. 2001. "3D Data Management: Controlling Data Volume, Velocity, and Variety,"

Lee, I. 2017. "Big data: Dimensions, evolution, impacts, and challenges," *Business Horizons* (60:3), pp. 293-303 (doi: 10.1016/j.bushor.2017.01.004).

Lewis, W. E., Dobbs, D. D., and Veerapillai, G. 2009. *Software testing and continuous quality improvement*, Boca Raton: CRC Press.

Lunde, T. Å., Sjusdal, A. P., and Pappas, I. O. 2019. "Organizational Culture Challenges of Adopting Big Data: A Systematic Literature Review," in *Digital Transformation for a Sustainable Society in the 21st Century*, I. O. Pappas, P. Mikalef, Y. K. Dwivedi, L. Jaccheri, J. Krogstie and M. Mäntymäki (eds.), Springer International Publishing, pp. 164-176.

Marchenko, A., Abrahamsson, P., and Ihme, T. 2009. "Long-Term Effects of Test-Driven Development A Case Study," in *Agile Processes in Software Engineering and Extreme Programming*, P. Abrahamsson, M. Marchesi and F. Maurer (eds.), Berlin, Heidelberg: Springer, pp. 13-22.

Martínez-Prieto, M. A., Cuesta, C. E., Arias, M., and Fernández, J. D. 2015. "The Solid architecture for real-time management of big semantic data," *Future Generation Computer Systems* (47), pp. 62-79 (doi: 10.1016/j.future.2014.10.016).

Maximilien, E. M., and Williams, L. 2003. "Assessing test-driven development at IBM," in *Proceedings of the 25th International Conference on Software Engineering,* Portland, OR, USA. 10.05.2003 - 10.05.2003, IEEE, pp. 564-569.

Ming, Z., Luo, C., Gao, W., Han, R., Yang, Q., Wang, L., and Zhan, J. 2014. "BDGS: A Scalable Big Data Generator Suite inBig Data Benchmarking,"

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

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).

Munir, H., Wnuk, K., Petersen, K., and Moayyed, M. 2014. "An experimental evaluation of test driven development vs. test-last development with industry professionals," in *Proceedings of the 18th EASE*, M. Shepperd, T. Hall and I. Myrtveit (eds.), London, England. 13.05.2014 - 14.05.2014, New York, USA: ACM Press, pp. 1-10.

Nagorny, K., Lima-Monteiro, P., Barata, J., and Colombo, A. W. 2017. "Big Data Analysis in Smart Manufacturing: A Review," *International Journal of Communications, Network and System Sciences* (10:03), pp. 31-58 (doi: 10.4236/ijcns.2017.103003).

NIST 2019. "NIST Big Data Interoperability Framework: Volume 1, Definitions, Version 3," Gaithersburg, MD: National Institute of Standards and Technology.

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).

Pappas, I. O., Mikalef, P., Dwivedi, Y. K., Jaccheri, L., Krogstie, J., and Mäntymäki, M. (eds.) 2019. *Digital Transformation for a Sustainable Society in the 21st Century*, Springer International Publishing.

Parlina, A., Ramli, K., and Murfi, H. 2020. "Theme Mapping and Bibliometrics Analysis of One Decade of Big Data Research in the Scopus Database," *Information* (11:2), p. 69 (doi: 10.3390/info11020069).

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. 2007. "A Design Science Research Methodology for Information Systems Research," *Journal of Management Information Systems* (24:3), pp. 45-77 (doi: 10.2753/MIS0742-1222240302).

Popovič, A., Hackney, R., Tassabehji, R., and Castelli, M. 2018. "The impact of big data analytics on firms' high value business performance," *Information Systems Frontiers* (20:2), pp. 209-222 (doi: 10.1007/s10796-016-9720-4).

Raguseo, E. 2018. "Big data technologies: An empirical investigation on their adoption, benefits and risks for companies," *International Journal of Information Management* (38:1), pp. 187-195 (doi: 10.1016/j.ijinfomgt.2017.07.008).

Redman, T. C. 2004. "Data: An Unfolding Quality Disaster," *DM Review*.

Runeson, P. 2006. "A survey of unit testing practices," *IEEE Software* (23:4), pp. 22-29 (doi: 10.1109/MS.2006.91).

Russom, P. 2011. *Big Data Analytics*: *TDWI Best Practices Report Fourth Quarter 2011*. https://vivomente.com/wp-content/uploads/2016/04/big-data-analytics-white-paper.pdf. Accessed 6 November 2020.

Safa, B., Zoghlami, N., Abed, M., and Tavares, J. M. R. S. 2019. "BIG DATA for Healthcare: A Survey," *IEEE Access* (7), pp. 7397-7408 (doi: 10.1109/ACCESS.2018.2889180).

Saggi, M. K., and Jain, S. 2018. "A survey towards an integration of big data analytics to big insights for value-creation," *Information Processing & Management* (54:5), pp. 758-790 (doi: 10.1016/j.ipm.2018.01.010).

Schroeck, M., Shockley, R., Smart, J., Romero-Morales, D., and Tufano, P. 2012. "Analytics: The real-world use of big data: How innovative enterprises extract value from uncertain data,"

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).

Slaats, T., Debois, S., and Hildebrandt, T. 2018. "Open to Change: A Theory for Iterative Test-Driven Modelling," in *Business Process Management*, M. Weske, M. Montali, I. Weber and J. Vom Brocke (eds.), Cham: Springer International Publishing, pp. 31-47.

Staegemann, D., Hintsch, J., and Turowski, K. 2019a. "Testing in Big Data: An Architecture Pattern for a Development Environment for Innovative, Integrated and Robust Applications," in *Proceedings*

of the WI2019, T. Ludwig and V. Pipek (eds.), Siegen, Germany. 23.02.2019-27.02.2019, pp. 279-284.

Staegemann, D., Volk, M., Daase, C., and Turowski, K. 2020. "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. 2019b. "Understanding Issues in Big Data Applications - A Multidimensional Endeavor," in *Proceedings of the Twenty-fifth AMCIS,* Cancun.

Staegemann, D., Volk, M., Nahhas, A., Abdallah, M., and Turowski, K. 2019c. "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, Italy.

Sudsee, B., and Kaewkasi, C. 2019. "An Improvement of a Checkpoint-based Distributed Testing Technique on a Big Data Environment," in *Proceedings of the 21st International Conference on Advanced Communication Technology (ICACT),* PyeongChang Kwangwoon_Do, Korea (South). 17.02.2019 - 20.02.2019, IEEE, pp. 1081-1090.

Tao, C., and Gao, J. 2016. "Quality Assurance for Big Data Applications– Issues, Challenges, and Needs," in *Proceedings of the Twenty-Eighth International Conference on Software Engineering and Knowledge Engineering,* California, Pittsburgh: KSI Research Inc. and Knowledge Systems Institute Graduate School, pp. 375-381.

Thones, J. 2015. "Microservices," *IEEE Software* (32:1), p. 116 (doi: 10.1109/MS.2015.11).

Tosun, A., Ahmed, M., Turhan, B., and Juristo, N. 2018. "On the effectiveness of unit tests in test-driven development," in *Proceedings of the 2018 International Conference on Software and System Process*, M. Kuhrmann, R. V. O'Connor and D. Houston (eds.), Gothenburg, Sweden. 26.05.2018-27.05.2018, New York, USA: ACM Press, pp. 113-122.

Tosun, A., Dieste, O., Fucci, D., Vegas, S., Turhan, B., Erdogmus, H., Santos, A., Oivo, M., Toro, K., Jarvinen, J., and Juristo, N. 2017. "An industry experiment on the effects of test-driven development on external quality and productivity," *Empirical Software Engineering* (22:6), pp. 2763-2805 (doi: 10.1007/s10664-016-9490-0).

Turck, M., and Obayomi, D. 2019. *The Big Data Landscape*. http://dfkoz.com/big-data-landscape/. Accessed 6 November 2020.

Vogel, O., Arnold, I., Chughtai, A., and Kehrer, T. 2011. *Software Architecture*, Berlin, Heidelberg: Springer Berlin Heidelberg.

Volk, M., Staegemann, D., Bosse, S., Häusler, R., and Turowski, K. 2020. "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.

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*, pp. 351-358.

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.

Yang, M., Adomavicius, G., Burtch, G., and Ren, Y. 2018. "Mind the Gap: Accounting for Measurement Error and Misclassification in Variables Generated via Data Mining," *Information Systems Research* (29:1), pp. 4-24 (doi: 10.1287/isre.2017.0727).

Yin, S., and Kaynak, O. 2015. "Big Data for Modern Industry: Challenges and Trends [Point of View]," *Proceedings of the IEEE* (103:2), pp. 143-146 (doi: 10.1109/JPROC.2015.2388958).