Tackling the Six Fundamental Challenges of Big Data in Research Projects by Utilizing a Scalable and Modular Architecture

Andreas Freymann[a], Florian Maier[b], Kristian Schaefer[c] and Tom Böhnel[d]
Anwendungszentrum KEIM, Fraunhofer Institute for Industrial Engineering IAO, Esslingen am Neckar, Germany

Keywords: Big Data Fundamentals, Scalability, Modular Architecture, Research Projects, Data Lake, Real Time, Open Source, Docker Swarm, Micro Services.

Abstract: Over the last decades the necessity for processing and storing huge amounts of data has increased enormously, especially in the fundamental research area. Beside the management of large volumes of data, research projects are facing additional fundamental challenges in terms of data velocity, data variety and data veracity to create meaningful data value. In order to cope with these challenges solutions exist. However, they often show shortcomings in adaptability, usability or have high licence fees. Thus, this paper proposes a scalable and modular architecture based on open source technologies using micro-services which are deployed using Docker. The proposed architecture has been adopted, deployed and tested within a current research project. In addition, the deployment and handling is compared with another technology. The results show an overcoming of the fundamental challenges of processing huge amounts of data and the handling of Big Data in research projects.

1 INTRODUCTION

Processing and storing of today's increasing amount of Big Data has become an important key factor in all areas of life such as research, industry, public or social networks (Y. Demchenko et al., 2013). One responsible factor is that data comes from everywhere and from everybody (S. Kaisler et al., 2013). It originates for example from an enormous amount of dynamic sensors and devices around the world creating massive amounts of data (M. Kiran et al., 2015), (L. Sun et al., 2017). Within companies, Big Data also plays a crucial role such as for decision-making (Stucke and Grunes, 2016). Thus, new technologies and architectures are necessary to deal with Big Data to reach valuable results (Katal et al., 2013), (Volk et al., 2019). However, bringing Big Data together with research projects which investigate new technologies and approaches causes additional challenges which need to be handled.

The general handling of Big Data requires to consider certain characteristics such as data *volume* or data *velocity* (Katal et al., 2013). However, Big Data faces challenges as well (S. Kaisler et al., 2013), (Katal et al., 2013), (Volk et al., 2019). They can be derived from the Big Data characteristics (Ahmed Oussous et al., 2018) which we identify as fundamental challenges *(FCs) of Big Data* at the same time. They make processing and storing of data more difficult. Also just the processing of data or the variety of nature might cause difficulties (Volk et al., 2019). In addition, these FCs of Big Data get intensified in conjunction with research projects as they represent additional challenges due to their settings such as for instance financial limitations. We call them *FCs of research projects*.

There already are solutions in practice and literature which try to handle FCs of Big Data (S. Kaisler et al., 2013), (M. Kiran et al., 2015). However, several challenges still persist: Firstly, traditional solutions for Big Data often show shortcomings in efficiency, scalability, flexibility and performance (Ahmed Oussous et al., 2018). Secondly, such solutions do not consider the additional challenges that come along with the FCs of research projects. Thirdly, many solutions have cost models instead of having an open-source character (M. Kiran et al., 2015).

This paper provides an architecture which has been adopted and developed further to overcome the difficulties of handling FCs of Big Data in conjunction with FCs of research projects by showing a suc-

[a] https://orcid.org/0000-0002-3735-4545
[b] https://orcid.org/0000-0002-5695-6509
[c] https://orcid.org/0000-0002-7855-6741
[d] https://orcid.org/0000-0001-6426-2606

Freymann, A., Maier, F., Schaefer, K. and Böhnel, T.
Tackling the Six Fundamental Challenges of Big Data in Research Projects by Utilizing a Scalable and Modular Architecture.
DOI: 10.5220/0009388602490256
In *Proceedings of the 5th International Conference on Internet of Things, Big Data and Security (IoTBDS 2020)*, pages 249-256
ISBN: 978-989-758-426-8

cessful deployment in a current project (i-rEzEPT[1]).

The content of this work is based on a previous publication which presents a flexible architecture for smart cities by taking up several architectural design patterns (K. Lehmann and A. Freymann, 2018). Our work provides a special architectural design featuring e.g., a scalable and modular design based on open source technologies or a distributed server cluster. For the evaluation of our suggested architecture, it is deployed, used and tested within the aforementioned research project. In addition, we compare the deployment of the architecture with two different technologies: Docker swarm and Kuberentes (Kubernetes Authors, 2020). The results show that the proposed architecture overcomes the FCs of Big Data and FCs of research projects.

The paper is structured as follows: After an insight into background information in Section 2, Section 3 describes the FCs of Big Data and FCs of research projects and how they influence one another. Section 4 presents our architecture derived from seven identified requirements. The architecture is then evaluated in Section 5. Finally, after the related work in Section 6, the last Section 7 discusses the conclusion of our work and gives a future outline.

2 BACKGROUND INFORMATION

Dealing with Big Data requires a well-defined architecture and technologies to be able to process the huge amounts of data (Katal et al., 2013). Strong basic features of those architectures usually are flexibility and scalability to cope with changes such as changing requirements or new data sources (K. Lehmann and A. Freymann, 2018). Different design patterns for architectures are state-of-the-art which have been used over the last years such as the lambda architecture (a), micro-services (b) and distributed systems (c).

A *Lambda Architecture* offers a solution for an efficient processing of large amounts of data (a). It enables simultaneously real-time analysis and more complex, accurate analysis using batch methods. The architecture consists of three layers: *speed layer*, *batch layer* and *serving layer*. The *speed layer* processes an incoming data stream in real-time. The *batch layer* executes heavy computations in a lower frequency. The output of speed and batch layer can be joined before presentation. The *serving layer* stores results of computations, handles queries and provides the interface for the user. (M. Kiran et al., 2015)

A *Micro-service* architecture divides a complex system into many small applications, called microservices (b). They offer an interesting contribution to the architecture, as they only processes small independent units and therefore provide a lot of flexibility (Peinl et al., 2016), (L. Sun et al., 2017). In comparison, the traditional monolithic approach unifies a software solution in a single unified application. Microservices benefit of being highly horizontally scalable, flexible and easy to maintain. (L. Sun et al., 2017)

'A *Distributed System* is a collection of independent computers that appears to its users as a single coherent system' (Tanenbaum and van Steen, 2007, p. 2) (c). They provide high scalability as computers respectively servers can be added, changed or removed. The challenge of distributed systems is to manage and allocate tasks (e.g. micro-services) between the available computation resources (Verma et al., 2015).

In addition, to manage micro-services in distributed systems is a significant factor for a well functioning operation of an entire system. For the orchestration of micro-services, they get packed into containers. Those containers enable faster booting of the services and easy deployment (H. Li et al., 2019). This additionally simplifies the service orchestration as a whole such as by using automation functions.

3 FCs IN DETAIL

3.1 FCs of Big Data

Many definitions mention five basic characteristics which are related to Big Data (Y. Demchenko et al., 2013), (Katal et al., 2013), (S. Kaisler et al., 2013). These are often described as the "5 Vs" of Big Data: *volume, velocity, variety, veracity* and *value* (cf. Table 1). This work also considers the specific attribute *complexity* mentioned in (S. Kaisler et al., 2013), as research data often has a complex structure which makes this characteristic especially important. Table 1 represents the Big Data characteristics in detail.

Data Volume. It deals with the huge amount of data which need to be handled (Volk et al., 2019). At the same time, processing the volume is a challenge that Big Data has to face due to the fact that new data is continuously generated everywhere (S. Kaisler et al., 2013). Especially, smartphones or RFID devices produce a massive amount of data around the world (M. Kiran et al., 2015).

Data Velocity. It describes the frequency of incoming data from different sources (Katal et al.,

[1]The project i-rEzEPT is promoted by the German Federal Ministry of Transport and Digital Infrastructure. It investigates the participation of battery electric vehicles in the primary reserve market (Funding code:03EMF0103B).

Table 1: Characteristics of Big Data.

Data ...	Short description
Volume	Available amount of data existing within a certain context (S. Kaisler et al., 2013), (Volk et al., 2019).
Velocity	Speed or frequency at which data originates from a certain data source (Y. Demchenko et al., 2013).
Variety	Diversity the data can be represented by, e.g. images, text or videos. This also addresses the data streaming and data aggregation (Katal et al., 2013).
Complexity	Interconnectedness and interdependence of data content (S. Kaisler et al., 2013).
Veracity	Plausibility and correctness of data (Y. Demchenko et al., 2013).
Value	Creation of valuable information which can be further used such as for decision-making (Y. Demchenko et al., 2013).

2013). A high velocity requires transmitting and processing data quickly (Ahmed Oussous et al., 2018).
Data Variety. Data variety measures the diversity (Volk et al., 2019). It comprises, e.g., possible data formats such as documents, time series or videos being processed. The related challenge is that data is often incompatible, non-structured and inconsistent (S. Kaisler et al., 2013). This is also based on the large amount of different IoT devices which produce different data formats (L. Sun et al., 2017).
Data Complexity. Relationships and interconnections between data from various sources represent the data complexity (S. Kaisler et al., 2013). This means that data content depends on other data content. Challenges are linking and changing interconnected data across a large Big Data system (Katal et al., 2013).
Data Veracity. It is mentioned by (Y. Demchenko et al., 2013) and comprises consistency and trustworthiness of data. Ensuring a non-manipulation of data is important during data processing, beginning from trusted sources to a secure storage. Implausible data needs to be detected while it is being processed. Otherwise, data that has no trustworthiness or consistency might have negative impacts, e.g., interpretations.
Data Value. The data value is the reason why all Big Data efforts are made. It is created through four processing steps: *collection*, *cleaning*, *aggregation* and *presentation*. The data value focuses on the usefulness of data which means to create valuable information and knowledge which can be further used such as for decision-making (Y. Demchenko et al., 2013). This characteristic depends on a good consideration of all other Big Data characteristics.

3.2 FCs of Research Projects

Research projects have special settings, which differ from non-research projects without a research context which lead to different methods and architectures (Y. Demchenko et al., 2013). In our practice, we identified several interdependent characteristics (described in Table 2) which make a research project unique.

Table 2: Characteristics of research projects.

Character.	Short description
Large amounts of data	Research projects create complex and large amounts of data. (Y. Demchenko et al., 2013)
Volatile requirements	Quickly changing requirements.
Developing prototypes	Focus on research results, less concern for marketable products.
Available budgets	A given scope which limits financial options.
Innovative character	Trying new concepts and technologies.
Research community	Have an open character to share research results (Y. Demchenko et al., 2013).

Large Data Amount. A typical property of research projects is a large amount of data which needs to be processed and stored as hypotheses and research goals are pursued (Y. Demchenko et al., 2013). In conjunction with a Big Data context, the derived challenge from research projects is the confrontation with an additional large volume of structured and unstructured data from different data sources.
Volatile Requirements. Research projects have clear research goals, however, how to technically reach the goals (e.g. software architecture design) is generally determined during the project. This depends on other factors such as later identified data sources or bad data quality (e.g., unstructured or volatile data) which might change during the project.
Developing Prototypes. Research projects have a strong focus on research results and on answering research questions. Thus, less focus is set on a broad functionality of software solutions. Implementation of the rudimentary functionality is generally realized by developing prototypes. An arising challenge is searching for technologies or methods on the fly, as this would result in a pieced-together solution which might influence scalability or adaptability.
Available Budgets. The financing of research projects

is usually characterized by a predefined budget. Changing that budget, especially in public research projects promoted by federal and state governments, might be connected with higher effort.
Innovative Character. Being innovative is an important factor within research projects. This means new technologies or frameworks need to be tested to achieve new experiences. However, using new innovative technologies, software or frameworks might cause a challenge due to their lesser maturity.
Research Community. Research projects have an open character to an open research community. This means that the published results can be validated and reproduced by other scientists (Y. Demchenko et al., 2013). This requires to produce valuable and meaningful knowledge through a well-defined solution.

3.3 Intensification of Big Data FCs in Research Projects

The FCs of Big Data and of research projects often go hand in hand. Sometimes they influence and in some cases even intensify one another such as the data volume characteristic which is intensified within research projects. Generally, such intensification requires an architecture allowing an easy and quick inclusion of additional data sources. This complication also affects the complexity of data handling throughout the whole project, as it can dynamically in- or decrease with every additional included data source. The ensuring of data veracity is affected in the same sense. Every new data source causes the implementation of new functions, e.g to detect outliers or to clean data. The handling of data velocity faces the aforementioned problems as well. If new data sources are acquired, which offer data of a higher or lower velocity than the sources that are already included in the project, more difficulties arise, such as to ensure a fast data consumption with different velocities.

4 A FLEXIBLE ARCHITECTURE FOR MANAGING BIG DATA WITHIN RESEARCH PROJECTS

This Section presents our architecture which supports managing Big Data (cf. Figure 1) by considering the FCs of Big Data (cf. Section 3.1) and the FCs of research projects (cf. Section 3.2). Generally, the structure of our architecture illustrates the data processing steps from data collection (bottom left), over data cleaning, data aggregation (bottom right) to data presentation (top left). The right side represents how the data is stored, using a data lake and a frontend database for outside requests. Passing on the data between the different data processing steps, the data is stored within data queues. Furthermore, for the separation between the frontend (cf. data presentation) and the backend (cf. data collection, cleaning and aggregation) a proper interface which separates the transfer between the backend and frontend is used.

In the following, we present identified architecture requirements and how they are realized within our architecture. In general, the identified requirements are derived from the described FCs of Big Data, the FCs of research projects and from the literature. They comprise *modularity*, *adaptability*, *scalability*, *well-defined data handling*, *distributed system*, *computing capacity*, *and infrastructure management*.

4.1 Modularity

We identified the modularity as a required feature which means to divide and structure a system into software and hardware modules realized by micro-services. Containers are a common approach as they offer, e.g, virtualization or lightweight operations in comparison to conventional virtual techniques (H. Li et al., 2019). This enables scalability and adaptability of a system and helps coming along with data variety. Therefore, our architecture has a modular design which is achieved by using micro-services. We use the Docker container technology to run each software component as a micro-service. This comprises, e.g., Docker containers for databases, for the frontend or for scripts to collect, clean and aggregate data.

4.2 Adaptability

We identified that being adaptable supports handling of volatile requirements. In general, adaptability describes the ability to modify and extend a system (K. Lehmann and A. Freymann, 2018). This means to be able to change, add or remove hardware, software or technologies such as databases, frameworks or programming languages. This also benefits the development of prototypes due to their innovative character which is known for changes, e.g., technologies or programming languages of the prototype.

In order to support adaptability, we use a standardized syntax for the data format (i.e. JSON) which is used for the data flow between each of the micro-services. Additionally, a standardized query language (at the frontend) is realized by using GraphQL (GraphQL Foundation, 2019) as well as an automated

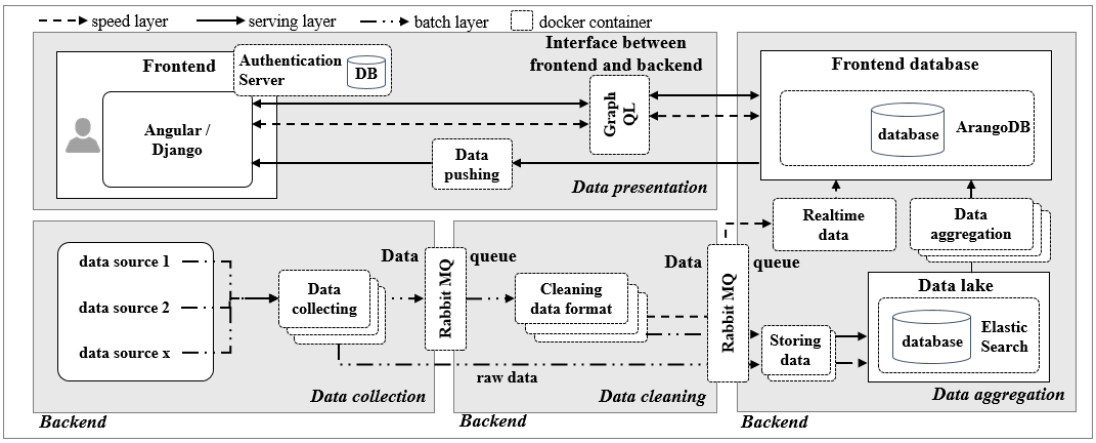

Figure 1: Overview of our architecture.

testing and delivery of Docker images which is realized with Drone (Drone, 2019).

4.3 Scalability

Scalability is an important and required feature within Big Data (Ahmed Oussous et al., 2018), (Volk et al., 2019). Offering scalability supports the expansion of a solution horizontally and vertically by its hardware and software components. This enables to store a large and constantly growing data amount for instance by adding new database nodes and helps coming along with volatile requirements. To realize scalability it is common to have a distributed system with distributed databases and servers to split data processing (S. Kaisler et al., 2013), (Sindhu and Hegde, 2017). We realize the scalability by using a server cluster managed by the Docker Swarm orchestration. To scale the data store, we use Elasticsearch (Elasticsearch, 2019) which allows to arbitrarily spread data and manager nodes over the server cluster.

4.4 Data Handling

A well-defined data processing should comprise the four data processing steps of Big Data (e.g., collection, cleaning, aggregation and presentation). This enables to come along with the FCs of Big Data and with the additional large amount of data related to research projects. Finally, a proper data handling creates a better research result which might get more attention within the research community. Additionally, the aspect of data flows needs to be addressed. Data flow means how the data is transported through the four data processing steps. For the transportation three important matters are recommended: Firstly, packing data into small units simplifies the data processing. Secondly, buffering data packages between data processing steps is a common approach, e.g., by using a message broker to save intermediate results. Thirdly, splitting the data flow into several data layers using the lambda architecture is a required way for Big Data processing (M. Kiran et al., 2015).

The architecture is designed to realize the four data processing steps (cf. Figure 1). Addressing the data collection, for each data source, we realized an individual micro-service running in a Docker container which collects and queues the data using the message broker RabbitMQ (RabbitMQ, 2019). The data gets pulled from the queue by further micro-services for data cleaning (e.g. checking the time format). The data is then embedded within an uniform data structure. Finally, other micro-services store the data within a data lake. Real-time data is directly sent to a frontend database. The stored data within the data lake is then aggregated using different micro-services.

Within our architecture, two databases are chosen: Elasticsearch (Elasticsearch, 2019) as data lake and ArangoDB (ArangoDB, 2019) as frontend database. This separates non-aggregated data (cleaned and raw data) from aggregated data (for the frontend). This also relieves the data lake because requests for aggregated data are only sent to the frontend database. For the data presentation two frameworks are being used: Django (django, 2020) and Angular (Angular, 2019).

4.5 Distributed System

Such systems have become a significant and required role within Big Data (Sindhu and Hegde, 2017), (Katal et al., 2013). The software is running on different interconnected servers. Distributed systems enable load balancing, distribution of computational power, data storage and efficient parallel process-

ing. Our architecture realizes this by using a Docker Swarm server cluster.

4.6 Infrastructure Management

Realizing the aforementioned requirements needs an overall management of the system to get transparency (Peinl et al., 2016). Beside the controlling and monitoring of the system, the orchestration of running Docker containers (e.g. micro-services) is a significant task for such a management (H. Li et al., 2019).

The entire system and the Docker containers are orchestrated by Docker Swarm and we use Portainer (Portainer, 2019) to manage (i.e. configure) the swarm. Portainer manages distributed servers from different locations, Docker container images, related Docker networks and volumes.

4.7 Computing Capacity

A significant requirement and key factor for processing Big Data is to offer appropriate computing capacity (Y. Demchenko et al., 2013). This is often related to parallel processing, especially, to enable real-time data processing (S. Kaisler et al., 2013). This helps to process large amounts of data as well as coming along with a high data veracity. Having appropriate servers with a high computational power, a distributed system can optimize utilization of the computational power by load balancing.

5 EVALUATION

In this work, we adapted and further developed our architecture in a current research project called i-rEzEPT. The project is characterised by processing data from various sources with high velocity. Data types comprise environmental data (e.g. temperature, humidity or cloudiness), telematic data from electric vehicles (i.e. GPS, speed or battery state) and smart metering data (e.g. power inverters from photovoltaic systems or frequency meters).

Table 3 lists the needed data storage in detail for the different data types. To ensure maximum availability, the data gets replicated within the data lake, which doubles the needed capacity. The raw data (cf. Figure 1) is saved in a compressed state, as it does not need to be accessed on a regular basis. Thus, it is expected to require 190GB (assuming a 90% compression ratio) of storage capacity. Aggregations of the different timeseries are expected to need another 200MB of storage. This accumulates to a total expected storage need of around 4.15TB.

Table 3: Storage requirement by data type for 2 years.

Data type	Gigabyte
+ smart metering data	1800
+ Environmental data	67
+ Telematic data	15
+ Raw data	190
= Subtotal	2072
+ With replication	4144
+ Aggregation (only ArangoDB)	0.2
= Total	4,144.2

Our evaluation of the presented architecture focuses, firstly, on the fulfillment of the most important Big Data characteristics and, secondly, of the architecture deployment using Docker Swarm. This deployment is then compared with the additional technology Kubernetes (Kubernetes Authors, 2020) by evaluating the handling of both technologies. This project currently runs on a server cluster with Docker Swarm. It consists of seven Ubuntu 18.04 virtual machines, utilizing a total of 42 CPUs as well as 82GB of RAM. The Docker Swarm shares this hardware with another research project, so it does not have exclusive access to the cluster's ressources. It remains to be seen, whether the architecture performs as well on bigger clusters.

5.1 Big Data Characteristics Evaluation

Variety: Elasticsearch allows to easily add new data sources without concerning the data format. By the end of the project, it is expected to have 25 different data sources, providing data in 18 different formats.
Velocity: The 25 different data sources each provide measurements ranging from two times per second up to once every thirty minutes which gets handled by the message broker. It splits the incoming data streams into easy-to-process data packages and temporarily stores them in queues, until another micro-service pulls them from queues and processes them.
Veracity: This can be checked outside and within the architecture. For some data sources, veracity can be ensured before the data even gets pulled from the API. For other data sources the veracity can be checked during the data cleaning phases. Simple plausibility checks can be performed before storing it in the data lake (e.g. invalid speed values).
Complexity: The evaluation shows that Elasticsearch is suitable for working with data having different data structures. Connections between different data types can easily be represented by adding additional meta-values to the different timeseries and the Elasticsearch query system allows for complex aggregations across multiple indexes.

5.2 Deployment Evaluation

In reference to the Portainer deployment, we evaluated that adding new servers to the Docker Swarm and micro-services for data processing was proven as simple. At the beginning of the project, the cluster comprised five virtual machines. During the project two additional database servers were added to scale the data lake and to set up the frontend database. The cluster also started out with only a couple of micro-services. The number of micro-services has been incrementally expanded by including new data sources, running new data aggregations and adding the frontend. It is expected that the number of micro-services will grow up to around 100 services by the end of the project. Adding these micro-services showed the adaptability of the architecture but also shows its limitations. Our current limitation for an in depth evaluation is the small size of the cluster. Furthermore the clusters scalability and load balancing capabilities are limited by the underlying storage layer since the database nodes are currently pinned to specific virtual machines with additional storage. Focusing Kubernetes and a distributed storage system in our testing deployments allows a single database node to move freely within the cluster and between different virtual machines. Therefore offering a promising solution for the further growing architecture, its scalability and load balancing features.

6 RELATED WORK

In the literature, publications exist which present architectures and frameworks for Big Data. According to challenges related to Big Data, several publications speak about the Big Data characteristics comprising data volume, data velocity, data variety, data value and data veracity (S. Kaisler et al., 2013), (Katal et al., 2013) and (Y. Demchenko et al., 2013). (Katal et al., 2013) as well as (S. Kaisler et al., 2013) added data complexity as an additional Big Data characteristic. In our work, we took these Big Data characteristics as a fundamental scope that needs to be considered.

The content of this work is based on a previous publication which presents an architecture for smart cities in the context of research projects and takes up several architectural design features such as a distributed Event Based System, micro-services and a lambda-architecture for the data handling (K. Lehmann and A. Freymann, 2018). Scalability and flexibility are described as basic features. Our architecture extends this previous work in different parts, e.g., by using a server cluster to distribute the micro-services which significantly enhances the scalability or by a proper orchestration to manage the distributed system. Furthermore, our architecture is designed for small, medium and large research projects.

The publication (Y. Demchenko et al., 2013) presents an architecture called the *Scientific Data Infrastructure (SDI)* which tackles challenges of Big Data in the context of science and also focuses on a general approach for a data lifecycle management in research and industry. The SDI also comprises the data lifecycle from data collection, processing and presentation. An additional micro-service architecture for IoT applications is proposed by (L. Sun et al., 2017) which also has strong consideration for scalability and adaptability by concerning it from a service layer to a physical layer (L. Sun et al., 2017). Furthermore, they address significant challenges which arise with the dynamically growing amount of physical IoT devices. In essence, they propose a system design comprising several core micro-services, a service orchestration and a lightweight communication deployed with Docker and Kubernetes. Additionally (Volk et al., 2019) address difficulties of creating a big data architecture in regards to requirements engineering, the technology selection and the project realization. They provide several references to existing architectures and propose a solution to find a Big Data architecture by utilizing a decision support system.

In comparison to (L. Sun et al., 2017), our architecture has a strong focus on challenges within research projects, presenting a clear comparison and intensification between challenges of Big Data and research projects. In addition, our architecture also considers scalability and modularity as an important feature for such an architecture in order to come along with the mentioned challenges which is missing in (Y. Demchenko et al., 2013). We also offer a concrete proposal how to implement or to deploy the architecture which is evaluated and shown with a current research project. This also stands in contrast to (Volk et al., 2019) who only propose a solution for finding an architecture, not a concrete architecture itself.

7 CONCLUSIONS AND FUTURE WORK

This work presented an architecture which deals with the fundamental challenges of processing Big Data, while also taking the unique characteristics and challenges of modern day research projects into account. Therefore, it supports the handling of Big Data in research projects comprising a huge amount of various high frequency structured, unstructured and complex

data. At the same time it is easily and quickly deployable. This work identified requirements needed to be considered during designing such an architecture comprising e.g. a well-defined data handling, an infrastructure management or scalability. Our architecture is scalable both horizontally and vertically.

A possibility of improving the architecture in the future would be to switch the container orchestration from using Docker Swarm to using Kubernetes. It offers a more robust solution and better fine tuning. It would allow the utilization of a lightweight operating distribution as opposed to the Ubuntu distribution that is currently used, which would free up a non-trivial part of the clusters resources and would reduce management efforts. Another desirable improvement of the architecture would be a more extensive focus on load balancing, synchronization between the cluster's machines and the ensuring of service and data consistency within the cluster. Problems with synchronization and consistency are handled on a code level and should optimally get shifted towards the cluster management as well, wherever applicable.

In conclusion, our architecture allows to easily handle all the aforementioned challenges which have been laid out under Section 4. In addition to that, it is completely made up by open-source solutions, allowing for more freedom in terms of budget allocation.

REFERENCES

Ahmed Oussous, Fatima-Zahra Benjelloun, Ayoub Ait Lahcen, and Samir Belfkih (2018). Big data technologies: A survey. *Journal of King Saud University - Computer and Information Sciences*, 30(4):431–448.

Angular (2019). One framework. mobile & desktop. URL: https://angular.io, accessed 2019-12-17.

ArangoDB (2019). One engine. one query language. multiple data models. URL: arangodb.com, accessed 2019-12-17.

django (2020). django: The web framework for perfectionists with deadlines. URL: djangoproject.com/, accessed 2020-02-18.

Drone (2019). Automate software testing and delivery. URL: https://drone.io/, accessed 2019-12-17.

Elasticsearch (2019). Get started with elasticsearch. URL: elastic.co, accessed 2019-12-17.

GraphQL Foundation (2019). A query language for your api. URL: https://graphql.org, accessed 2019-12-16.

H. Li, N. Chen, B. Liang, and C. Liu (2019). Rpbg: Intelligent orchestration strategy of heterogeneous docker cluster based on graph theory. In *2019 IEEE 23rd Int. Conf. on Computer Supported Cooperative Work in Design (CSCWD)*, pages 488–493.

K. Lehmann and A. Freymann (2018). Demo abstract: Smart urban services platform a flexible solution for smart cities. In *2018 IEEE/ACM Third Int. Conf. on Internet-of-Things Design and Implementation (IoTDI)*, pages 306–307.

Katal, A., Wazid, M., and Goudar, R. H. (2013). Big data: issues, challenges, tools and good practices. In *2013 Sixth int. conf. on contemporary computing (IC3)*, pages 404–409.

Kubernetes Authors (2020). Production-grade container orchestration: Automated container deployment, scaling, and management. URL: kubernetes.io, accessed 2020-02-21.

L. Sun, Y. Li, and R. A. Memon (2017). An open iot framework based on microservices architecture. *China Communications*, 14(2):154–162.

M. Kiran, P. Murphy, I. Monga, J. Dugan, and S. S. Baveja (2015). Lambda architecture for cost-effective batch and speed big data processing. In *2015 IEEE Int. Conf. on Big Data (Big Data)*, pages 2785–2792.

Peinl, R., Holzschuher, F., and Pfitzer, F. (2016). Docker cluster management for the cloud - survey results and own solution. *Journal of Grid Computing*, 14(2):265–282.

Portainer (2019). Making docker management easy. URL: portainer.io, accessed 2019-12-17.

RabbitMQ (2019). Understanding rabbitmq. URL: rabbitmq.com, accessed 2019-12-17.

S. Kaisler, F. Armour, J. A. Espinosa, and W. Money (2013). Big data: Issues and challenges moving forward. In *2013 46th Hawaii Int. Conf. on System Sci.*, pages 995–1004.

Sindhu, C. S. and Hegde, N. P. (2017). Handling complex heterogeneous healthcare big data. *Int. Journal of Computational Intelligence Research*, 13(5):1201–1227.

Stucke, M. E. and Grunes, A. P. (2016). *Big data and competition policy*. Oxford University Press, Oxford, 1st edition edition.

Tanenbaum, A. S. and van Steen, M. (2007). *Distributed Systems: Principles and Paradigms*. Pearson Prentice Hall, Upper Saddle River, NJ, 2 edition.

Verma, A., Pedrosa, L., Korupolu, M., Oppenheimer, D., Tune, E., and Wilkes, J. (2015). Large-scale cluster management at google with borg. In *Proceedings of the Tenth European Conf. on Computer Systems*, page 18.

Volk, M., Staegemann, D., Pohl, M., and Turowski, K. (2019). Challenging big data engineering: Positioning of current and future development. In *Proceedings of the 4th Int. Conf. on Internet of Things, Big Data and Security*, pages 351–358. SCITEPRESS - Science and Technology Publications.

Y. Demchenko, P. Grosso, C. de Laat, and P. Membrey (2013). Addressing big data issues in scientific data infrastructure. In *2013 Int. Conf. on Collaboration Technologies and Systems (CTS)*, pages 48–55.