Decision Support for the Technology Selection in Big Data Projects: An End-to-End Approach

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

angenommen durch die Fakultät für Informatik der
Otto-von-Guericke-Universität Magdeburg

von **Matthias Volk, M. Sc.**
geboren am 05.04.1991 in Magdeburg

Gutachter:
Prof. Dr. Klaus Turowski
Prof. Dr. Alexander Zeier
Prof. Dr. Vijayan Sugumaran

Magdeburg, den 4. November 2022

„Part of the journey is the end.“
— T. Stark (2019)

Abstract in English

For more than a decade now, data has been considered as the new digital oil when it comes to IT projects. Be it as a basis for the development of new ideas and business processes or the optimization and improvement of existing processes, products, or other tangible and intangible assets, data is of incalculable value today, both in business and science. The successful collection, persistence, processing, and visualization of this data is essential for every IT project that deals with measured data. Despite or even because of this fact, implementing such data-intensive projects, usually referred to as Big Data, is not a trivial undertaking. Very extensive and specialized knowledge is often required, which goes far beyond the actual application fields, such as Industry 4.0, Data Science, Internet of Things, Machine Learning, or Artificial Intelligence. In addition to fundamental principles dealing with the successful implementation of such projects and thus with the project management, this also includes selecting and combining the appropriate technologies for the intended application.

This issue is particularly intensified by the steadily increasing number of novel solutions that can be assigned to this area. Numerous scientific papers provide information on how potential new technologies can look and how they can be combined to form complex entities. It is therefore not surprising that today, more than ever, experts who are experienced in planning and implementing such data-intensive projects and creating corresponding system architectures are needed. However, a holistic understanding coupled with specialist expertise is rare, meaning that expert knowledge often has to be acquired independently in a cumbersome and time-consuming manner or procured externally at a high cost, either in the form of consultants or additional training courses, or teaching materials. In addition to the constant change and the emergence of new solutions, this further complicates the prevailing problem.

In the present work, based on the application of the Design Science Research methodology, a holistic decision support approach is to be delivered, which aims to help potential users to plan their Big Data projects from end to end. Starting from initial ideation, requirements are to be developed, the meaningfulness of an application of Big Data technologies is to be determined, and their specific selection and combination are to be supported. The envisaged component-based structure makes it possible to use the obtained results for the further modeling of the corresponding architectures as well as their deployment, for which also approaches are examined in the present work. The created framework and the instantiation of this in the form of an extensive prototype are extensively evaluated at the end according to well-known methods . The goal of the work is not only to support such undertakings in their critical start-up phases but also to be able to uncover and convey targeted information in the area of Big Data.

Abstract in German

Seit nunmehr einem Jahrzehnt gelten Daten als das neue digitale Öl, wenn es um die Durchführung von IT-Projekten geht. Ob als Basis für die Entwicklung neuer Ideen und Geschäftsprozesse oder zur Optimierung und Verbesserung von Prozessen, Produkten oder anderweitigen materiellen und immateriellen Gütern, Daten sind heute von unschätzbaren Wert, sowohl in Wirtschaft als auch Wissenschaft. Die erfolgreiche Sammlung, Speicherung, Verarbeitung und Visualisierung dieser ist dabei heute für jedes IT-Projekt, welches sich mit maßenhaften Daten beschäftigt, von essenzieller Wichtigkeit. Trotz oder gerade wegen dieses Umstandes stellt die Durchführung solcher datenintensiven, meist als Big Data-bezeichnete, Projekte jedoch kein triviales Unterfangen dar. Oftmals wird ein sehr umfangreiches Spezialwissen benötigt, welches weit über die eigentlichen Anwendungsfelder, wie Industrie 4.0, Data Science, Internet der Dinge, Maschine Learning oder künstliche Intelligenz, hinausgeht. Neben allgemeinen Grundprinzipen, die sich mit der erfolgreichen Durchführung solcher Projekte und damit dem Projektmanagement beschäftigen, gehören dazu auch die Auswahl und Kombination der für den Anwendungszweck richtigen Technologien.

Verstärkt wird diese Problematik insbesondere durch die stetig steigende Anzahl neuartiger Lösungen, die sich diesem Bereich zuordnen lassen. Zahlreiche wissenschaftliche Abhandlungen geben dabei Aufschluss darüber, wie potentielle neue Technologien aussehen und diese zu komplexen Gebilden zusammengesetzt werden können. Es erscheint deshalb nicht verwunderlich, dass heute mehr den je Experten benötigt werden, die sich mit der Planung und Durchführung solcher datenintensiven Projekte, sowie der Erstellung entsprechender Systemarchitekturen auskennen. Das ganzheitliche Verständnis gepaart mit fachspezifischem Wissen ist jedoch rar gestreut, wodurch oftmals mühevoll und aufwendig Expertenwissen eigenständig angeeignet oder kostenintensiv fremdbeschafft werden muss, sei es in Form von Beratern, zusätzlichen Schulungen oder Lehrmaterialien. Zusätzlich zu dem stetig fortschreitenden Wandel und dem Aufkommen neuer Lösungen wird dadurch die herrschende Problematik weiter verkompliziert.

In der vorliegenden Arbeit soll basierend darauf, durch Einsatz der Design Science Research Methodologie, ein ganzheitlicher Entscheidungsunterstützungsansatz geliefert werden, der potentiellen Anwender dabei helfen soll ihre Big Data Projekte von Ende-zu-Ende zu planen. Ausgehend von einer initialen Ideenfindung sollen so, gezielt Anforderungen ermittelt, die Sinnhaftigkeit eines Einsatzes von Big Data Technologien ermittelt, sowie deren spezifische Selektion und Kombination unterstützt werden. Die dabei anvisierte komponentenbasierte Struktur ermöglicht es die gewonnen Resultate dann wiederum für die weitere Modellierung der entsprechenden Architekturen sowie deren Bereitstellung zu nutzen, zu denen auch Ansätze in der vorliegenden Arbeit untersucht werden. Das erschaffenen Framework, sowie die Instanziierung dieses, in der Form eines umfangreichen Prototypen werden zum Ende umfangreich nach anerkannten Methoden ganzheitlich evaluiert. Das Ziel der Arbeit ist es solche Unterfangen nicht nur in ihren kritischen Startphasen zu unterstützen, sondern gezielt Informationen im Bereich Big Data aufdecken und vermitteln zu können.

Acknowledgments

I have to thank a lot of people, without this work would never have been possible.

First of all, I would like to express my sincere gratitude to my supervisor Prof. Dr. Klaus Turowski, who gave me constant support, guidance and suggestions throughout this project. I also would like to thank my other supervisors Prof. Dr. Alexander Zeier and Prof. Dr. Vijayan Sugumaran for their constant recommendations and guidance.

Many thanks also go to all of my colleagues who have accompanied and supported me during my time. Special thanks go to Daniel Staegemann for his countless comments and remarks during proofreading. Although these have led to many sleepless nights, his support has made the work what it is today.

I would like to express my deepest gratitude to my parents Annett and Ronald, as well as my brother Martin for their unwavering support and encouragement, which gave me strength not only during the last six years leading up to the completion of the doctoral thesis, but also along the way.

Finally, this endeavor would not have been possible without my wife Vivien and my children, who have tirelessly supported and motivated me, even in difficult and stressful times. Despite the little time I was able to give you recently, you always showed patience and motivated me with loving and encouraging words, until the very end. *Thank you*!

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

AA ..... Automation Acting

AG ..... Data Aggregation

AHP ..... Analytical Hierarchy Process

ANP ..... Analytical Network Process

API ..... Application Programming Interface

ASUM ..... Analytics Solutions Unified Method

AV ..... Availability

BC ..... Book Chapter

BDA ..... Big Data Architecture

BDE ..... Big Data Engineering

BDEP ..... Big Data Engineering Process

BDTOnto ..... Big Data Technology Ontology

BigDAF ..... Big Data Complexity Framework

BMP ..... Bitmap

BP ..... Batch Processing

BPD ..... Business Process Diagram

BPMN ..... Business Process Model and Notation

C ..... Cost

CA ..... Conference Article

CC ..... Computational Complexity

CF ..... Data Classification

CI ..... Consistency Index

CI/CD ..... Continuous Integration/Continuous Development

CL ..... Data Cleaning

CM ..... Cluster Management

CP ..... Consistency Preservation

CR ..... Consistency Ratio

CRISP-DM ..... Cross Industry Standard Process for Data Mining

CSV ..... Comma-Separated Value

CU ..... Data Clustering

DA ..... Data Analysis

DC ..... Dublic Core

DD ..... Data Preparation

DECIDE ..... Decision Support for Big Data Projects

DI ..... Data Ingestion

DM ..... Data Mining Algorithm Support

DMI ..... Data Magnitude Index

DO ..... Documentation and Support

DOLCE ..... Descriptive Ontology for Linguistic and Cognitive Engineering

DP ..... Data Preparation

DS ..... Data Selection

DSR ..... Design Science Research

DSS ..... Decision Support System

ELECTRE ..... ELimination Et Choix Traduisant la REalité

EP ..... Event-Data Processing

F ..... Data Formatting

FR ..... Functional Requirement

FS ..... Flexibility and Scalability

FT ..... Fault Tolerance

GCP ..... Google Cloud Platform

GIS ..... Geoinformation System

GSM ..... Global System for Mobile Communications

GUI ..... Graphical User Interface

HANA ..... High-Performance Analytic Appliance

HDFS ..... Hadoop Distributed File System

IAO ..... Information Artifact Ontology

IDE ..... Integrated Development Environment

IM ..... Installation and Maintenance Effort

INCOSE ..... International Council on Systems Engineering

ISO ..... International Organization for Standardization

JA ..... Journal Article

JPEG ..... Joint Photographic Experts Group

KDD ..... Knowledge Discovery in Databases

KVM ..... Kernel-based Virtual Machine

MADM ..... Multi-Attribute Decision Making

MCDM ..... Multi-Criteria-Decision-Making

MH ..... Message Handling

ML ..... Machine Learning

MO ..... Monitoring

MODM ..... Multi-Objective Decision Making

NBDRA ..... NIST Big Data Reference Architecture

NFR ..... Non-Functional Requirement

NoSQL ..... Not Only Structured Query Language

NP ..... Near Real Time Processing

NTCP ..... Novelty, Technology, Complexity and Pace

OMG ..... Object Management Group

OS ..... Operating System

OWL ..... Ontology Web Language

P ..... Data Pipelining

PNG ..... Portable Network Graphics

PoC ..... Proof of Concept

PoU ..... Proof of Use

PP ..... Parallel Processing

PROMETHEE ..... Preference Ranking Organization Method for Enrichment Evaluation

R ..... Reliability

RC ..... Recovery Mechanics

RDF ..... Resource Description Framework

RE ..... Regulations

RI ..... Randomized Consistency Index

RM ..... Resource Management

RP ..... Reporting

RT ..... Real Time Processing

RV ..... Rating Value

S ..... Sustainability

S.T.A.D.T ..... Strategy Time Analytics Data Technology

SC ..... Storage Capacity

SD ..... Store Structured Data

SE ..... System Engineering

SEMMA-DM ..... Sample, Explore, Modify, Model, and Assess Method

SHA ..... Secure Hash Algorithm

SL ..... Support Scripting Language

SO ..... System Operation

SP ..... Streaming Processing

SQL ..... Structured Query Language

SRQ ..... Sub-Research Question

SS ..... Store Semi-Structured Data

ST ..... Data Streaming

SU ..... Store Unstructured Data

SUC ..... Standard Use Case

SUMO ..... Suggested Upper Merged Ontology

SWO ..... Software Ontology

SY ..... Security

TOPSIS ..... Technique for Order Preference by Similarity to Ideal Solution

UI ..... User Interface

UML ..... Unified Modeling Language

V ..... Data Visualization

VM ..... Virtual Machine

WWW ..... World Wide Web

XML ..... Extensible Markup Language

YAML ..... Yet Another Markup Language

YARN ..... Yet Another Resource Negotiator

ZB ..... Zettabyte

1 Introduction

In this chapter, an extensive motivation for the research problem is given, which is intended to highlight the importance of the underlying work and to expose the research gap. Subsequently, the main hypotheses and the research questions are described. The scientific methods used for this purpose, their application in detail, and the structure of the work are presented afterward. In the end, a detailed overview of all relevant published articles is given.

1.1 Motivation

"Data is the new oil. It's valuable, but if unrefined it cannot really be used" [Hau02]. The headlines that today´s data can be described as the new oil recurrently circulates on the internet [Rus03; Mar24], research articles [JNL19], or even official documents of the European Parliament [Szc20] for many years. Although this striking title is constantly used to motivate new data-related concepts and topics, such as the importance of social media or data governance, at the core, it highlights the relevancy of sophisticated algorithms, paradigms, and technologies that can be used for the *refinement* and thus the value proposition. In this context, big data is almost always named, encompassing the correct craftsmanship to handle the constantly increasing data for more than a decade. Initially standing for the sheer size of the data in the early 2010's [MCB+11], the term has changed considerably in the past few years. This refers to the data characteristics that are being used and the general nature itself. Today, big data concerns rather the absence of sufficient traditional technologies and the necessity of sophisticated concepts and solutions that are capable of managing and processing massive amounts of differently structured data that may income even in real-time [CG19a].

Consequently, a multitude of different *big data* technologies emerged within the last decade that shall help with the realization of data-intensive projects, forming concurrently the (technical) foundation for numerous related domains [OBA+17; PZ14a], like internet of things (IoT), deep learning [AHN+20], industry 4.0, data science, and artificial intelligence. Hence, it became natural that big data solutions are being utilized in a wide variety of application areas [VST+20], such as agriculture [BK16; KKP17], healthcare [WKC+14; ABC+19; BZA+19; RR], tourism [ABS19; FHL14; Gaj19; LXT+18], transportation [EEE+18; FSW+18; FEP+19; LDK+14] and construction [BOQ+16]. Re-

searchers and practitioners are aware of the great potentials that come with the realization data-intensive projects. Among other things, this includes the discovery of new business models, the improvement of existing processes and the associated achievement of a general competitive advantage over the competition, for instance, by improving existing products and services [CP16; MFV18].

Even though big data has existed for a long time, most of the conducted data-intensive projects fail, never reach a productive status, or are at least less successful [Hot21]. The reasons for this are manifold. In [KNZ18] a lack of requirements engineering in the early stages is stated as one of the main reasons. Because often it can be noticed that the capabilities of the later systems differ too much from the user's expectations. The same is reported by David Becker, in his research article [Bec17]. Here it is amongst other reasons mentioned that those projects fail because of incorrect business objectives. Furthermore, the complexity of the technologies, their incorrect use, and the missing skills for realization are some of the leading causes of a potential failure. This ambivalent situation, in which a high willingness of potential users exists while the general success of such projects is lacking, can be traced back to the opacity and complexity of existing technologies, the shortage of big data experts, and the absence of established standards.

Due to a large number of available technologies, it is difficult to gain an overview and decide which of these might be suitable for a particular endeavor. Hence, it is not surprising that *"Is it Pokemon or Big Data?"* projects have emerged, asking users to determine whether a given name refers to a big data technology or a Japanese pocket monster [Git22a]. Organizations are looking for talents capable of filling the missing knowledge along with the project's realization to prevent a potential project failure. As a result, in just a few years, the need for experts has increased to such an extent that the training of their own staff is no longer sufficient enough to fill up current demands. Instead, they have to compete for the attention of external talents [GAR+18; Kru16]. At this point, not all companies can keep up to the same pace. Small and medium enterprises (SMEs), in particular, which are only sparsely able to raise the necessary financial resources, face a major problem here [LRT+26].

The worldwide COVID-19 pandemic reinforced this situation. While some industries had to cope with a severe drop in sales and layoffs, the need for IT skills and related people increased more than ever [Sei05]. Even in pre-pandemic times, extensive studies and analyses were carried out that highlighted the needs of IT professionals, also in the area of big data [GAR+18; Kru16]. In [GAR+18], an analysis of numerous job offerings in this field was carried out, revealing that, above all, there is a need for people who are familiar with the technologies and their selection as well as with the creation of the corresponding systems.

While many different approaches exist that attempt to describe and deliver information to help with the realization of big data projects partially, most of them are unique in their applicability, which is especially showcased by the uniqueness of previously referred

use cases. For this reason, a structured approach that supports both the identification of the necessary expertise and the planning of such projects, from *end-to-end*, would be sensible.

1.2 Research Questions

While the domain of big data and adhering disciplines are well researched, the holistic view from end-to-end is rarely employed. This is particularly evident in the planning, development, and implementation of corresponding systems. Non-experts confronted with the endless abundance of information, concepts, and technologies are therefore faced with a diffuse picture regarding implementing such projects [KNZ18]. However, enriched with domain-specific knowledge, established approaches from adjacent fields of project management, system engineering (SE), and data mining can be helpful, allowing a more comprehensive assessment. These include, for instance, the identification of the reasonability of a big data technology application, the development of big data-related requirements, multi-criteria decision-making for a technology selection, system modeling, or a prototypical set-up of respective systems. Most of them also denote the required skills requested from companies in their big data-related job offerings [Kru16]. As a result, this can provide a detailed picture of the potential solution and the knowledge required to increase the overall success, even for non-experts. Based on this and the previous discussion, the first hypothesis of this thesis is:

Hypothesis 1
The success of a data-intensive or big data project for non-experts can be increased by utilizing well-known principles from SE, project management, and data mining domains. A thorough understanding of big data projects can be achieved by identifying the general reasonability of a potential big data technology application, the selection of potential technologies, the architecture modeling, and their deployment. Decision-makers, independent of their domain-specific expertise, can use this information to identify initial recommendations for engineering and integrating related systems. Thus, relevant (expert) knowledge related to big data system components, the composition of those as well as required interfaces and requirements for integration, can be acquired.

Due to the complexity of the envisaged steps and the extent of the domain itself, it can be assumed that the manual processing of related information would only be possible to a certain degree and, beyond that, extremely time-consuming. This becomes obvious when simply looking at the existing technology sets, whose sequential and manual comparison can sometimes be cumbersome and tedious, regardless of the type and quantity of the criteria to be considered. All further supplementary steps, including possible modeling of system architectures and their distribution, are to be considered similarly. In this sense, a

computer-based solution is conceivable, at which a decision support system (DSS) forms a sensible technical foundation [VSB+20b; VSP+19]. Emerging from this observation, the mentioned solution may help in significant parts related to big data engineering (BDE) activities. Hence, the following second hypothesis is discussed throughout this thesis:

Hypothesis 2
A DSS can support a structured technology recommendation procedure for big data projects, whenever sophisticated decisions need to be made that are not manually feasible in a short period of time, especially for non-expert users. Typically, related BDE activities are based on a multitude of different information and calculations, at which numerous interconnections and dependencies have to be considered. Encompassing support can be facilitated through computer-aided assistance, allowing integration for further SE relevant steps, including modeling, deployment, and operation. Hence, an end-to-end big data project instantiation using a computer-supported solution is possible in the form of a DSS.

There are two main objectives in answering these hypotheses. The first goal is conceptual and looks at a wide range of potential steps for the realization of such projects. Essential engineering activities, which are specific for the technologies to be used, deal with the project initialization, technology selection, and system creation. The second goal is primarily a prototypical implementation, which allows supporting a multitude of related steps and sub-steps. Non-experts in this field will benefit from a general framework and a computer-based solution to help them. In summary, the research goal of this thesis is:

Research goal
Design, develop and implement an end-to-end procedure that supports researchers and practitioners in realizing their big data projects. The planned procedure should range from the general identification of the overall meaningfulness of a big data technology application, through the selection of appropriate technologies, to the provision of sandbox environments. A DSS is developed that provides decision options, guidance, and automation throughout the end-to-end procedure to reduce and simplify the steps that need to be conducted by a potential user.

In order to approach this goal, the concept of a research question is to be considered, which in the scientific field contributes significantly to the achievement of a possible solution and measures for its evaluation [HMP+04]. Therefore, the following research question is in the focus of the thesis, whose answer should lead to the conceptual as well as prototypical implementation:

Research question
How can end-to-end decision support be facilitated concerning big data engineering activities that assists decision-makers with selecting, combining, and deploying big data technologies in their projects?

Researchers and practitioners have been relying on tools, techniques, and best practices that support them in realizing their projects for many years. Hence, it becomes for a potential solution mandatory to not only cover prominent and vital aspects that may positively affect the sophisticated procedures. At the same time, relevant elements that are unique to the targeted domain need to be considered. Therefore, additional sub-research questions are formulated that contribute to the overarching goal and thus the answering of the leading research question. Most importantly, an initial overview of related activities and their structured sequence is needed before developing a stepwise procedure and a prototypical implementation. Since BDE coined to be as the primary discipline when it comes to the creation of related systems and, thus, the successful realization of big data projects, the first sub-research question (SRQ) can be formulated as follows:

Sub-Research question 1
Which steps are required that support decision-makers in realizing their big data endeavors?

The heart of the big data system is formed by novel technologies, which are capable of handling massive amounts of differently structured data, sometimes originating and processing at high-speed [GH15; CG19a]. As a crucial step of SE procedures, in *classic* IT systems, the selection of appropriate technologies denotes one of the cornerstones of a successful architecture. Based on the underlying requirements that the system shall fulfill, those can affect the provided functionalities themselves and how the system may interact with other components or the user, respond to external requests, and other non-functional aspects. Emerging out of the drastic evolution that occurred within the last decade, the nature of the requirements for the selection of relevant big data technologies needs to be examined.

Sub-Research question 2
Which requirements need to be considered when it comes to the selection of appropriate technologies for a big data system?

By considering the effort required for the general understanding of the domain itself and related technologies, their selection, and setup, the realization of those projects can be time-consuming. Especially for people that are no experts and host only little to no knowledge about the particularities and technical expertise, the utilization of a computer-supported solution appears to be desirable. In this context, essential elements need to be

identified that comprehensively cover and support appropriate steps along with a potential end-to-end procedure.

Sub-Research question 3
What elements are required to create a computer-supported solution to provide decision support for big data projects?

Furthermore, the relevant elements and their composition can be a cumbersome task. Although best practices for developing such a computer-supported solution may exist, particularities that may increase the complexity of creation need to be recognized. Hence, it is not only required to identify potential elements of a computer-supported solution but also in which way those could be connected, implemented, and utilized, with the lowest possible effort, including at least some very basic actions in a semi-automated way.

Sub-Research question 4
How could a computer-supported solution be designed and utilized that allows a semi-automated selection and deployment of big data technologies in related projects?

In the following, the methodology shall be presented, which was used as a baseline to answer these questions.

1.3 Methodology

In today's world, information systems have a socio-technical character [BS11] and their application only provides added value through human interaction and an organizational context, which is also the case for big data systems [SVJ+19]. As a result, two main research approaches have been established in the field of computer science, which comprise behavioral and constructional scientific methodologies. While the first deals with the handling and empirical observation of processes, behaviors, and interactions with such, the latter aims to create novel artifacts that extend and expand the boundaries of existing perceptions [HMP+04; PTR+07].

The design science research (DSR) methodology describes a constructional scientific research methodology. Through the use of formal methods in the creation, application, and verification of artifacts, this paradigm is, inter alia, applied to find solutions to organizational problems as they become increasingly prominent in the field of business informatics [Sta19]. The artifacts, which are constructed in the form of models, frameworks, prototypes, or other solutions, often address complex problems that involve different perspectives. This means not merely using existing theory and established methods, procedures, or best practices, instead, the environment itself creates the business needs for such a solution. It encompasses the organizations, related people, but also technological concepts. Hevner et al. [HMP+04] proposed a framework that considers all of the aforementioned

aspects and denotes the dependencies between the intended research activity, the existing knowledge base, and the environment. An instantiation of the framework, which comprises all relevant information of this work, is shown in Figure 1.1.

By taking an in-depth look, one can note that the complex research endeavor addresses a multitude of different domains, covering topics such as big data, data-intensive systems, project management, and system architectures. At the same time, established approaches and methodologies are required that deliver essential details about well-known principles, procedures, and guidelines of those topics that can be harnessed to find and improve a potential solution. In particular, this refers to big data-, system- and ontology engineering as well as multi-criteria decision making. As a result, the entire environment will benefit in such a way that constant decision-support along the realization of big data projects can be expected. Appropriate steps, such as planning, technology identification, system modeling, SE, and deployment, can be assisted independently, whether standalone or interwoven in complex procedures, eventually leading to process optimizations, cost reductions, and many more other benefits. Notwithstanding that, other methods are required to facilitate such a complex undertaking. This does not only incorporate the observation of the environment itself or the investigation of further literature. Instead, a sophisticated baseline is required for the stepwise creation, ongoing assessment, and refinement of single elements.

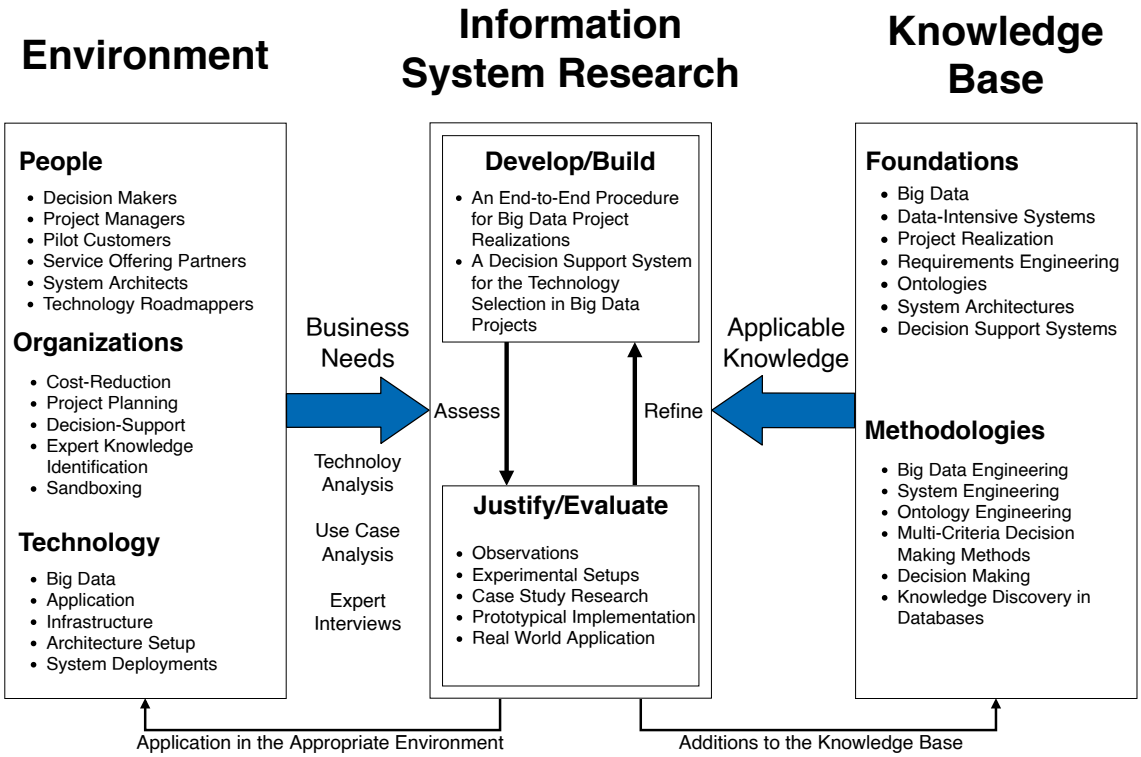


Figure 1.1: The DSR methodology environment of the contribution at hand

Due to this situation, it doesn't seem to be surprising that many different authors took advantage and delivered best practices and guidelines, attempting to reveal how

these endeavors can be conducted in a structured manner. Although approaches, such as [NCP90; SV12a; VK15; PTR+07] exist that differentiate not only terms of the provided procedure but also the overarching scope, all of them share a similar principle at which theory and practical building are almost equally considered [DVS+22]. One of the most frequently cited DSR approaches serves as the methodological foundation for this project. The recommended procedure by Peffer et al. [PTR+07] provides six essential steps, which are required to conduct DSR successfully.

Based on a problem-centered approach, within the first step, Identify *Problem & Motivate* the problem is highlighted, and the value of a potential solution is justified. This is done by stressing the complexity of big data projects and the effort required to realize them. Especially the selection and combination of related technologies require a comprehensive understanding of the domain itself and associated activities. Emerging out of this, the objectives of a solution are inferred. Here, end-to-end decision support shall be facilitated that helps researchers and practitioners from industry and academia to conduct their big data projects. Due to the extent of this intended solution, a computer-supported approach is aimed at that greatly assists during this procedure. Notwithstanding that, before the actual engineering of a potential system takes place, the framework itself is required. This results from existing theory, observations, and previously made findings in the adhering step *Design and Development*.

The following *Demonstration* represents an instantiation and application of the solution in a detailed manner. Here, the developed framework consisting of the stepwise procedure and the detailed insights of each required activity is implemented and showcased in a prototypical DSS. Before the results are communicated, which was done so far via the published research articles and the doctoral thesis at hand, the *Evaluation* as the fifth step is required. Here, through the use of various metrics and analyses, such as mathematical proofs, experiments, expert interviews, and organizational implementations, this activity is used to *"observe and measure how well the artifact supports a solution to the problem"* [PTR+07]. For this particular step, a use case-based application and expert interviews are conducted.

As one can note, within the DSR less effort is spent for the actual evaluation, compared to the design and development, due to this Sonnenberg and vom Brocke proposed in their research article [SV12a] the design-evaluate-construct-evaluate procedure. It consists of four different evaluation activities: *Eval 1, Eval 2, Eval 3, and Eval 4*. These are concurrently applied to the realization of the DSR to facilitate an evaluation of the artifact ex-ante and ex-post. *Eval 1* predominantly focuses on the validation of the purpose and scope of the DSR, to ensure that a meaningful value is created. *Eval 2* assesses the construction of the artifact. Since in most of the cases an evaluation will be only performed after the development of the desired solution, here the construction and needed design specifications are checked. Inter alia, this includes the design objectives, tools, methodologies, and specifications. *Eval 3* denotes the initial ex-post evaluation step, in which

the demonstration and investigation of the correctness of the instantiated artifact are observed. Here, the aforementioned evaluation within the recommended workflow by Peffers et al. [PTR+07] is conducted, for instance, depicted by a proof of concept (PoC). The last of the four patterns focuses on the actual implementation within an organizational context, to *"ultimately show that an artifact is both applicable and useful in practice"* [SV12a], indicating the proof of use (PoU). In conformance with the recommended application of the patterns, the individual Eval steps as well as the conducted DSR procedure itself, together with the main outcomes of each step, are depicted in 1.2.

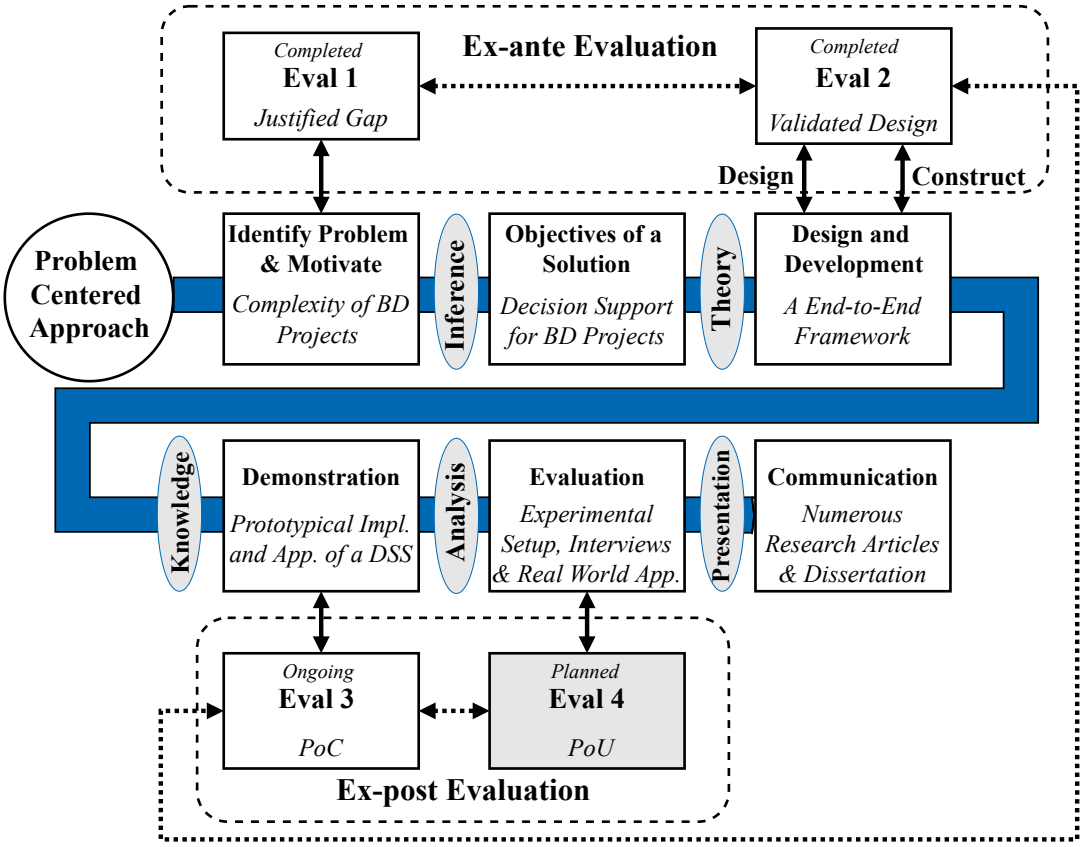


Figure 1.2: Workflow of the conducted DSRM based on [PTR+07; SV12a]

To overcome existing problems in terms of the ex-ante and ex-post evaluation as some constructivists, such as [VPB16] criticize them, additional evaluations are conducted for each identified system element intermediately. In the end, to additionally communicate the DSR as a whole, a *DSR grid*, based on the approach of [VM19], is provided. It is intended to help with *planning and communication of DSR projects*.

Further research methodologies which were used in the already published contributions, relevant for this doctoral thesis, comprise, among others, structured literature methodologies [LJ06; WR02; Fin13; May10], use case analysis [Yin09], as well as specific system development and engineering methodologies [Ebe14; NCP90; Pow02; TAL05;

FGJ97a; UG96]. Most of them are additionally highlighted in the respective sections.

1.4 Structure

Emerging out of the used methodologies, the structure of this work is oriented on the recommended workflow by [PTR+07] as it is depicted in Figure 1.2. Within chapter 1, the overall problem situation and motivation were given, which highlighted the prevailing gap and the importance of a potential solution. The research questions as well as the derived objectives of this work, were further described in detail. To design and develop a suitable solution for the intended research endeavor, a thorough overview of the current state of the art and needed theory is required. In conformance with the used methodology, this is realized within the subsequent chapter 2 - *Research Background*. In here, essential definitions and information of all contiguous domains and sub-domains are described. Remarkably, the topics of SE, big data, ontologies, and DSSs are introduced, as well as in which way they are related to each other.

Chapter 3, *Design of an End-to-End Procedure Supporting the Realization of Big Data Projects*, aims to answer the first sub-research question. As the name of the chapter implies, its main goal is the creation of a comprehensive and detailed procedure that uncovers the required steps for conducting big data projects from *end-to-end*. In doing so, all of the discussed findings from the second chapter are taken into consideration. Based on those, essential requirements are developed and compared to existing approaches. The conclusions made are then taken to develop the addressed procedure. In the end, essential activities are analyzed, and components identified that offer the opportunity for computer-aided assistance.

Following that, in the main part of this work, chapter 4 - *A Decision Support System Framework for Big Data Projects*, all of these components are investigated in more detail. In doing so, a significant part of the already conducted research (cf. Table 1.1), described in the aforementioned sub-chapter, is integrated for the further conceptualization and alignment of the relevant components and activities. Culminating out of this, a comprehensive framework is described, at the very end of this chapter, to deliver the main research artifact of this work [PTR+07; HMP+04]. It comprises not only the essential steps for realizing a big data project but also intermediate features and elements for a computer-based solution, as a DSS.

The instantiation of this framework and, thus, the prototypical implementation is done in chapter 5. Based on the recommended methodologies of DSSs and SE, each layer is described separately. After that, the main functionalities are further presented, and a demonstration of actual usage is showcased. Although each of the developed components is individually evaluated and the results are shared within the fourth chapter, a detailed evaluation of the DSR and artifact is done within chapter 6, titled *Evaluation*. Besides describing an experimental setup and expert interviews, the different *Eval* steps and the

DSR grid are introduced. As a result, the final artifact, the conducted DSR itself, as well as each of the components shall be sufficiently evaluated to ensure that the intended business need can be sufficiently fulfilled. The work ends with concluding remarks in the 7th chapter, at which a thorough discussion and an outlook on the future research are given. Even though each chapter, except the first, closes with a summary, all of the performed steps and made findings are highlighted and discussed here.

1.5 Contributions of the Author

The thesis culminates the put effort of a comprehensive six-year research project. During this time, a multitude of 58 research contributions were achieved. This includes not only many conference articles (CA), but also several journal articles (JA) and book chapters (BC). In the following, the relevance of many of those is described that contribute in great parts to the thesis at hand, especially in the context of the already mentioned structure of this work. The impact of the individual research contributions is additionally addressed within the introduction of each of the subsequent chapters.

The starting point of the present project is the first published research article *How much is Big Data? A Classification Framework for IT Projects and Technologies* [VHB+16], in 2016, in which the complexity of those endeavors was investigated from a data-driven perspective, to identify the overall reasonability of a big data technology application. In particular, the characteristics of the relevant data of a potential big data system were examined using a specific framework and an adaptive calculation of the arithmetic mean. It became apparent here that the general success of a potential big data project may greatly benefit from such kind of decision support, even though only the data to be processed and their characteristics are observed. However, it was noted that even more sophisticated approaches that incorporate existing project requirements are required to identify the relevant data characteristics that, in turn, allow the determination of the overall sensibility of a big data technology application. Using existing contributions and concepts, including also the one mentioned above, in *Ask the Right Questions: Requirements Engineering for the Execution of Big Data Projects* [VJT17] a comprehensive requirements engineering approach was presented that was proposed to assist researchers and practitioners with the planning of their intended big data projects.

The article *Providing Clarity on Big Data Technologies - A Structured Literature Review* [VBT17] comprises a thorough literature review, which was performed to identify relevant concepts, metrics, and ideas to classify big data technologies and also initial criteria for the selection of those. The paper itself greatly contributes to the research background, discussed in chapter two of the thesis, with the focus on existing technology classifications. At the time of the publication of this paper, no satisfactory classifications could be found that allow a possible structuring and presentation of relevant tools and technologies. For that reason, a first attempt, which addressed the lack of existing classification approaches,

was achieved after a meticulous analysis of existing concepts through an ontology. This approach was presented in *Classifying Big Data Technologies - An Ontology-based Approach* [VPT18], where a domain-specific ontology, called BDTOnto, was built that attempts to segregate known synonyms, homonyms, and other distinctive designations. Further, an initial concept of a big data technology classification was presented as the primary outcome. It delivered a thorough overview of many different technologies, their functionalities, and their relations to each other. More significant changes on this ontology and an extensive evaluation are presented in an extended form in the journal paper *Providing Clarity on Big Data Technologies: The BDTOnto Ontology* [VSJ+20]. Here, a real-life big data project was used for evaluation, which made use of the supplied ontology and its structure during the initial planning to implement the targeted big data technologies in this project. Details about it were published in 2019 in the conference article *Machine Learning Techniques for Annotations of Large Financial Text Datasets* [RVN+19].

The book chapter *Big Data* [VST20], published in the German Handbook of Digital Economy, forms an essential part of the theoretical foundation of big data, listed in section 2.2. Here, a thorough overview of the term of big data, its specifics, as well as related technologies and projects, is given. Additional information regarding the existing definitions and relevant data chracteristics, discussed in section 2.2, are extracted from the article *"Providing Clarity on Big Data – Discussing its Definition and the Most Relevant Data Characteristics"* [VST22], which is currently under review. The articles *Challenging Big Data Engineering Positioning of Current and Future Development* [VSP+19] and *Approaching the Big Data Science Engineering Process* [VSB+20a] complement the book chapter and mentioned article. While in the former, an attempt was made to uncover a definition, existing pitfalls and potentials of the engineering of a related system, the latter delivers specific steps required for the successful planning, development, testing, implementation, and operation. However, in both contributions, only a birds-eye perspective is given; thus, general extension and implementation ideas are shared, especially in the end. This includes among other things a DSS that shall allow potential decision-makers to identify relevant technologies for their specific undertaking. The concept itself was elaborated and presented in *Towards a Decision Support System for Big Data Projects* [VSB+20b].

Components that were considered and proposed in this conceptual contribution were then further elaborated and integrated. In the contribution *Towards an Automated Way for Modeling Big Data System Architectures* [VSP+20], the current state of big data architecture (BDA) modeling was determined by conducting an extensive literature review. Due to the absence of established methods, a modeling profile was developed and prototypically implemented with the help of the knowledge and principles gained from Unified Modeling Language (UML). The developed artifact allows the (semi-) automatic modeling of deployment diagrams, using configuration files at which big data technologies and their combinations are declared. Potential recommendations of those can be provided through the use of well-acknowledged methods. In *Applying Multi-Criteria Decision-Making Meth-*

ods for the Selection of Big Data Technologies [VST21] a multi-staged application of an Analytical Hierarchy Process (AHP) was proposed that utilizes the developed BDTOnto and different requirements for the identification of potential big data technologies.

Another application of the AHP was performed in *Decision-Support for Selecting Big Data Reference Architectures* [VBB+19b] that revealed the first promising insights of potential use when setting up BDA. Based on qualitative scenarios, different reference architectures in the domain of big data were examined and compared to each other. Stemmed from the given preferences by the user, a recommendation for a suitable reference architecture is provided. In *Facing Big Data System Architecture Deployments: Towards an Automated Approach Using Container Technologies for Rapid Prototyping* [VSI+22] a potential concept was proposed, implemented, and evaluated that allows the deployment of selected big data technologies, either in an isolated or combined way. Besides the application of known principles and prominent container technologies, the BDTOnto was used another time as the foundation to enable automatization.

The article *New E-Commerce User Interest Patterns* [VSJ+17] depicts a concrete big data project, at which click-stream analyses were performed in one of the largest B2B online shops to discover the user movements on their website. Emerging out of this, it becomes apparent that big data projects are unique in their execution, implementation, and application. Potential users of related technologies are frequently overwhelmed by the number of existing research contributions that deliver particular use case descriptions as they form a great part of today's applied information system research. Consequently, in the journal article *Identifying Similarities of Big Data Projects – A Use Case Driven Approach* [VST+20] these are analyzed. By conducting a thorough literature review, use case analysis, and adhering agglomerative clustering, a set of nine standard use cases (SUCs) is derived that allow project managers and other interested people to acquire yet unknown information for their potential project. An extended form was published in the article *Lowering Big Data Project Barriers – Identifying System Architecture Templates for Big Data Standard Use Cases* [VSS+22]. Here, further details of the use cases were analyzed regarding functional requirements (FRs), non-functional requirements (NFRs) and their severity. The obtained results play an essential role in integrating the referred SUCs in the overarching decision support procedure.

In the research article *Understanding Issues in Big Data Applications – A Multidimensional Endeavor* [SVJ+19] the general nature and existing problems of big data projects are investigated in more detail. Instead of listening a collection of existing articles focusing on this topic, a structured observation was performed from a socio-technical perspective. This comprises the creation of those systems, including the system's planning, engineering, and deployment and the eventual quality it may achieve. In doing so, the mutual influence and possible improvement measures were uncovered.

The journal article *Discussing Relations Between Dynamic Business Environments and Big Data Analytics* [SVD+20] extends this discussion, in the way of further short-

comings and problems which originate out of the profound dynamic nature of most of these big data endeavors. Apart from identifying and presenting existing problems and their solutions, specific measures to cope with dynamic business environments in the big data analytics domain and a microservice-based architecture are introduced. In summarizing that, the latter articles serve as supplementary material and auxiliary methods to increase the probability of a big data project success in highly dynamic environments, from the very beginning and during their run-time. However, these articles implicitly highlight the necessity of a computer-assisted approach and thus the intended solution of this work.

The aforementioned research articles are implemented either implicitly or explicitly within this work, dedicating separate sections or only textual fragments. In the course of this research project, sometimes comprehensive adjustments and extensions have been conducted that lead to differences in the original contribution. These are reflected by the ongoing changes not only of the project itself but also in the domain of big data. For instance, especially the developed BDTOnto was subject to major changes. Nevertheless, to highlight the contribution of each research artifact to the individual chapter, sections, and sub-section, the original articles are mentioned at the beginning of each chapter. In doing so, the related reference is given, for looking up required details that would extend the scope of this work. Additionally, it is intended to inform about the origin of certain text passages and selected sentences that were used from those. To provide a first glance about the main application of each of those, the following Table 1.1 provides further information, such as the publication year, the title, the publication type, the use within the work, and the specific reference.

No.	**Year**	**Type**	**Title**	**Main Implementation**	**Reference**
1.	2016	CA	How much is Big Data? A Classification Framework for IT-Projects	Description of the big data technology application check procedure, as described section 4.1	[VHB+16]
2.	2017	CA	Providing Clarity on Big Data Technologies: A Structured Literature Review	Results of the literature review are used within section 2.2.4 to highlight missing classification approaches	[VBT17]
3.	2017	CA	Ask the Right Questions: Requirements Engineering for the Execution of Big Data Projects	Primary implementation in 4.1 to sensitize for fulfilled preconditions. Additional use in section 2.2	[VJT17]

Table 1.1 continued from previous page

No.	Year	Type	Title	Main Implementation	Reference
4.	2017	CA	New E-Commerce Patterns	Mentioned in section 1.1 as a case for potential big data scenarios, to highlight and motivate the diversity of each project.	[VSJ+17]
5.	2018	CA	Classifying Big Data Technologies - An Ontology-based Approach	Implementation in the research background chapter, especially in context of the fundamentals in section 2.2 and section 2.4. The main artifact is introduced in section 4.3	[VPT18]
6.	2019	CA	Machine Learning Techniques for Annotations of Large Financial Text Datasets	Used in section 4.3 as the case for the evaluation of the proposed ontology.	[RVN+19]
7.	2019	CA	Decision-Support for Selecting Big Data Reference Architectures	Implementation in section 3.1, the investigation of the research background about existing classification approaches, as well as the chapter 5.	[VBB+19b]
8.	2019	CA	Challenging Big Data Engineering: Positioning of Current and Future Development	Implementation of motivational aspects and details in section 2.2.6	[VSP+19]
9.	2019	CA	Exploring the Specificities and Challenges of Testing Big Data Systems	Implementation throughout the work in parts. Mainly for motivation and background, cf. section 2.2	[SVN+19b]

Table 1.1 continued from previous page

No.	Year	Type	Title	Main Implementation	Reference
10.	2019	CA	Understanding Issues in Big Data Applications - A Multidimensional Endeavor	Implementation in section 2.2, highlighting the socio-technical-nature.	[SVJ+19]
11.	2020	CA	Towards a Decision Support System for Big Data Projects	Implementation in chapter 3, at which the requirements and overall concept are introduced.	[VSB+20b]
12.	2020	JA	Providing Clarity on Big Data Technologies: The BDTOnto Ontology	Similar implementation as No. 5, but in a higher level of detail	[VSJ+20]
13.	2020	CA	Approaching the (Big) Data Science Engineering Process	Builds a great part of the research background in section 2.2 as well as the motivation of this work in general (cf. section 1.1)	[VSB+20a]
14.	2020	BC	Handbuch Digitale Wirtschaft - Big Data	This German book chapter builds great parts of the foundation of the theoretical background section 2.2 - “big data“	[VST20]
15.	2020	CA	Towards an Automatized Way for Modeling Big Data System Architectures	Mainly implemented in section 4.5, at which the creation of deployment diagrams is discussed	[VSP+20]
16.	2020	JA	Discussing Relations Between Dynamic Business Environments and Big Data Analytics	Implementation in section 2.2, at which some potentials, challenges, and countermeasures are introduced	[SVD+20]

Table 1.1 continued from previous page

No.	Year	Type	Title	Main Implementation	Reference
17.	2020	JA	Identifying Similarities of Big Data Projects - A Use Case Driven Approach	Foundation for the identification of the SUCs in section 4.2	[VST+20]
18.	2021	CA	Applying Multi Criteria Decision-Making for the Selection of Big Data Technologies	Implementation in section 4.4, at which the multi-criteria decision-making for selecting big data technologies is introduced	[VST21]
19.	2022	CA	Facing Big Data System Architecture Deployments: Towards an Automated Approach Using Container Technologies for Rapid Prototyping	Used in section 4.6 to present the container-based rapid deployment solutions for big data system architectures	[VSI+22]
20.	2022	CA	Lowering Big Data Project Barriers - Identifying System Architecture Templates for Big Data Standard Use Cases	Implementation in section 4.2 as an extension of the previous work (see No. 17)	[VSS+22]
21.	2022	CA	Providing Clarity on Big Data - Discussing its Definition and the Most Relevant Data Characteristics (Under Review)	Used in section 2.2.1 and section 2.2.2 to discuss existing definition approaches and data characteristics	[VST22]

Table 1.1: List of related own contributions in ascending order, according to the year of publication

2 Research Background

The following chapter provides a thorough overview of the required background knowledge while attempting to deliver the latest state-of-the-art insights. Accordingly, different domains are described in detail as a foundation for the terms, methods, procedures, and paradigms used in this work. This includes not only the most focused domain of *big data* and relevant insights about characteristics, technologies, existing challenges, and the engineering of related systems (cf. section 2.2). Initially, essential details of systems engineering fundamentals are introduced within the first sub-section (cf. section 2.1). Apart from generic concepts and procedural models, related activities such as *requirements engineering* are described in detail (cf. section 2.1.2). Due to the targeted goal of an *end-to-end* procedure and a *DSS*, as a prototypical implementation, relevant background information about the latter are addressed in sub-chapter 2.3. One of the main layers of the intended DSS, the knowledge base, utilizes an ontology for storing and managing the required information. Due to the differences to classic databases, comprehensive information regarding their engineering and application are provided in the last sub-chapter (cf. section 2.4). The chapter itself is based upon the following research articles [VST20; VSP+19; VSJ+20; VPT18; VSB+20b; VSB+20a; SVD+20; SVN+19a; SVJ+19; VST22]. In these, extensive observations were often made with regard to the fundamentals presented here. Although the content of each paper can be found in every sub-chapter, it is primarily sub-chapter 2.2 that is largely composed of these contributions.

2.1 System Engineering

Designing and developing systems has remained one of the core disciplines in the computer science domain. Starting in early 1950s, the first standards and best practices for the creation of systems emerged [HWF+19]. Generally speaking, a system typically consists of different elements connected via relations with each other. These can be seen as essential building blocks or rather components that may, in turn, also serve as single *subsystems* [HWF+19, p. 4]. More particular components can be defined as *"a reusable, self-contained and marketable software building block that provides services via a well-defined interface and can be used in not necessarily predictable combinations with other components at the time of development"* [Tur03, p. 19]. Typically, those fulfill various characteristics, such as the reusability, service definition via known interfaces, encapsulation of the implemen-

tation, loss decoupling, distribution, and installation [Tur03]. As a consequence, such system elements are not necessarily always part of the same system. Instead, also loosely coupled architectures are imaginable, as they are the de facto standard for today's SE and are highly requested in a big data environment [SPB16].

Modularity in the context of those is often requested but can also be very complicated. The next step towards this evolution are microservices, which depict single operable applications that can interact via defined interfaces with each other. They decompose larger functionalities of an application into smaller *services*. Hence, those can be rather seen as system components that may interact with each other without having interdependencies to other elements [NMM+16]. Independent of the nature of the elements, whenever there are numerous elements and connections between them, those systems can be described as a *complex system* [HWF+19, 4–11].

The complexity and way of composition are just a few characteristics. One of the most important aspects is the life cycle each system goes through, covering inter alia, the planning, design creation, development, operation. Nicholas et al. define four consecutive phases, named conception, definition, execution, and operation [NS21a]. Others, such as Mobus et al. [MK15] define them in more detail. In particular, they reach from the initial *identification* of the need for the system, over the *system analysis* to the *design*, *construction*, and *operation* until the *decommissioning*.

In their core, all of them a very similar to each other. This applies to the different stages and phases as well as the idea to use these as a foundation for the stepwise engineering of related system. Nicholas et al. [NS21a] refers in the context of this systems engineering as *"a way to bring a whole system into being and to account for its whole life cycle"*. This is comparable to other definitions, such as from the non-profit organization International Council on Systems Engineering (INCOSE). According to them, the term can be described as a *"transdisciplinary and integrative approach to enable the successful realization, use, and retirement of engineered systems, using systems principles and concepts, and scientific, technological, and management methods"* [INC20]. Despite those explanations and the needed integration of different concepts, technologies, components, and (sub-) systems, the SE can be seen as a meta-engineering discipline. Hence, further methods, processes, tools, and information are required, all along the different stages of the life cycle [MK15].

An overall framework that contains those important aspects of SE is depicted in Figure 2.1. It identifies two main areas essential for the engineering of a system. Each of them includes further modules. In particular, those are the *SE principles* as well as *problem-solving process*. The first area represents all elementary modules that are indispensable for the successful creation of a system. Apart from using specific procedure models, as they are already mentioned before and in the following section 2.1.1 concretized, the general systems thinking also belongs to it. Essential terms, definitions, and relationships, as described in this section, are always necessary [NS21a; INC15].

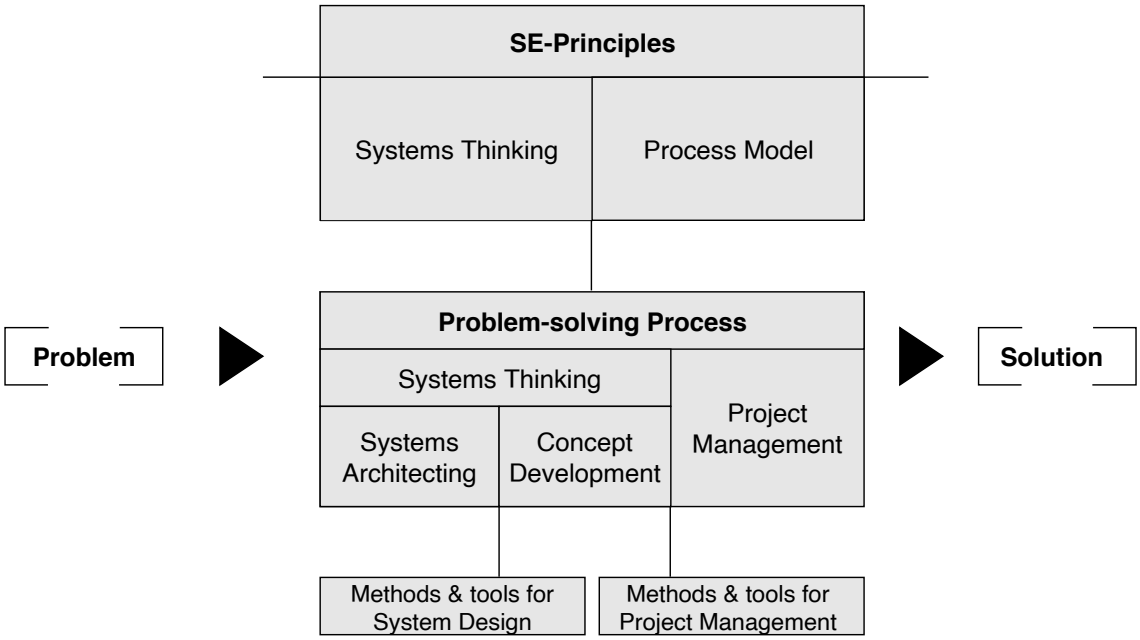


Figure 2.1: Systems engineering framework according to [HWF+19]

In contrast to this, the second area addresses modules that are significantly responsible for achieving the goal. Their employment and application depend on the problem definition and the desired solution. The *system design*, as one main module, is divided into *systems architecting* and *concept development*. As the name already implies, basic system architectures are to be created within the former. In turn, the *concept development* refers to the specification of the architectural design in more detail. Here, based on further information, for instance, emerging out of situation analysis, are reflected and used for the further specification of the system architecture [HWF+19].

According to the ISO:42010, a system architecture describes *"fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution"* [ISO]. Hence, these represent a decisive link between the functionalities to be achieved by a system, the requirements to accomplish these, as well as the used tools and technologies that will help to fulfill them [Som16; DL20b]. The engineering of the systems goes hand in hand with the planning and realization of a project [HWF+19, p. 112]. Although these are sometimes explicitly referred to IT projects, resulting from the strong focus on technologies, the terms *project* is synonymously used here, as in most related literature in the computer science domain. According to the Project Management Institute a project can be described as *"a temporary endeavor to create a unique product, service, or result"* [PMI08, p. 442]. Therefore, it sets the boundaries for the SE process in terms of the existing problem, the available resource as well as the expected solution. Thus, efficient project management represents a core discipline for systems creation [NS21a; HWF+19; Som16].

According to [HWF+19] it should be noted that cutting-edge methods and tools do

not characterize the SE. Thus, it can be rather seen as a disciple that greatly profits from knowledge from various other domains and experiences. For that reason, in the following, not all existing SE details are shared. Instead, the remaining modules highlighted in Figure 2.1 are discussed, which are of major importance for this work, especially in the context of BDE. For further insights, the following material can be consulted [ISO; HWF+19; NS21a; Som16; INC15].

2.1.1 Process Models

For the realization of a system architecture, basic principles and procedures are required, as thoroughly described before. Numerous approaches emerged within the last decades that could be used as guidelines. However, they all differ in terms of their complexity and agility. While some deliver a very comprehensive structure consisting of multiple steps, others are instead built to react to new results and insights. Famous examples for the latter are those which are also known from the IT-project management or agile software engineering domain, such as Scrum or Extreme Programming [HWF+19; Som16]. For the intended solution of this thesis, basic principles are required that can also be harnessed by non-experts, in terms of big data, SE, and their intersection. Although critics exist that those approaches are sometimes too complex and cumbersome, they deliver good insights of the SE without requiring years of experience. According to [NS21a, pp. 119–132] six stages along the project life cycle are necessary for the SE. These range from the *identification* of the conceptual design over the detailed *design* and *construction* to the *operation* and *support* of the system. Similar setups can be found in well-known approaches that can be aligned to the referred structured process models. These are, for instance, the *waterfall* or *V* approach [HWF+19; Som16].

The waterfall model is one of the best-known procedures for system development. It consists of five consecutive steps: *definition of requirements, system and software design, implementation and component testing, integration and system tests, operation, and maintenance*. Each of those is followed by the other. If a problem occurs during the operation and maintenance, previous steps can be revisited, forcing the potential user to finalize everything, related to the project planning, at the beginning. Mainly if problems regarding the requirements occur in later stages, a modification will result in drastic cascading changes [HWF+19; Som16]. However, if everything is well planned, the waterfall model denotes an easy-to-use procedure that moves from the general to the detail, typically called top-down approach. Vice versa, bottom-up approaches, such as from INCOSE, attempt to move *"from the detail to an (improved) whole"* [HWF+19, p. 110]. However, in many cases, a mixture of both is followed. This is primarily aimed at improving, re-creating or replacing existing solutions. Prototypes previously named as a possible result of the solution construction are often used in this context [HWF+19].

The V-model represents such a mixture and combines both approaches. The epony-

mous shape can be seen and treated as an answer to the waterfall model that *only drives down but not up again*. Using a top-down approach, different requirements are developed and sub-systems and components are identified. This is down from general to specific, whereas the beginning is described by the overall system design and the end by the definition of the single components. Through the use of the FRs, further specification for each *module* of the later architecture is identified. Afterward, these are stepwise implemented as well as their integration verified and validated, using the bottom-up approach [NS21a; HWF+19]. The feedback, which may result in single testing of modules or the architecture as a whole, can be quickly brought back for further analysis and improvements. Hence, a user is not forced to restart everything from the beginning. Instead, it is recommended to *"don't rush to solutions! Look for alternatives"* [NS21a, p. 53]. By comparing these, and other existing approaches (cf. [HWF+19] – sub-chapter 2.2), it becomes apparent that many similarities between them can be ascertained. This is not limited to the idea of the system's life cycle. Instead, the different steps ranging from problem identification to system deployment are considered here. An *adaptable and evolvable* version that implicitly comprises the essential core steps of each approach is proposed by Mobus and Kalton in their work [MK15].

According to their approach, depicted in Figure 2.2, the SE process starts with problem identification. Within this step, the same will be performed. It is crucial here that the actual problem will be discovered and not only (obvious) implications, providing a problem-centric instead of a solution-centric view. Afterward, the problem needs to be specified in more detail by developing requirements within the *problem analysis* stage. Inter alia, this can be realized through decomposition and separate observation of relevant sub-problems. Besides that, the boundaries of the system functionalities can be determined. The identified problems are subsequently used as an input for the *solution analysis* that pursues to present a possible system specification. In doing so, smaller, logically independently acting parts of the system (sub-system) and their interconnections are identified. Those specifications should conform to the needs ordinated from the problem analysis.

As a transition to the next step, these information can be brought together in a potential modeling approach. Independent whether these are graphical or mathematical representations, they may deliver beneficial insights and a big picture of the intended outcome. For visual presentations, potential solutions can be found, for instance, in the UML, where different types of modeling diagrams are described. Most of them are applicable in later SE stages and during the initial steps, such as requirements engineering [Dic17]. After all relevant specifications were made, the *solution design* takes place, in which the physical aspects are determined. By the end, design documents are formulated, which serve as an input for the *solution construction*. The actual systems, often referred to as the artifact, are developed within this step. Eventually, the developed artifacts need to be evaluated, which will be performed in the *solution testing* phase. Although a comprehen-

sive verification will be performed at this point, concurrent validations during each of the previous stages are recommended that are covered under the *discrepancy resolution feedback*. This, in turn, may lower the need to perform changes in later stages. If everything was successfully developed, the solution is delivered and productively used. Continuing steps cover the monitoring, performance monitoring, and further analysis, which may lead to modifications, upgrades, or decommissioning [MK15].

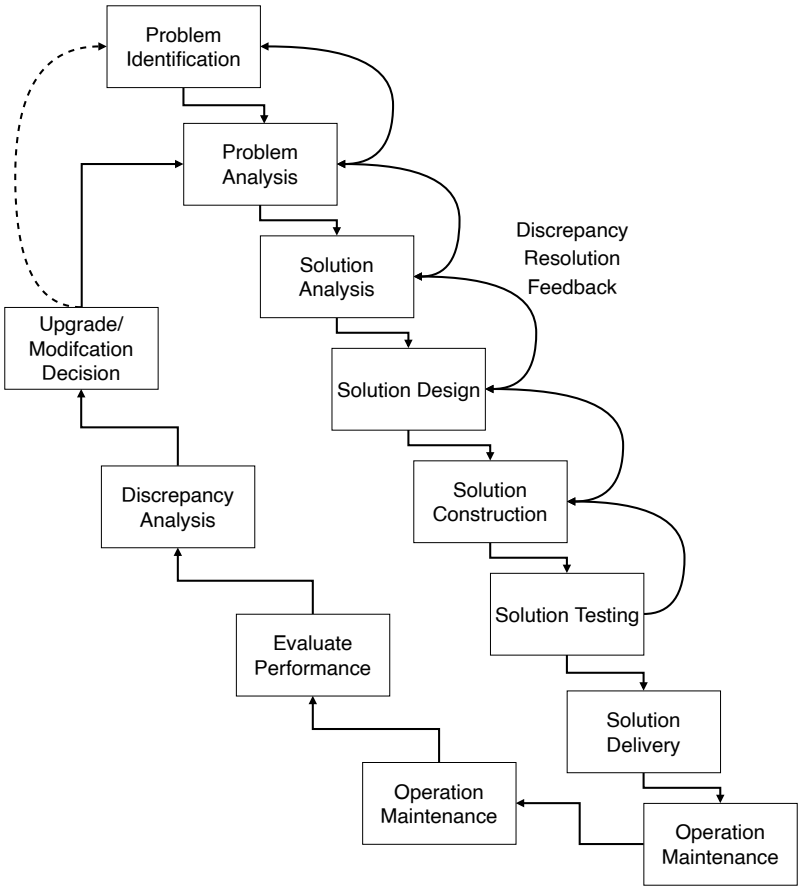


Figure 2.2: The systems engineering life cycle based on [MK15]

Often, multiple solutions may exist for the project instantiation that appear to be desirable. However, only one of them is realizable due to the given restrictions and constraints. Hence, in the beginning, a detailed analysis is conducted. In doing so, within the feasibility study, qualitative and quantitative information are compared using specific metrics and calculations like the use of a weighted sum. After the selection of a suitable concept, it is further specified and analyzed [NS21a]. Due to this, similar steps are applicable in the case of multiple solutions, which may also occur during the different SE process steps. Because of its complex nature, almost entirely multiple iterations are required to achieve constant validation, and improvement [MK15; NS21a]. Hence, prototypes are used, which can be described as *"an early running model of a system or component built for purposes of demonstrating performance, functionality, or proving feasibility"* [NS21a].

As part of the *solution construction*, these provide detailed insights when there is still uncertainty regarding the usability of the solution design in practice. However, it should be noted that one of the huge disadvantages is not only the time spent but also the monetary expenditures that might be required [ISO11]. Notwithstanding that, throughout the entire process and most importantly during the beginning of the procedure, the identification of requirements that define the solution space is needed. As a major part, their sufficient engineering is crucial for the successful instantiation of the procedure and the systems engineering in total [Dic17]. This was not only frequently addressed, implicitly, via the required project management, but also explicitly, as it is in the case of the waterfall model [HWF+19, p. 57]. In the following, the focus is put on the requirements engineering as it is indispensable for the final artifact of this work.

2.1.2 Requirements Engineering

Efficient project management is not only an essential part of the SE process. Furthermore, it plays a vital role in today's IT projects when it comes to their successful realization [She07]. Project management can be described as *"the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements"* [PMI08, p. 6]. Consequently, this discipline comprehensively covers everything needed to achieve the *goal* of a project. One of the most crucial steps at the beginning of each project management process is the requirements engineering, which is intended to define the system functionalities, boundaries, and habits, thus, the project requirements [Som16; PMI08; ISO11]. Apart from the general identification or collection of the requirements, various other activities are connected to this function. Hence, it represents an *"interdisciplinary function that mediates between the domains of the acquirer and supplier to establish and maintain the requirements to be met by the system, software or service of interest"* [ISO11, p. 6].

As one may note, various requirements are developed and used as the foundation for a project description as well as the instantiation of a systems engineering approach [PMI08; NS21a; HWF+19; MK15; ISO11]. A requirement, in the context of the systems, is a well-formed statement that can be verified, met, or possessed by a system, qualified by measurable conditions, and focused on the system either via a direct fulfillment or via interaction with a user [ISO11]. These requirements often differ in their overall comprehensibility and applicability in the current body of knowledge. However, they always need to be documented and contribute to achieving the overarching goal. Originating from the user perspective, *high-level requirements* are used as a project's starting point and, thus, the SE process. They describe an initial list of basic system functionalities that are required for the targeted solution [NS21a]. An example could be: *The system is capable to provide data.* The origin of the requirements are primary *stakeholders*, which comprises all relevant persons involved in the project. Besides the user that later interacts with the system, developers, managers, lawyers, and many more are covered by this term [PR21].

Further, more detailed, requirements are developed from the user perspective, which can be then distinguished. The most prominent are FRs and NFRs [Som16; ISO11; NS21a; PR21]. FRs specify the functions that the new system needs to provide [NS21a, 121–122]. Additional specifications of those can be achieved by so-called *performance requirements*, which are associated with each FR individually. Those deliver further (technical) insights and critical conditions that are measurable [NS21a; ISO11]. Based on the ongoing requirements, an example could be: *The request should be processed in 98% of the cases in less than 2 seconds.*

NFRs deliver information *"under which the system is required to operate or exist"* [ISO11, p. 12]. Compared to the FRs these are not bound to the functionalities of the system, instead they deliver further insights of the system habits and overall properties. Relevant ones are, for instance, compatibility, commonality, cost-effectiveness, reliability, maintainability, testability, availability, robustness, usability, and expandability [NS21a, p. 113]. Although an isolated recognition of each of those is possible, a change or special treatment might also influence other NFRs. These are not completely inseparable from each other [Som16]. As this type has a great influence on the solution, especially those which are often ending with *-ility* can be further classified as quality requirements [ISO11]. An example for an NFR, the availability, in particular, could be: *The system must be available 98% of the time during a year.* Further categorizations of these quality requirements can be found in [ISO17].

Apart from those overarching requirements categories, in the *ISO 29148-Systems and software engineering - Life cycle processes – Requirements* [ISO11], additional types are introduced. Namely, these are *usability-*, *interface-*, *process-*, *human factor requirements*, and *design constraints*. For instance, the latter denotes limitations due to given boundaries, such as coupe with legacy system elements. In addition to the requirement categories, a structured sequence is described in [ISO11], which forms the de facto standard today. While in many models, such as the waterfall or V model, requirements engineering is only described as a single step of many and is not specified accordingly. This is not the case for the ISO29148 [PR21]. A three-stepped procedure is described within the ISO norm, contributing to the overall system creation. All steps can be iteratively and recursively performed until each element is sufficiently observed. The procedure, based on [ISO11] is depicted in Figure 2.3.

The requirements identification is performed within the stakeholder requirements definition process (1). Initially, the various stakeholders throughout the system life cycle need to be identified. Their involvement and the different perspectives need to be incorporated to obtain a comprehensive overview of the habits of the system and its functionalities. Among other things, this can be done by interviews, questionnaires, and workshops. Additionally, it is required to identify constraints that may impact the overall solution analysis, such as deadlines, budgeting, technical decisions, or existing agreements.

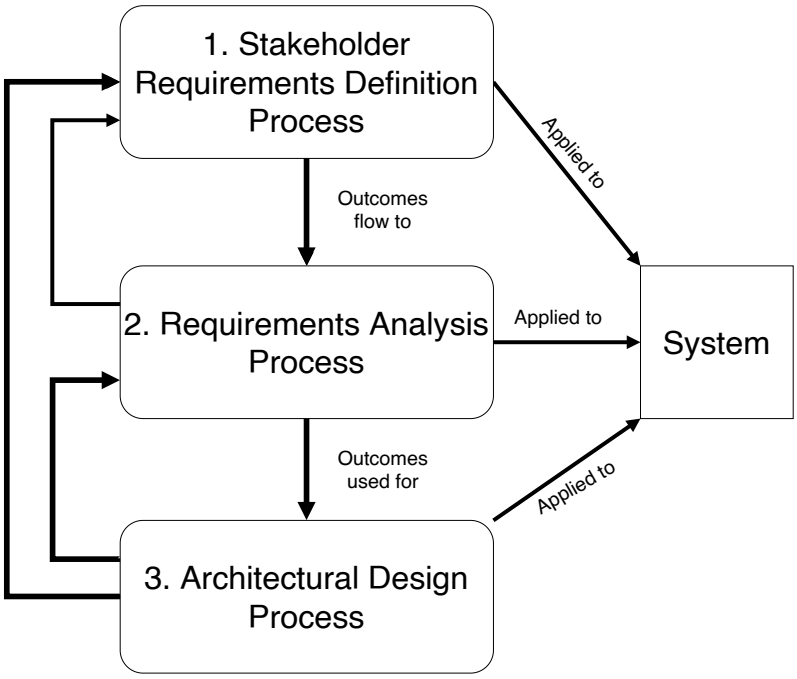


Figure 2.3: Requirements engineering process according to the ISO 29148:2011 [ISO11]

Despite the fact that many materials exist that deliver guidelines for the requirements collection, documentation, and refinement in early stages [Som16; NS21a; ISO11], some of them might still be hidden due to unforeseen interactions and interconnections between single activities. This also includes stakeholders, which can be unknown until a certain time. As another way for the identification, documentation, and validation of requirements, different modeling or scenario descriptions can be utilized [PR21; ISO11]. To support the scenario-based approach, for instance, use case diagrams are applicable that illustrate basic functionalities and relevant users. Because these diagrams don't specify any further information, additional formulation guidelines or specific templates can be used [PR21]. In the end, most of the requirements are stated in the language of the stakeholders. The respective first documentation is thus often called stakeholder requirements document [NS21a].

After formulating all of these, feedback should be given and problems resolved as an essential activity. This is performed to allow further specifications in the different requirements types and to resolve open issues, such as conflicting requirements or blurry structures. If the potential stakeholders are insecure about the relevance of single requirements or their particular specifities, further *priorities and margins* could be provided during the requirements definition. In particular, according to [NS21a], requirements can be ranked based on their priority, expressing their relative importance. Especially in the case of conflicting resources or functionalities, a prioritization might be sensible. Furthermore, when particular values are to be expected, additional margins can be used and added to the essential requirement.

An example could be: *The data is stored within 5 milliseconds, with a margin of 2 milliseconds.* By conducting requirements and SE procedures, various interest groups and stakeholders are involved, which leads to different knowledge levels between them.

Consequently, during this procedure, the requirements need to be sharpened and detailed as much as possible. Although this is intended by following the three steps iteratively or recursively, further auxiliary steps can be performed here. In doing so, to achieve better comprehensibility and transparency, often those FRs and NFRs are aligned to different categories and logical groups, such as *system data processing functionalities* (category) with *data ingestion* (group) and *data analysis* (group). This is especially the case to achieve a common understanding, which is heavily required if many people are involved in the process, and hundreds or thousands of requirements exist. As a suitable representation, so-called requirements breakdown structures are used [NS21a, 113–114]. Although those structures deliver sensible insights of the intended system functions and habits, not all detailed information are covered at the beginning. When the initial idea of the project and the potential solution exist, only vague requirements are defined. As a first step, this is a completely normal phenomenon, as these are also subject to an iterative development process. Notwithstanding that, a certain degree of precision is required at one point in time.

The adhering requirements analysis process step (2) transfers the defined requirements into system specifications emerging from the stakeholders. Essentially these describe the *end-item* in detail, including subsystems, components, and other information [NS21a, p. 114]. At this stage, creating the specific architecture has not started yet. Instead, the system specifications are defined, and their applicability is checked. In parts, this also includes some of the architectural realizations. Eventually, different criteria are specified with which the solution can be verified, and the architecture construction in the particular environment starts. One of the outcomes of this stage also involves modeling or prototyping the potential solution to deliver feedback to the relevant stakeholders. This shall ensure that the system requirements were adequately reflected [ISO11].

The process of rapid prototyping focuses on the rudimentary provisioning of an incomplete excerpt of the solution that can be quickly realized and used for initial experiments. Typically, these are more intended to provide an initial idea about the potential capabilities as well as the overall feeling of working with the system [NS21a, p. 117]. Therefore, certain types of requirements are not always fully applicable here. NFR, for instance, have a significant impact on the overall quality of the solution. Hence, a full recognition, even in the early stages of the prototyping, is mostly hard to realize and can be neglected to some degree [Som16]. Notwithstanding that, similar to the prototyping, the modeling and simulation of system parts could also be used here to facilitate sensitive analysis if the user expected something different [ISO11].

The transition to the third step of the requirements engineering procedure is fluent. The previous step's outcomes are used for the architecture creation, further analysis,

demonstration, and testing in the architectural design process. Here, the focus is instead put on essential interfaces and connections of the planned system elements and the verification and validation of the requirements. Hence, detailed planning for the demonstration and testing is required to ensure that the targeted solution fits the existing requirements and is sufficiently created. In case further revisions need to be performed, single activities within each step can be reperformed. This applies to each of the system levels, including also its subsystems or components [ISO11]. For further best practices and details related to the design and development of requirements, the following materials can be recommended [ISO11; Som16; NS21a]

2.2 Big Data

For many years now, the domain of big data has received lots of attention, as numerous studies, reports, and research articles have revealed. Up to this date, a multitude of different definitions, technologies, architectures, and best practices appeared that were supposed to provide clarity in the *jungle* of existing solutions. Instead, it led to further confusion regarding the general nature and the applicability of big data. The following sub-sections provides a thorough description of the term big data, its particularities, and existing challenges to elucidate those obstacles in detail. First, a short historical outline is presented, and existing definitions are discussed. Since the underlying data characteristics are not only the foundation for most of these approaches but also essential to the overall understanding of the domain itself, these are discussed afterward. Additionally, for the further understanding of essential terms, concepts, and ideas, used in the contribution at hand, an introduction to big data projects, technologies, potential architectural setups, as well as a comprehensive engineering approach, are given.

2.2.1 Definition

In the yearly published hype cycle of the business analytics company Gartner, the maturity and adoption of recent technologies are graphically represented according to the five key-phases that each of those experience during their life cycle. Namely, those are the *innovation trigger*, *the peak of inflated expectations*, the *trough of disillusionment*, the *slope of enlightenment*, as well as the *plateau of productivity* [Gar22b]. Throughout those stages, each depicts whether new technology can be seen as hype or an ideal market solution that creates revenue. An example of the hype cycle in 2011 is provided in Figure 2.4. In 2011 the term big data appeared for the first time in that life cycle [Gar11]. After becoming a hype topic quickly, in the following years, in 2014 it passed the peak of inflated expectations [Gar22c] before it vanished completely in 2015 [Gar15]. One of the main reasons lies in the tremendous effort put into researching and applying this topic in academia and industry, especially in the first years of its incorporation into the hype cycle.

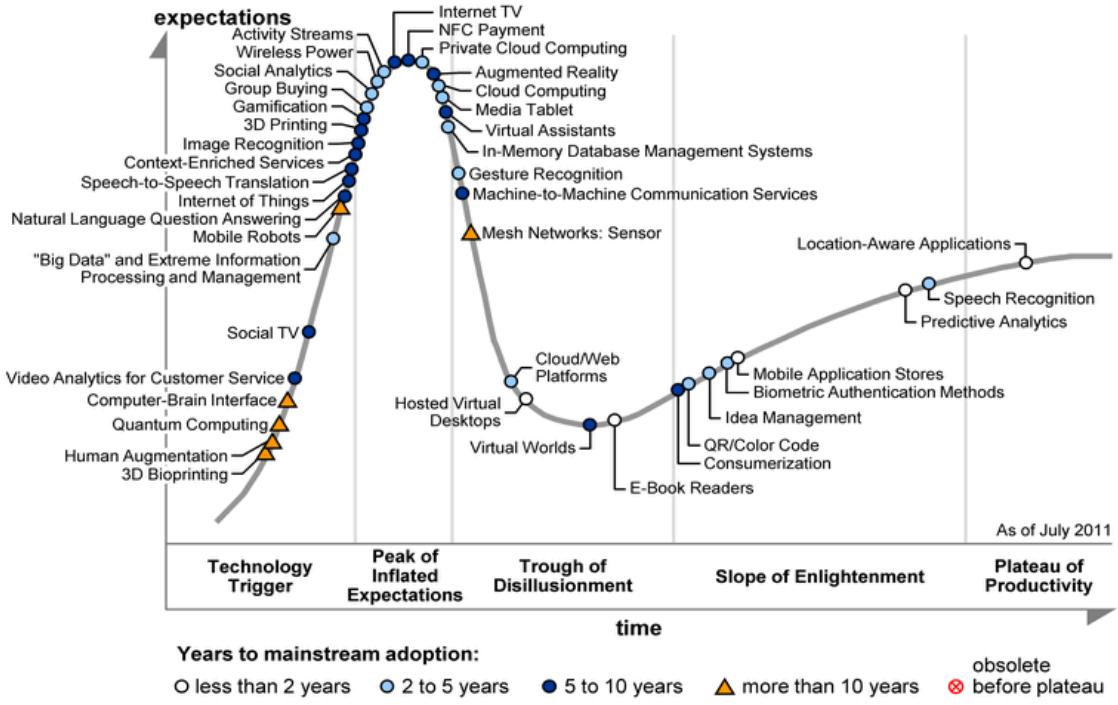


Figure 2.4: The Gartner hype cycle 2011 [Gar11]

According to Betsy Burton, a business analyst of Gartner, the term big data *"has become prevalent in our lives across many hype cycles"* [Woo15]. This statement indicates the importance and interdisciplinary nature that big data quickly achieved. Compared to other technologies, it has become rapidly indispensable for today's organizations.

Since the initial mention in a research article in [CE97], the definition and application of the term *big data* tremendously changed. According to Cox and Ellsworth [CE97], big data can be distinguished in *collections* and *objects*. While the former refers to the data sets acquired and aggregated, the latter solely focus on single data elements. In both cases, these can sometimes be *too big* under the given context. Although some minor evidence was put on other relevant *data characteristics*, these are only implicitly mentioned, such as highlighting the relevance of meta-data that sometimes comes with an extensive collection or the origin out of heterogeneous databases. This trend slowly continued until the subsequent mention by Roger Mouglas from O´Reily [MKP+21], who emphasized that big data *"refers to a large set of data that is almost impossible to manage and process using traditional business intelligence tools"* [Don13].

Concurrently to that, the first release of Apache Hadoop, which is today known as one of the core technologies in the domain of big data, emerged (cf. section 2.2.4). Other approaches that arose afterward didn't exclusively consider the *volume* of the data [CML14]. Starting from that, other data characteristics were recognized [SS18; AA19], and sometimes additional specificities are highlighted, such as the need for scalable technologies [CG19a] or the challenges with established technologies [MCB+11; TMK12].For that rea-

son, it is not surprising that many different research articles almost exclusively deal with the exploration of an applicable definition, such as [WB13; YP16]. Some of those definition approaches, over the course of the years, are depicted in Table 2.1.

Reference	**Year**	**Definition**
[CE97]	1997	*"Big data objects are just that – single data objects (or sets) that are too large to be processed by standard algorithms and software on the hardware one has available."*
[MCB+11]	2011	*"Big data refers to datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze."*
[BIT12, p. 7]	2012	*"Big Data refers to the analysis of large amounts of data from diverse sources at high speed with the aim of generating economic benefits."*
[CCS12]	2012	*"Big data summarizes technological developments in the area of data storage and data processing that provide the possibility to handle exponential increases in data volume presented in any type of format in steadily decreasing periods of time."*
[TMK12]	2012	*"Big data' refers to data sets whose size is beyond the capabilities of the current database technology."*
[MC13, p. 6]	2013	*"Big data refers to things one can do at a large scale that cannot be done at a smaller one, to extract new insights or create new forms of value, in ways that change markets, organizations, the relationship between citizens and governments, and more"*
[HYA+15]	2014	*"Big data is a set of techniques and technologies that require new forms of integration to uncover large hidden values from large datasets that are diverse, complex, and of a massive scale"*
[CG19a]	2015	*„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."*
[SS18]	2018	*"Big data is defined as a large collection of multifaceted data sets, which can also be described as being high volume, variety and velocity, making difficult to move and process instantly with the traditional database management systems."*
[Gar22a]	2022	*"Big data is high-volume, high-velocity and/or high-variety information assets that demand cost-effective, innovative forms of information processing that enable enhanced insight, decision making, and process automation."*

Table 2.1: Selected definitions of the term big data

Those definitions showcase the maturation of the term. However, at the same time, it becomes evident that many discrepancies still exist, and the domain is undergoing

continuous transformations. Although the data characteristics, or the nature of the data regarding different dimensions, are still the central focus today, it is above all the technologies, techniques, and general paradigms that make big data appear remarkable. Therefore, it serves today as a technical foundation for many different data-intensive application scenarios requiring sophisticated technological concepts, as it was already indicated in the introduction (cf. chapter 1). The specificities, which are often differently discussed, are presented in the following sub-sections of this sub-chapter.

2.2.2 Data Characteristics

The nature of the data plays a decisive role in this big data domain and is often referred to by the term *data characteristics*. As already highlighted by a multitude of different definitions (cf. Table 2.1) and contributions, as thoroughly investigated in [VHB+16], the quantity (*volume*), structure (*variety*), and speed (*velocity*) of the data are the most widely acknowledged ones. Notably, all of those start with the letter *V*, an implicitly followed property of the data characteristics formulation in the big data domain. Presumably, this can be traced back to their origin. The main characteristics, which are commonly abbreviated as the *3V's* go back to the former Gartner (formerly META) data analyst Doug Laney and his report from 2001, titled *3D Data Management: Controlling Data Volume, Velocity, and Variety* [Lan01a]. In this report, he discussed the potentials and challenges associated with considering these three dimensions. In the following years, these characteristics were widely applied and further developed in the context of a continuous data increase. To this day, the 3V's are probably the most critical differentiators when it comes to the consideration of a data-intensive endeavor. However, even though a broad acceptance was achieved by many researchers and practitioners, sometimes different descriptions of each of them can be found. Generally speaking, as highlighted by the given definitions before, this circumstance is notable for almost all specificities of the domain of big data.

As a result of the ongoing global digitalization, for the year 2025 a volume of 175 Zettabytes (ZB) of data is expected to be generated worldwide, compared to 33 ZB in 2018 [RGR18]. *Volume*, as the most prevailing data characteristic, stands eponymous for the amount of data that must be acquired, stored, and processed. Within the literature, as most of the other characteristics too, it is considered heterogeneously and refers to the number of data elements and their sheer size [DGL+13; CG19a; AA19]. At what point in time one can speak of large volumes of data has not yet been clearly defined [ACB+15a]. Individual studies repeatedly deal with different definitions and metrics for the analysis and assessment of the volume, as found out in previous research [VHB+16]. While many of them attempt to provide generic descriptions, such as *too huge to handle with established databases*, others indicate data in terabytes, like [PSK17]. However, those often don't follow any processes that provide enough evidence or a clear argumentation.

Variety, as another core characteristic, refers to the diversity of the data in terms of structure. Usually it can be distinguished into structured, semi-structured and unstructured data [Erl16; GSS+16, p. 31]. The first type describes data sets with a fixed scheme and can be stored, managed, and analyzed without great effort. A common example is the relational data model, in which information is stored row-based in a tabular format [Erl16, p. 34]. The semi-structured data partially contain information about the underlying structure. Still, they are further modifi- and extendable, for example, when using the exchange format of the XML [Erl16, p. 35].

Contrary to the types above, unstructured data can mostly be handled by special procedures only [GH15]. This circumstance is mainly due to the fact that, despite an alleged similarity, great differences in the ability to process the data may occur, for example, when different file formats are present. Image files are a good example of this. While common formats, such as Joint Photographic Experts Group (JPEG), Portable Network Graphics (PNG), or Bitmap (BMP), can be used to display the images, vector graphics, or program-specific formats with additional meta-information are also capable. Many authors understand diversity not only in terms of pure structure. For instance, in [CG19a], the incorporation from various sources is meant. Further factors such as the content of the data, the underlying context, the used language, units, or formatting rules are also addressed. In the case of the combination of differently structured data, e.g., through merging internal and external holdings, poly-structured data is also sometimes referred to [BIT14].

Velocity, as the last of the three big data core characteristics, refers to the speed of the data. As in the case of the previous characteristics, there is no universally accepted definition. While many experts in this field address the speed of the pure data processing [KNH+19; PSK17], others also refer to the speed with which the data arrives [GH15]; [Erl16, p. 30]. Notwithstanding that, velocity is usually expressed in batch, (near) real-time processing as well as streaming. While batch processing is a sequential and complete processing of a certain amount of data [BIT14]; [Gha21b, p. 13], (near) real-time processing is described as an almost continuous processing method. Especially when the speed requirement increases, the processing becomes a non-trivial task. Hence, various technologies, frameworks, and architectures have been developed that will be discussed in the upcoming sub-sections.

A graphical overview containing the referred core data characteristics, building the foundation of big data and supplementary ones, are depicted in Figure 2.5. Noteworthy, apart from these characteristics, many more have emerged since the origin of the term big data and the first mentioning of the data characteristics. Some of these have become widely accepted and used by researchers and practitioners today.

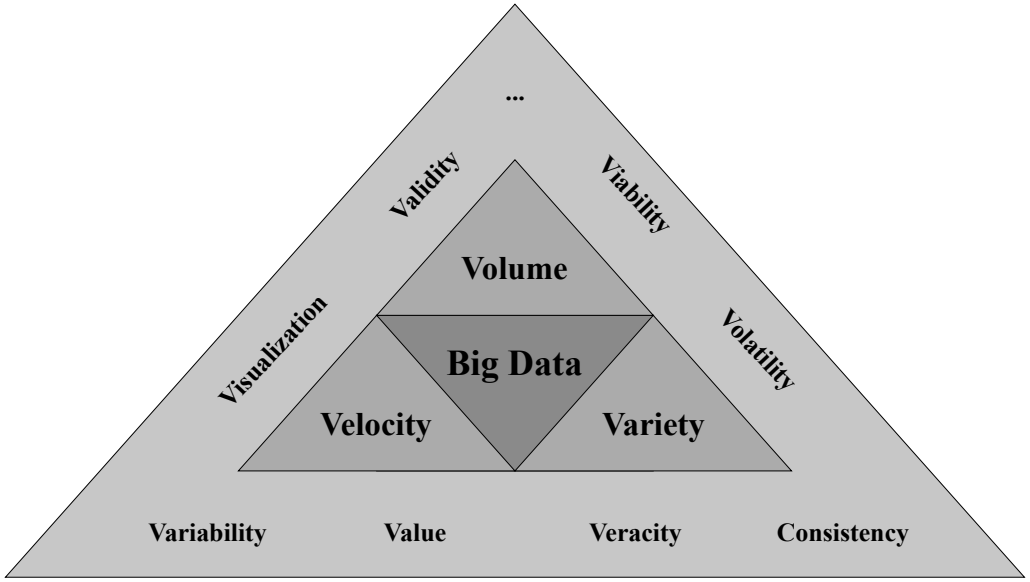


Figure 2.5: Overview of existing characteristics, based on [VST20]

Volatility, as another characteristic, *"refers to the tendency for data structures to change over time"* [CG19a]. While this statement may lead to the general idea of a data structure change, as described by the variety, it rather focuses on the overall changes of the data over time and its usability. The analysis of the data that was collected, stored, and processed initially may experience a certain *shift* regarding its level of detail or use [CG19a; KAK16]. Hence, a holistic consideration of novel and historical data is required.

Variability, as an additional characteristic, refers to fluctuations that may occur in the context of the processed data. While some authors, such as [Hus20], focus on the changes of the data itself, others understand by it the variations of the three core characteristics, either in an isolated or compound way [CG19a; KWG13; GH15]. Because of that, an impact on the overall data processing may occur that needs to be counteracted. This includes the data flow rate, structure, and volume. In doing so, sophisticated approaches are required to overcome emerging problems and increase the system's flexibility. One of them is the scaling of related systems, horizontally and vertically, for instance, by harnessing the capabilities of cloud computing [CG19a].

Veracity provides information about the data quality and thus the reliability of the raw data. It was first coined by IBM report in 2013 [IBM13]. Especially when it comes to the necessity to utilize unreliable data for big data projects, both the correct context and the used analysis method are of utmost importance [GH15]. In the case of unreliable data, one measure could be to merge several sources in order to increase the overall quality, and thus the trustworthiness [IBM13; IAG+15].

With the *value* of data, Oracle [Dij13] introduced a meta-data characteristic, rather than a stand-alone data characteristic, as it is heavily influenced by other characteristics, such as veracity, velocity, volume, and variety [Erl16, p. 32]. It focuses on the economic

value of the data to be processed. It is possible to extract information and gain previously hidden knowledge, especially with semi- and unstructured data, which differ from traditional data structures. Often, however, the data in its pure form do not contain any significant benefit, which means that preceding processing steps and analyses are necessary to generate such value [GH15].

Apart from those existing V's, other concepts were introduced in recent years, providing further data characteristics that do not follow this pattern. The 3C model addresses the cost, complexity, and consistency in the context of the data to be processed. While the first focuses on the overall monetary expenditures, required for a technological realization, the second indicates the severity of connections and relations between the data. The last C, the consistency, refers to the *"data that flows among various sources and is shared by multiple users"* [BJG14]. Generally speaking, keeping a consistent state can be demanding due to the distributed nature of data and related systems in big data. In many cases, simultaneous writes and reads must be carefully observed to avoid adhering problems [CG19a; BJG14].

While the characteristics mentioned above have been widely recognized today, numerous other papers, studies, and reports have attempted to establish additional data characteristics that are less widely used. As a result, reports emerged that present 10V's [KAK16], 17V's [PSK17], 42 V's [KDn22], 51V's [KNH+19] or even 56V's [Hus20]. However, as many authors already highlight by themselves, not all of them are always important or even applicable. An overview of the essential definitions for this thesis is depicted in Table 2.2.

Characteristic	**Definition**
Volume	Volume indicates the amount of data that has to be handled.
Variety	Variety refers to the heterogeneity of data and its sources.
Velocity	Velocity denominates the speed at which data are incoming, and the speed at which received data must be processed.
Volatility	Volatility refers to the tendency for data structures to change over time.
Variability	Variability corresponds to the change of the other characteristics.
Veracity	Veracity reflects the reliability and trustworthiness of the data.
Value	Value refers to the economic value that emerges out of the processing of the data.
Consistency	Consistency refers to the data that flows among various sources and is shared by multiple users. In doing so, the data needs to be in a consistent status all time.

Table 2.2: Data characteristic definitions based on [VSP+19]

2.2.3 Big Data Projects

According to the described aspects inherent in the big data domain, many differences can be noticed that distinguish a big data endeavor from a classic project with a strong focus on IT (IT project). Most of all, the focus is rather put on novel approaches and technologies to store, manage, and process large amounts of data [MK14; ZRI+16; MSD+14]. However, related IT projects that follow a similar data-driven direction are known even from times before the first occurrence of big data. In particular, this refers to *data mining*, which is *"also known as data discovery, [..] a specialized form of data analysis that targets large datasets"* [Erl16, p. 189]. Data science, as an often discussed evolution of this, can be described as *"a set of fundamental principles that support and guide the principled extraction of information and knowledge from data"* [PF13]. According to [MK14] big data-related skills intersect with those of a data scientist. Precisely, a *big data analyst* or *technologist* can also be a data scientist and vice versa. This conforms the definition provided by the NIST Working Group, who define a data scientist as *"a practitioner who has sufficient knowledge in the overlapping regimes of business needs, domain knowledge, analytical skills, and software and systems engineering to manage the end-to-end data processes in the analytics life cycle"* [CG19a].

By harnessing big data in IT projects, value at several stages can be created, including knowledge, organizational agility, business process, and competitive performance [COR17]. Notwithstanding that, by taking this information into account, one can assume that the procedural layer of a big data project realization differs from those of a classical IT or data mining project [MK14; MSD+14]. As such, related projects must consider numerous aspects that are prominent in this domain. Therefore, specific technologies, data characteristics, and other specificities that were not incorporated in such detail before, need to be observed [ZRI+16].

Various authors highlight standard methodologies that are known from the data mining domain and data-intensive application scenario, as a suitable foundation for a big data project [DB15; ZRI+16; Gra16]. Namely, those are the *Knowledge Discovery in Databases* process (KDD) [FPS96], the *Cross Industry Standard Process for Data Mining* (CRISP-DM) [CCK+00; Wir00], and the *Sample, Explore, Modify, Model,* and *Assess* method (SEMMA). Even though many researchers found out that those are not one-to-one applicable and modifications are required in a big data project, many highlight their importance and possibilities [SS16a; ZRI+16; DB15]. To better understand their high-level process flow and potential gaps, in the following, the previously mentioned approaches are described in more detail.

The KDD essentially describes a multi-staged procedure proposed in 1996 to overcome the scarcity of existing approaches that allow a computer-supported data analysis, discovering yet unknown knowledge of it. Predominantly it is intended for non-trivial data-driven processes that could be solved by *"a straightforward computation"* [FPS96,

p. 41]. Starting from the extraction, various preparation methods are used to preprocess, clean, transform, and analyze the data in order to identify patterns [FPS96]. These patterns have the potential to reveal new insights and knowledge, which, in turn, can later be interpreted and applied, for instance, in projects and scenarios [FPS96].

The procedure starts with the *basic identification and understanding* of the targeted application domain and the already available knowledge. After that, the relevant data is identified, which also includes relevant subsets, data samples, or variations. The *cleaning* and *preprocessing* are performed afterward. Then, the data is *transformed* in a way that a reduction and projection are made. As a result, only relevant data will be analyzed through the application of sophisticated data mining techniques. The found *patterns*, referred here as a valuable obstacle to the obtained results, need to be interpreted, and finally potential *knowledge* derived. Through the high dependencies and interactive nature, particular steps could be re-performed if needed [FPS96]. The stepwise procedure is depicted in Figure 2.6.

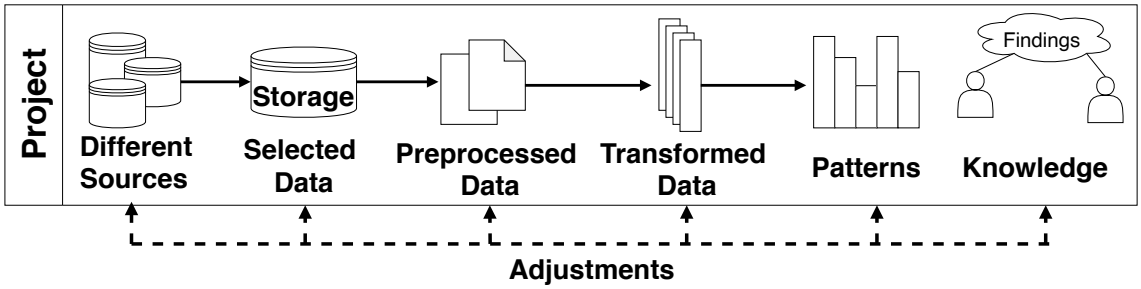


Figure 2.6: The KDD process based on [FPS96; VJT17]

Similar to the KDD the CRISP-DM process consists of six consecutive steps. Starting with the *business understanding*, an initial idea of the project, its objectives, and the relevant requirements are obtained to deliver a preliminary project plan. Then, the *data understanding* is observed in more detail. In particular, this includes collecting the data and some initial investigation steps. Since no real transition between the first two activities exists, high interdependency is possible. This also applies to the third and fourth steps. Within the third, the *data is prepared*. Here, essential efforts related to the preprocessing are targeted. As soon as the data is cleaned, transformed, and refined, these are used within the *modeling*. In this step, different data mining techniques are applied to obtain the intended results, or rather the yet unknown knowledge. Further *evaluations* are afterward performed in the adhering step. The solution is *deployed* once a final judgment about the received results can be given. The complete process can be seen in Figure 2.7. The circle shall symbolize the ongoing nature of a data mining project. According to the authors, a data mining project is never fully finished after the deployment [Wir00].

The SEMMA consists of five steps, forming the acronym. The data is collected within the *sample* step, extensive enough to provide novel insights. This is followed by the *explore*

step, in which anomalies, trends, and other specifities are searched. The third step, called *modify*, comprises the data preparation. Here, similar to the CRISP-DM all relevant data modifications, such as transformation, are recognized. After that, in the second last step of the *model*, the same is build. In the end, the evaluation is performed within the asses step [AS08].

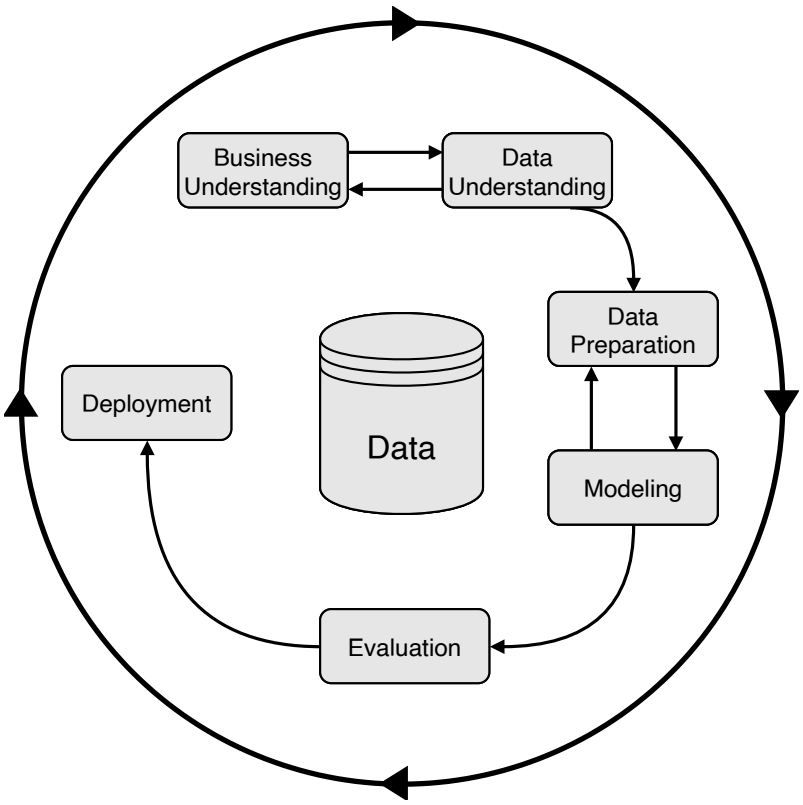


Figure 2.7: The CRISP-DM based on [Wir00]

By comparing these approaches, one can notice that they share a similar idea for conducting a data mining project. Only in terms of the comprehensiveness more significant differences can be ascertained [AS08]. Similar to the investigations and made findings here, numerous authors followed a similar discussion, such as [DB15; ZRI+16]. They attempted to fuse the aforementioned methods into structured processes tailored for the realization of big data projects. In [DB15] a comprehensive framework is provided that combines insights from a general project realization combined with the presented data mining methodologies. As a result, a framework is introduced consisting of ten steps that can be aligned to three overarching categories: *strategic groundwork*, *data analytics*, and *implementation*. A similar idea and setup were proposed in [MSD+14]. Here the three overarching activities are named *planning*, *implementation*, and *post-implementation*. Slightly different compared to these is the approach of [MK14], which put greater focus on project management-related aspects, such as a detailed requirements engineering that particularly considers the data characteristics. Notwithstanding that, the complexity is

highlighted to investigate the relevant requirements, data characteristics, and potential technologies in all of them. Although ideas for the realization are often shared, they lack in terms of a clear structure and level of detail. Instead, overarching ideas, generic guidelines, and best practices are provided.

One can conclude that big data projects pursue a similar implementation as other analytical scenarios in the context of these observations. Particularly this refers to the discussed specificities of an IT project (cf. section 2.1), and the investigated data mining approaches. However, these projects differ, above all, especially in terms of the particularities of the domain, including but not limited to the incorporation of the data characteristics, identification of suitable requirements, as well as the choice of the necessary technologies and their combination [CG19a; ZRI+16]. If not planned carefully, ongoing changes of the requirements, data characteristics, and other obstacles might appear, requiring a reevaluation of already made decisions. This may include the overall underestimation of the complexity of the targeted endeavor, carelessness by involved stakeholders, as well as insufficient communication between those [SVD+20]. Not least, the choice of the appropriate technologies also forms the basis of the architecture to be selected. Resulting from this discussion, we define a big data project in the context of this work as follows: *"A big data project can be described as an objective-oriented temporary endeavor with a precisely defined timeframe, whose implementation requires a combined use of big data technologies."*

The NTCP diamond model can be used to highlight the relevance of a dedicated big data project recognition and, thus, the importance of an elaborate and comprehensive approach capable of conducting those. As stated in section 2.1, IT projects are quite unique in their specifities. Apart from the time and resources spent, further classifications can be made. In [She07], the comprehensive approach is introduced in times before big data, which classifies a project according to the solution and its fulfillment in terms of different dimensions. These are namely *novelty (N)*, *technology (T)*, *complexity (C)*, and *pace (P)*. Each of them includes various levels that indicate the severity of this domain [She07]:

- **Novelty**
  - *Derivate*: the solution is an extension
  - *Platform*: new generation of something well-established
  - *Breakthrough*: the solution is something entirely new and was never seen before
- **Technology**
  - *Low-tech*: the solution harnesses only currently established technologies
  - *High-tech*: the solution mainly uses technologies that are unknown to the organization but already exist

  - *Super-high-tech*: the solution uses technologies that do not exist before the project initiation

- **Complexity**
  - *Assembly*: the solution represents a collection of different elements that are combined to fulfill a single function
  - *System*: the solution represents a collection of different (interactive) elements that perform multiple functions
  - *Array*: the solution represents a large collection of different systems that together fulfill a common purpose

- **Pace**
  - *Regularly*: the project is not critical for the success of the organization
  - *Fast/Competitive*: the project is intended to achieve competitiveness by approaching new opportunities and strategies
  - *Time-Critical*: the project needs to be finished by a specific date, otherwise it is considered a failure
  - *Blitz*: these projects are urgently important and are strict in time. They can be also described as crisis projects

The numerous novel technologies that are applied in the context of big data are mainly unknown to organizations, as indicated in the beginning. Although available, users never had any prior experience in many cases, which indicates the level here to high-tech. Mostly, the planned systems are intended to either provide multiple functionalities or shall serve a common purpose. Hence, these can be described as complex systems (cf. section 2.1) and as projects with at least a *system* complexity level. The novelty can be either aligned to a platform or breakthrough level in big data projects. Often, new ideas and approaches are being created or utilized to create further entirely new yet unknown solutions. Despite that, the pace is often not determined or cannot be judged from existing big data projects, as they were presented during the motivation and later on when presenting the SUC descriptions (cf. section 4.2), at least a competitive advantage is commonly required which puts the severity on the second level.

Eventually, all criteria define the foundation of a *large diamond* when it comes to the classification of a big data project. The graphical presentation, based on the NTCP diamond from [She07], is depicted in Figure 2.8. Again, this shall emphasize the complexity and importance of this domain. While the data characteristics were mentioned before, different big data technologies are described in the following, succeeded by architectural concepts. Every given detail, then, culminates in an extensive engineering approach, which is introduced the last sub-section of this sub-chapter.

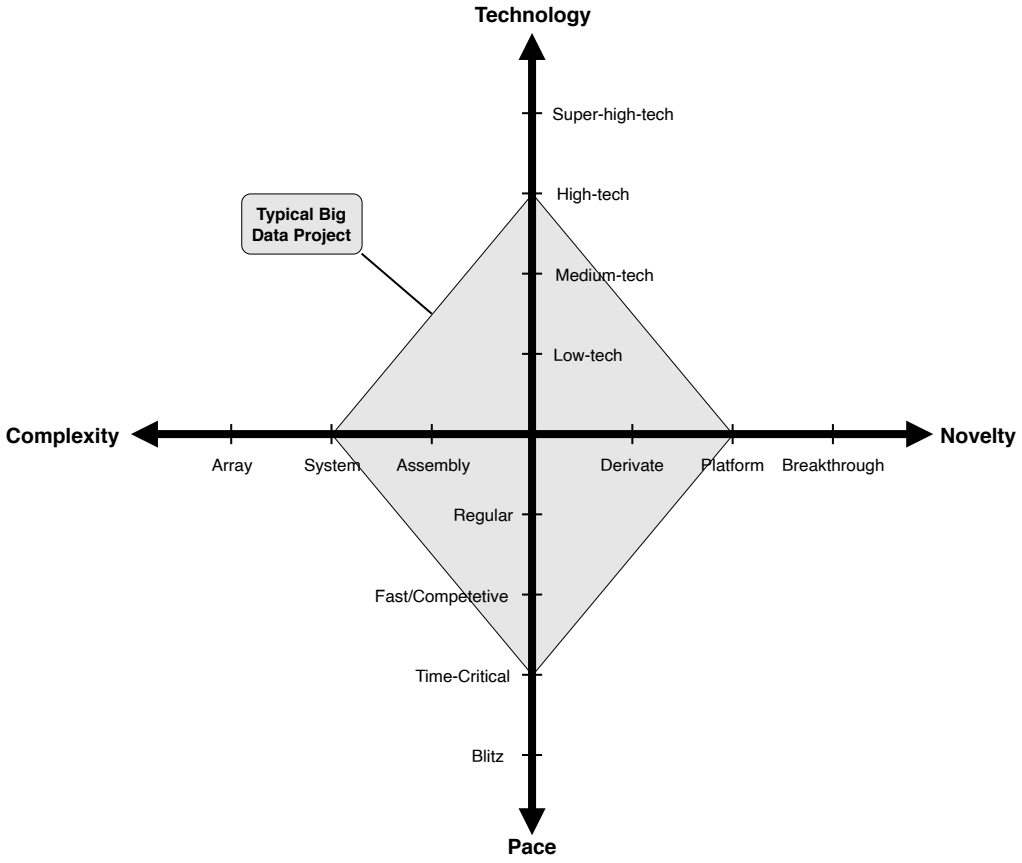


Figure 2.8: The NTCP model depicting a typical big data project, based on [She07]

2.2.4 Big Data Technologies

As highlighted in the previous sections, the realization of big data projects is often accompanied by numerous challenges. One of the arguably most important ones is selecting and implementing the corresponding technologies. The *high-tech* technologies required to develop the respective systems capable of fulfilling the project goals are often eponyms for the term big data (cf. section 2.2.1). Their distinctive features often lie in dealing with data that traditional, already established technologies are not capable of handling. Hence, some authors drive valid argumentations towards their *disruptive* nature [SHB+14]. With respect to the underlying data characteristics, one can easily perceive that even the storing of a massive amount of differently structured data can be considered a non-trivial undertaking. While for structured data, typically coming in classic database schemes, regular databases, as well as the *Structured Query Language* (SQL), can be used, for storing and managing semi-structured and unstructured data, *Not Only SQL* (NoSQL) databases are utilized [CG19a].

In addition, there are numerous technologies whose sole purpose is to cope with distinct expressions of data characteristics [OBA+17; MBB+20], such as massive processing of data in real-time, merging of different data sources, or functionally related to specific

disciplines, such as machine learning. It seems, therefore, not surprising that several hundreds of such technologies exist today, with their number constantly increasing [BJC16]. Matt Turck [Tur21] present a landscape encompassing existing technologies that can be assigned to big data and adhering domains on a yearly basis. Concurrently, a list is maintained, which backs the proposed landscape up. Although some of the entries are no longer valid, the sheer size of almost 1500 entries reinforces the problem of technology selection and, therefore, the aim of this thesis [Tur22].

Due to the large number of these solutions and the expert knowledge required for each of them, the following section describes general technologies and principles that are currently considered the de facto standard and are usually mentioned when discussing big data. Initial investigations of those, with the focus on possible classification approaches, as we conducted in [VBT17], have revealed that there is still disagreement on this topic among researchers and practitioners. A similar observation was made in another independent research, where existing classifications in the domain of big data were investigated [SVG+20]. Here, the systematic approaches in the form of taxonomies were targeted. Another comprehensive taxonomy that covers big data technology-related information was provided in [VF20]. Here, the authors name and describe ten different categories in which the technologies can be assigned according to the life cycle of the data in the systems.

Notwithstanding that, a disagreement for terms, definitions, and relations was also frequently ascertained. This includes, inter alia, a clear distinction of certain descriptions, such as (big data) technology, technique, and paradigm. Since these are frequently used in the context of this work, definitions for each of them, as they were derived in [VBT17] and further examined in [VPT18] are depicted in Table 2.3.

Term	**Definition**
Technology	A collection of systematic knowledge.
Technique	A specific procedure with the aim of achieving a certain result.
Paradigm	A superordinate design pattern.
Manifestation	Tool, product, and service as concrete instantiations of individual technologies.
Big Data Technology	A collection of systematic knowledge in the area of big data, whose application is used to deal with data-driven problems facing at least one of the underlying data characteristics.

Table 2.3: Relevant definitions related to big data technologies [VPT18]

Techniques related to this area can be, for instance, allocated to optimization methods, statistics, and data mining algorithms. A specific example for the latter could be the K-Means algorithm [PZ14a]. Related technologies can often be assigned to the respective

phases of the related data-intensive projects, as they were described in section 2.2.3. Namely, this comprises *data ingestion*, *data preparation*, *data analysis*, *data result delivery*, and *operation*. Although many researchers and practitioners deal with various definitions, the largest consensus exists regarding storage solutions in the big data domain. This was not only found out in a structured literature review [VBT17], but is also highlighted in numerous contributions, such as [Sha16; STG+15; RRS+16]. All of them refer to NoSQL databases as a key term for all non-relational database structures used to store data that does not come in a structured format. In particular, various types exist in which those can be classified. These are *key-value*, *document-oriented*, *column-based* and *graph-oriented* databases [CML14; Lea10; STG+15; VF20].

In the first form, the assignment takes place employing unique keys. Each of those is related to a specific value, which can be structured differently. Document-oriented databases are based on a similar concept, but a value is specified as a separate document. This document does not have a fixed structure and can be changed and extended at the user's convenience [CML14]. Column-based databases are another approach. Unlike the row-based approach, where all values are stored and processed on a row-by-row basis, the persistence and processing are done column by column. This special structure allows the data to be efficiently managed, partitioned, distributed, and queried [CML14]. However, problematic operations here are queries that are too specific, the insertion of new data that affect multiple columns, and the possible combination of different data sets. Graph databases are intended in particular for the representation of relations between data. While with relational databases, a representation of graph-like structures is connected with high expenditures as well as numerous tables and data sets, these can be converted in the appropriate databases without difficulty [BIT14]. Ultimately, none of the solutions can guarantee strict ACID compliance, which refers to the *atomicity*, *consistency*, *isolation*, and *durability* properties of each transaction [STG+15].

Especially the consistency, as it was previously introduced, is often problematic in that regard. Big data is defined, among other things, by the horizontal scalability and the parallelization [CG19a]. As a consequence, the data is not only persisted through multiple duplicated storages, also concurrent read and write operations occur. This leads, in contrast to traditional systems, to a possible forfeit of a strict consistency when the availability or partition tolerance shall be guaranteed. In general, this reflects one of the expressions of Eric Brewer's CAP theorem [Bre00b; Bre00a], which says that the *consistency*, *availability*, and *tolerance of network partitions* cannot always be fulfilled at the same time. Only two of these can be guaranteed simultaneously in distributed systems. Hence, in most cases, those NoSQL solutions only indicate an eventual consistency that ensures a consistent state amongst the different nodes of a distributed system at some point in time [HWC+14].

In order to not only store but use the large quantities of data of various types, specific methods and complementary technologies are necessary, which therefore have a special

significance in the environment of big data. One of the best-known solutions in this field, called Hadoop found its origin in 2003 with the Google File System release [GGL03], which is a scalable and distributed filesystem. It was later adapted and is now widely known as Hadoop Distributed File System (HDFS). In the following year, the MapReduce approach was published, which makes use of this principle through distributed computation [DG04]. The name is derived from the two successive phases *Map* and *Reduce*. In the first, the Map step, all existing data is divided into fragments and marked with a suitable key. Subsequently, these fragments are linked, and the respective partial results are passed on. After this intermediate step, all key-value pairs are merged in the Reduce phase. The user can customize the implementation of the Map and Reduce phase according to the respective application purpose [DG04].

Today, Hadoop, like many other big data technologies, is maintained and further developed by the voluntary non-profit organization Apache Software Foundation. The first version was released in 2006. Since then, it has been considered *the* solution in the big data space to tackle data-intensive problems. While Hadoop was initially considered merely a free, highly scalable implementation of the MapReduce paradigm that allowed batch processing of massive data, today, the term stands more for an entire ecosystem that includes numerous other big data technologies and extends the original functionality. Due to this, multiple research articles deal with its use, modification, and integration, such as [PRG+14]. In addition to the primary main components, MapReduce, HDFS, and the resource manager YARN (Yet Another Resource Negotiator), this also includes extensions such as Spark, HBase, Hive, and many others. An overview of these and their purpose within the ecosystem can be seen in Figure 2.9.

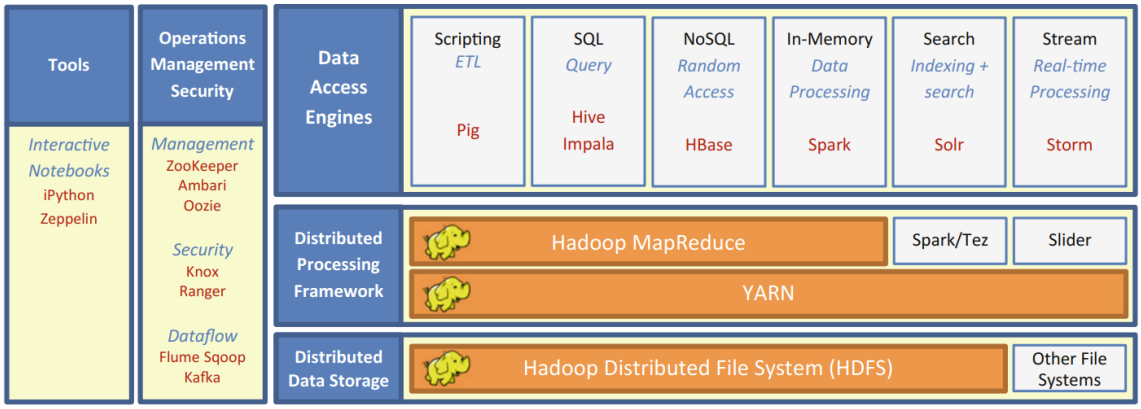


Figure 2.9: An incomplete excerpt of the Hadoop ecosystem, according to [Mro18]

Hadoop is primarily required for mass data processing that can be processed in batches, thus focusing on a low velocity. In turn, Apache's Spark aims at processing data in real-time where interactive analytics and streaming are possible. HBase, on the other hand, represents a distributed scalable storage in the Hadoop ecosystem that makes use of HDFS and is based on the BigTable approach developed by Google [CDG+06].

This resembles the row-based database system, at first sight, known from the relational databases. In contrast to these, for example, the number of rows, as well as columns, can be changed at any time, which provides greater flexibility. BigTable and, thus, also HBase can therefore instead be assigned to the NoSQL databases, more precisely to the column-oriented databases described above [CML14]. Storm is a prominent streaming big data technology that focuses on a high data velocity, required for (near) real-time processing scenarios, where the data need to be handled immediately [RRS+16; OBA+17].

Apart from Hadoop and its ecosystem, many authors discuss further technologies, application scenarios, and potential ideas in their research, with which a utilization can be assisted. A substantial question concerning each area was investigated in [VBT17]. Here, potential classification concepts and their extensibility were examined to shed light on the existing *jungle* of technologies. Although classification attempts were also performed in other studies, such as [HWC+14; RRS+16; MBB+20] they were often limited, permitting a further extension. For instance, Macak et al. [MBB+20] give insights into a potential classification of big data technologies, with the goal to facilitate decision support when it comes to a selection of potential solutions. However, only a few technologies are given, solely focusing on processing and storing the data. To facilitate further extensions, thorough revisions and refinements are needed, hampering the overall applicability. Hence, sustaining approaches are required, especially in fast-paced fields such as big data or related data-intensive domains, where new technologies quickly emerge.

This also confirms the initial criticism delivered towards technology-specific classification approaches once again. Very specific technologies, which are not covered by a base technology concept, such as a storage or processing solution, would drop out. Consequently, as found out in [VBT17], a classification that generally delivers applicable information about the different technologies, their interconnection, and the fulfilled functionalities appears to be promising. This is especially the case for discussed methodologies used to realize those projects. Multiple contributions have already attempted to perform something similar by either explicitly or implicitly following the data life cycle or individual steps within the known data mining process, namely KDD or CRISP-DM (cf. [VBT17] – Table 6). Despite the fact that those concepts provide essential information that *users could utilize*, they do not deliver specific details about their selection and composition into comprehensive system architectures. However, research and practitioners are aware of this circumstance and produced different baseline architectures that can be harnessed for related setups in recent years. These will be described in the following sub-section.

2.2.5 Big Data Architectures

The purposeful composition of big data technologies, in the form of a BDA, is the primary goal of the underlying engineering process and thus the goal of most of the related projects, as it is generally the case for system architectures [Taw20]. Compared to generic system

architectures (cf. section 2.1), a BDA can be defined as an *"architecture that provides the framework for reasoning with all forms of data. Thus, it is a logical structure of core elements used to store, access and manage the big data"* [IPO15]. Due to the quantity and diversity of available technologies [Tur22], their optimal selection is a non-trivial undertaking, especially with a view on the data characteristics that heavily influence those [AL20]. Sometimes, selecting the right technology for a particular problem can have significant consequences if, for example, compatibility with existing systems or among the technologies themselves is not given [OBA+17]. It, therefore, comes as no surprise that best practices have been established for specific deployment scenarios to help with the technical implementation [PP15].

Reference architectures are one example of this. These *"combine general architectural knowledge and experience with specific requirements to create an overall architectural solution for a specific problem area. They document the structures of the system, the essential system building blocks, their responsibilities and their interaction"* [VAC+09]. Thus, they represent a widely tested starting point for the design of concrete system architectures (cf. section section 2.1). This is also constituted by [ISO14], in which it is also emphasized that reference architectures are needed (1) to create a shared understanding of existing components, processes, and systems, (2) to provide a technical reference used for discussing existing big data solutions, and (3) to encourage the use and distribution of standards. Hence, in the following, some of the most prominent approaches are discussed to sensibilize for their complexity while delivering further information about detailed elements and their composition.

The *Lambda* architecture is considered the first and probably best-known reference architecture in this regard. The architecture, which consists of several levels, was developed by Nathan Marz [MW15]. These are the batch, serving, and speed level, which were introduced based on the problem of creating a scalable, expandable, and fault-tolerant architecture, capable of processing large volumes of data in near real-time while concurrently recognizing historical data. The batch level includes the basic data stock (master dataset), which is extended by the newly arriving data at runtime. Due to the problem that these cannot be considered and synthesized immediately during the batch processing, a pre-calculation of batch views occurs, which are transmitted to the serving layer, represented by a distributed database. The created views depict the results of the precalculation on the current inventory data. Immediate processing of new incoming data, on the other hand, takes place in the speed layer. In terms of the data to be processed, this level operates similarly to the batch level, except for the significant distinction that only data which was not applied during the current batch processing is taken into account. Analogous to the previously mentioned views, here, the partial results are recorded in real-time views. The final provisioning of the result that can be used for a further query, then, takes the combination of all layers into account, namely the speed and serving layer [MW15]. The composition of the reference architecture is depicted in Figure 2.10.

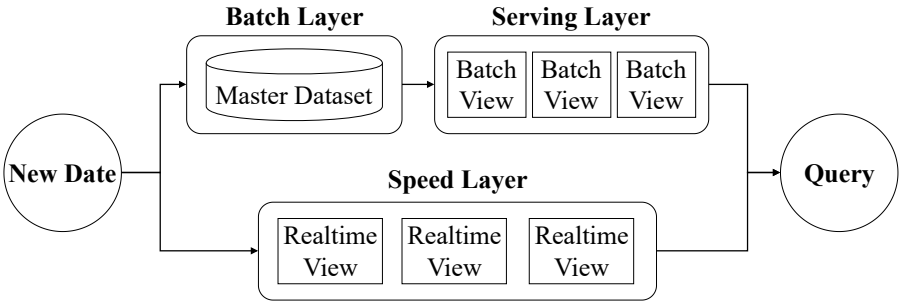


Figure 2.10: Schematic representation of the Lambda architecture [VST20]

While the former attempts to reduce the existing number of layers to reduce an extra effort in maintaining two processing layers [Kre14], the latter includes another semantic layer, which is mainly intended to improve the reliability of the data [NHR+17]. Compared to those generally applicable solutions, designed for specific problem areas, further, even more specified reference architectures exist that are tailored in respective contributions. Mostly, these narrow the problem area to very use case-specific problems, as constituted, for instance, by [APA+15; IFC20; PC18].

The commonly known levels of infrastructure/storage, computing/processing, and application/visualization can also be found in these architectures [HWC+14; AL20]. One benefit here is the provision of very specific technology recommendations, as they can be generally noticed for use case descriptions in this domain (cf. section 4.2). In contrast to these concrete approaches, however, there are also efforts made to provide generally applicable reference architectures designed through induction. Widely known approaches in this regard include, inter alia, the *NIST Big Data Reference Architecture* (NBDRA) designed by the NIST [CG19b] and the reference architecture presented in the contribution by Pääkkönen and Pakkala [PP15].

The *NBDRA* is built upon observing and investigating numerous independently collected use cases. All details of the component-based architecture are thoroughly described, together with the corresponding roles that are in constant interaction with it. These include, for example, the System Orchestrator, Data Provider, Data Consumer, and Fabric Roles. Each of these is described in detail in the corresponding documentation while keeping potential realizations as generic as possible: *"The baseline NBDRA does not show the underlying technologies, business considerations, and topological constraints, thus making it applicable to any kind of system approach and deployment"* [CG19b, p. 17].

A similar process was followed by Pääkkönen and Pakkala [PP15]. A generic reference architecture was built by investigating the architectures of large companies, such as Netflix, Facebook, and Twitter. It consists of different components in charge of the functionalities along the data lifecycle. Compared to other approaches, relevant technologies for the single steps are presented at the end of the contribution, extracted from use cases and through further research. An overview of the architecture and its elements is depicted

in Figure 2.11. Notably, in a scenario-based comparison conducted in previous research [VBB+19b], it was determined that the generic approach almost consistently outperforms other specific reference architectures, as they were named before.

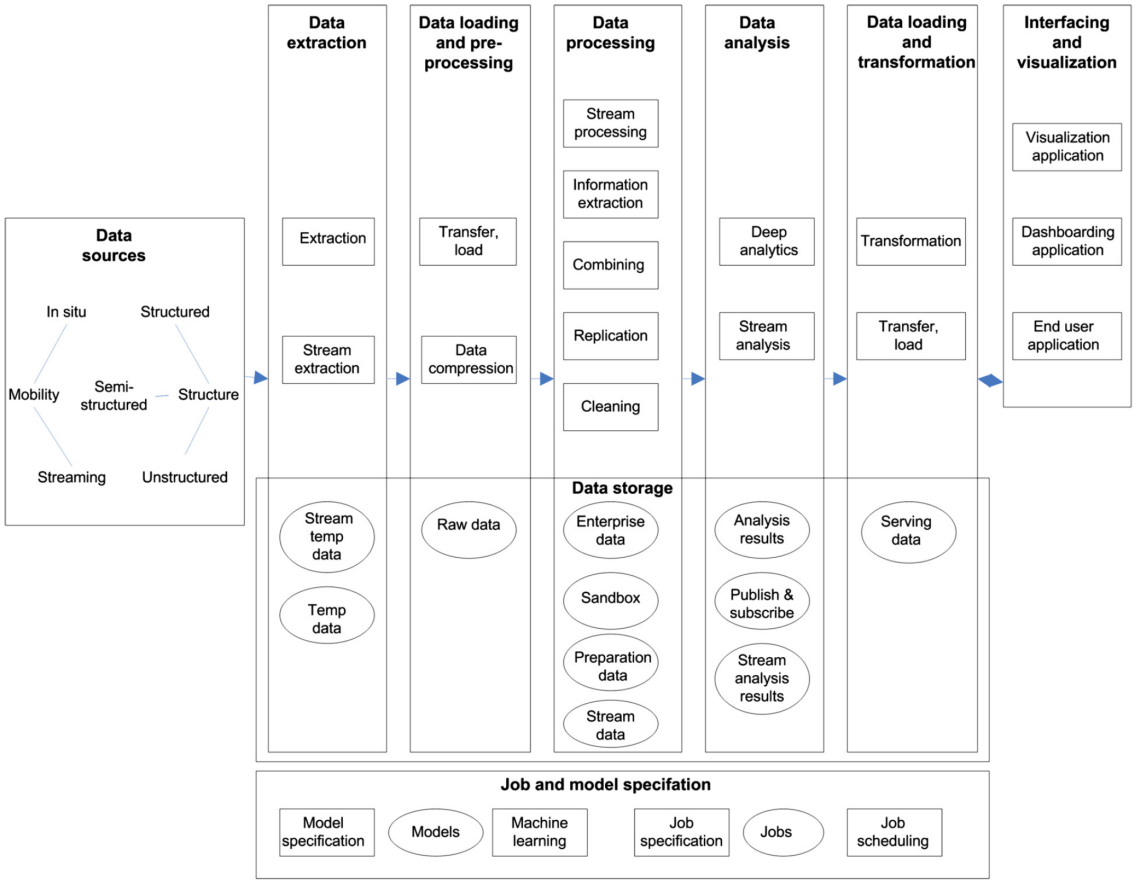


Figure 2.11: High-level big data reference architecture [PP15]

One of the highlighted reasons for this is the distinct nature between a generic concept (reference architecture) and the particular instantiation in an organizational context. Apart from detailed planning, in which all relevant stakeholders are sufficiently considered, they highlight that either a larger number or more detailed scenarios are created for a potential application of the reference architecture. Although something similar was already done in previous research [VBB+19b], utilizing a multi-criteria decision support method and scenario-based observations, with whose help a selection can be simplified, still, extensive information about the planned undertaking must be obtained in advance. To be able to support the system conceptualization as far as possible in advance [PZ14a], seven principles are provided that shall help with the design of a big data or rather data-intensive system and its implementation.

- **Principle 1.** Good architectures and frameworks are necessary and on the top priority: *Use of high-level architectures and frameworks that assist the creation of proper architecture for the big data system.*
- **Principle 2.** Support a variety of analytical methods: *Most of the requested tasks are very complex. Therefore, various disciplines and analytical methods should be considered and combined.*
- **Principle 3.** No size fits all: *Every existing big data tool has limitations to some extent, meaning that single solutions cannot solve everything. Hence, a combination of multiple tools appears to be promising, especially with the idea that the environment might change.*
- **Principle 4.** Bring the analysis to data: *Due to the volume of the data, the collection and processing cannot always be achieved. Instead, approaches are required to bring the analysis to the data.*
- **Principle 5.** Processing must be distributable for in-memory computation: *In-memory technologies may boost the overall processing, especially concerning real-time analytics. The processing needs to be distributed to facilitate the general applicability of specific solutions.*
- **Principle 6.** Data storage must be distributable for in-memory storage: *Here the focus is on cloud solutions if the data is stored in data centers. .*
- **Principle 7.** Coordination is needed between processing and data units: *Additional tools should be used to increase the overall efficiency of the system, referring to scalability, fault-tolerance, and the proper coordination between data and processing units (e.g., using tools like ZooKeeper).*

Even though not all of them are always fully applicable, they deliver essential insights that need to be considered when engineering a BDA. This applies first and foremost to the use of existing guidelines and architectures and a sensible combination of various technologies. The latter includes big data-related technologies, such as those described in section 2.2.4, and underlying base technologies, such as cloud computing. Furthermore, these systems need to be designed and constructed to ensure a long-lasting setup, suited for a specific application scenario. In turn, the first aspect sufficiently highlights the importance of the required baseline architectures and framework, which can be utilized for the setup. An initial attempt to deliver those (reference) architectures was made within this section. However, this represents just a small excerpt, as already stressed [AL20]. Even more detailed insights are discussed in the later stages of this work (cf. section 4.2). Notwithstanding that, at this point, the presented architectures highlight and emphasize the most important aspect, which needs to be considered here – the technologies. As highlighted by the third principle and various other authors [AL20; CG19b], no size fits it all, and for that reason, individual solutions are required for different use cases. Influenced by those and former considerations, the creation of relevant BDSA is thoroughly discussed.

2.2.6 Big Data Engineering

As comprehensively described and repeated throughout this chapter, the realization of big data projects differs significantly from conventional IT projects. Primarily, the collection, processing, and management of the data differ strongly in terms of volume, variety, and velocity (cf.sections 2.2.2 and 2.2.3). Novel technologies need to be identified and integrated, resulting in an increasing complexity for the engineering of related systems. Many researchers are aware of this problem. Hence, not only best practices in the form of guidelines, reference architectures, or potential classification approaches are developed. Also ideas for the specific BDE itself are proposed. In a report from the International Organization for Standardization (ISO), big data engineering (BDE) is explained as *"the storage and data manipulation technologies that leverage a collection of horizontally coupled resources to achieve a nearly linear scalability in performance"* [ISO14]. Yet, this statement ignores the specifics of the big data characteristics, possibly being too vague for practical application. Along with this, it doesn't accommodate the increased difficulty of testing due to the high complexity of the resulting systems [TG16; SHT19; SVN+19a; SVP+21]. Yingxu Wang provided a similar comprehensive definition as he highlights that it *"investigates analytic technologies for efficiently dealing with the inherent complexity and exponentially increasing demands in big data representation, acquisition, storage, organization, manipulation, searching, retrieval, distribution, standardization, consistency, and security"* [Wan15].

The NIST provided one of the most frequently used definitions of the term big data (cf. Table 2.1). However, they recognized only selected properties in their definition for BDE. According to them, BDE *"is the discipline for engineering scalable systems for data-intensive processing"* [CG19b]. Stemmed from the observations and explanations given prior, we extended the given descriptions and defined BDE as *"a systematic approach of designing, implementing, testing, running and maintaining scalable systems, combining software and hardware, that are able to gather, store, process and analyze huge volumes of varying data, even at high velocities"* [VSP+19]. Resulting out of this, the discipline conveys the good craftmanship for setting up big data systems and, thus, ultimately realizing big data projects.

To obtain a further overview about particular approaches in this domain, a literature review was conducted [VSP+19]. Eventually, it was revealed that in most of the cases, only the project realization [MSD+16; Gra16; DB15; LTO16] or specific activities needed for this were thoroughly investigated. This includes, for instance, the general planning, requirements engineering steps [ANN+17; DL20a], the identification of the suitable technologies [LFV16], and most of all relevant reference architectures, such as those described in the previous sub-section (cf.section 2.2.5). Especially the latter can be highly beneficial when it comes to the limitation of available options for technologies to be applied and guidance during the construction of the system. However, selecting these can be a very

demanding task, mainly due to the same reason as for big data technologies, where the number of available solutions can be overwhelming (cf. section 2.2.4). By thoroughly planning these activities, not only the overall probability of the success of a big data project can be greatly increased but also the required functionalities identified [KNZ18]. Hence, thorough planning and requirements engineering represent an initial step for constructing the needed system that is directly followed by activities that identify relevant components and detail them in terms of their connection and technological implementation. (cf. section 2.1.2). Considering the previously highlighted intersections and similarities to other domains, such as data mining, data science, and systems engineering, it becomes apparent that the discipline of BDE unites aspects from all of them, as illustrated in Figure 2.12.

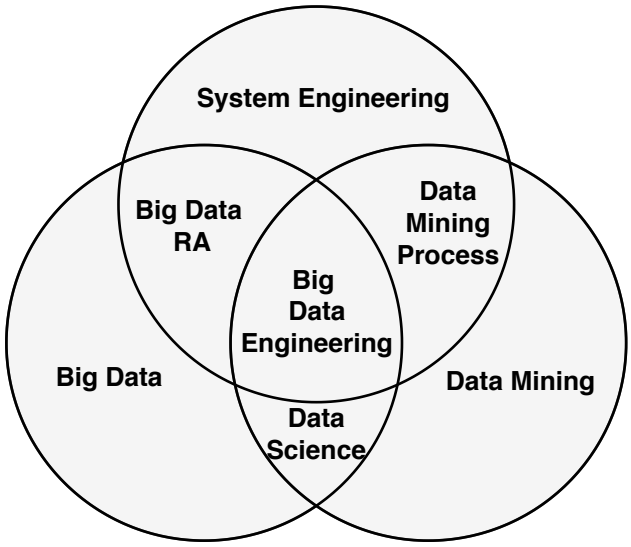


Figure 2.12: The intersection of the discussed domains

Many researchers are aware of this interconnection. However, the creation of related systems is often equated with software engineering rather than systems engineering, as described, for example, in [DL20a]. Although in the context of these, repeatedly holistic approaches are requested, the focus is usually put only on individual activities, which are in turn implicitly related to SE, methodologies for data-intensive systems, or other related fields. The common denominator for most authors is the successful realization of big data projects. Due to this, many attempts to provide guidelines for realizing such kinds of *projects* exist.

Dutta and Bose [DB15] introduced a holistic roadmap that attempts to guide organizations by the conceptualization, planning, and implementation of big data projects. An explicit connection between big data and the previously referred data mining processes was made in [Gra16]. Here big data is viewed from a technological perspective and forms, together with these workflows, the foundation of the data science domain. In particular, a mixture of the KDD, the CRISP-DM, and parts of the big data domain are described,

resulting in a five-stepped procedure that covers the *planning*, *collection*, *curating*, *analysis*, and *acting*. As reinforced by [EBF+21] established processes, known from the data mining domain, are today the foundation for *data-driven* projects where further *science* is required. In conjunction with such a data-centric view, as well as principles, technologies, and additional steps for the analyses form the data mining and big data domain. Data science can be seen as an intersection out of both.

Another process model that interconnects the KDD with big data was presented by Li et al. [LTO16] In their contribution, *a snail shell process model for knowledge discovery*, the proposed eight-stepped procedure heavily relies on the key activities used in the KDD process and involves the lifecycle presentation of the CRISP-DM model. A similar approach was found in [MSD+16]. The authors propose a big data project workflow that describes the realization *step-by-step*. Additionally to that, concrete technical implementation details, such as specific technologies, are addressed. These detailed system observations are even more concretized in [CKH+15], which proposes a new method called *Big Data System Design*. The procedure consists of ten essential steps, starting from the requirements analysis to the design and implementation. Here, reference architectures are considered a suitable foundation, as introduced and explained in section 2.2.5. The same applies to the implicit application of SE-related activities, such as decomposing the solution for better understanding. Compared to the previously described contributions, this work rather focuses on the technical implementation and thus the SE of big data-related systems. However, the theoretical background is little described, and data science-related activities are not included. IBM developed a step-by-step guide that extends the CRISP-DM. The Analytics Solutions Unified Method (ASUM) presents a hybrid approach that attempts to integrate agile as well as traditional principles in combination with big data relevant aspects [IBM16]. Yet, as in the case of the previous approaches, the process describes the needed steps without any concrete implementation details.

Again, in none of the approaches, a completed BDE process that enlightens the realization of big data projects was found, combining the data mining and systems engineering domain. Instead, different combinations of particular activities of all aforementioned domains were ascertained. Especially the CRISP-DM [She00] and SE methods as they were presented in section 2.1, were either implicitly or explicitly used. The prominence of this approach was also independently highlighted and discussed in [SDT14; EBF+21, p. 243].

Due to this, it can be argued that the linkage of both approaches, in addition to big data-related specifics, appears sensible. Although both approaches attempt to achieve different goals, a closer comparison of each of the related steps reveals similarities. This applies not only to the general problem identification (business understanding) and for problem analysis (data analysis) but to the solution construction (modeling), solution testing (evaluation), and solution delivery (deployment) as well. Differences, in turn, are predominantly noticeable in terms of the main scope. While the CRISP-DM intends to rather focus on the data analysis, the SE pursues the engineering of the implementation.

However, in both processes, the supplemented steps are implicitly integrated. Due to the aforementioned reasons above, a mixture of both approaches was chosen. In particular, the SE process from [MK15] is used as a base and extended by the thorough data investigation. The concrete workflow of the process as it was first published in [VSB+20a] is called the BDE process (BDEP) and depicted in Figure 2.13. Within this figure, the referred *foundation* comprises all steps of the SE procedure until the operation, in combination with the data understanding from the CRISP-DM. In contrast to the other measures, the data understanding was explicitly integrated due to the importance of the data being processed.

It comprises three levels, namely the *process steps (i)*, *content description (ii)*, and *focus (iii)*. On top, highlighted by the dotted line, the foundation from a data-driven domain is indicated, as it forms a baseline for most of the projects, using the CRISP-DM as reference (cf. section 2.2.3). Compared to other solutions, it provides a big picture of the overall steps, their content, and the interconnection to other steps, not only from a SE but also project planning perspective. Level *(i)* contains the general project steps and their interconnection. Each of them is described as execution directives. The second level *(ii)* provides a basic content description of the related step. The stated information should be covered and identified. Lastly, the overarching scope that is targeted by both levels is addressed by the last *(iii)*.

The process starts with the ideation *of the project foundation*. Due to the inherent concept of IT project realizations, the starting point is not necessarily limited to an existing *problem*. Moreover, the general description of a superior vision, a promising idea, or even a contract may initiate the procedure (cf. section 2.1). Independent from its origin, the detailed identification of the main scope is the result of this step. This serves as a transition to the in-depth analysis in the subsequent steps as an input. Within the *use case analysis*, potential scenario descriptions and use case diagrams need to be created to formalize the project foundation and its potential boundaries in a more straightforward way, such as highlighted in [CKH+15; Som16] and in previous sections. Apart from this, relevant stakeholders and especially the data to be used need to be determined here. For instance, if the data is gathered multiple times from a multitude of data sources, sophisticated orchestration activities are later on required [KES+16].

Furthermore, the specific characteristics should be uncovered due to the strong relationship between the data and requirements in data-intensive environments. Among other things, the template of Chen et al. [CKH+15] could be taken into account, which comprises 14 essential data requirements. Further, the RE step finishes the general planning of the projects by developing the FRs and NFRs as well as potential constraints. While the FRs define the general functions of the system to be performed, the NFRs focus on system properties [Som16]. Prioritizations and feasibility analysis can be helpful in this process step [NS21a].

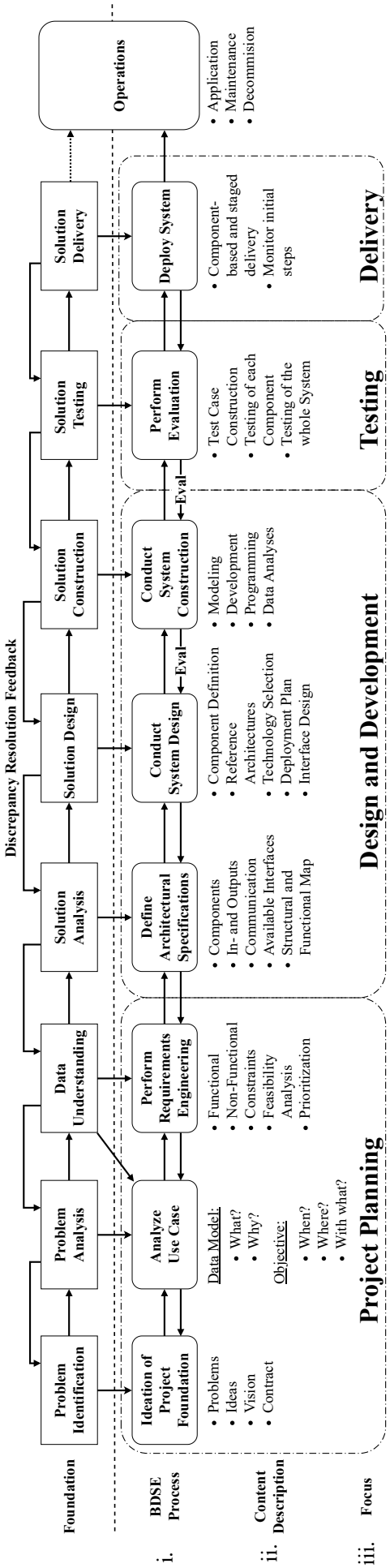


Figure 2.13: A Big Data Engineering Process (BDEP), based on [VSB+20a]

In any case, the requirements should be developed as thorough as possible to avoid massive changes to the system architecture at later development stages. After the project planning is finished, the design and development takes place. In the beginning, specifications are needed, where basic components, their relation, the provided functionalities, the overall performance, and future tests are defined [MK15; NS21b].

Additional inputs and outputs and available interfaces are relevant in terms of this. For better depiction and understanding of the system, structural and functional maps could be used, known from the system decomposition [MK15]. After each of the needed *elements* is determined, the system design is conducted. This includes, most of all, the definition of the component specifics. In the area of big data, multiple technologies exist that can be used for different purposes (cf. section 2.2.4). Hence, the adequate selection of suitable solutions during the specification of the required architecture represents a sophisticated undertaking.

At this point, best practices [PP15], reference architectures [MCA+15; Kre14; NHR+17] and DSSs appear to be useful, as found out in previous research [VSP+19]. All of them intend to provide general guidelines for the construction of the system architecture and, in parts, concrete implementation details and technological recommendations. In any case, the requirements originating from the previous phase need to be discussed thoroughly. However, the decisions are not always final, and in some cases, further modifications are required, for instance, in terms of the technologies or patterns to be used [MK15; LTO16]. After all of the required elements and their interconnections are identified, the actual combination and construction of the solution take place. Apart from the development of the system itself, this includes the programming or modeling of the needed application running on the system. After the solution is constructed, it needs to be evaluated, examining the overall correctness. For that reason, a thorough testing procedure is needed, comprising significant test cases that cover the validation of the separate components and the system as a whole.

However, the properties of the big data domain turn this into a highly sophisticated task [SVN+19b]. It is necessary to cover a variety of technologies, types, and sources of data, connections, and requirements. At the same time, the demand for future scalability and an often prevailing lack of knowledge regarding the correct outcome, which complicates a verification, pose additional challenges. Furthermore, even minor flaws like, for example, rounding errors can be built up during the processing, amounting to considerable derivations from the correct result [YAB+18]. At the same time, while being highly important and complex, the testing of big data applications is not sufficiently acknowledged in the literature [SVJ+19]. Despite that, this problem is not further focused on in the following course of this work. Subsequently to the successful evaluation, the developed solution can be deployed. In the context of the described process, this step refers to the actual distribution of the solution in the targeted environment.

In the case of complex systems, Mobus and Kalton [MK15] highlight that this should

be realized in a staged process to uncover unforeseen issues. Especially in the domain of big data, this should be recognized. Due to the high number of existing technologies and their versions, compatibility issues can quickly emerge. This is not only restricted to the dependencies between the used components but also the targeted environment [CKH+15]. Hence, during the delivery, comprehensive monitoring activities are required. Eventually, the actual application of the developed solution and its further maintenance will be performed during the operation phase. As prescribed in most of the existing approaches, for each problem encountered in one of the steps, considerations, and tasks of a previous step should be revised. Agile principles applied in each of them, as well as the automation to some degree, might give additional support, following some of the recommended procedures [KNZ18]. Hence, also computer-supported assistance might be helpful at this point, especially with regard to the tremendous amount of information that need to be discovered and processed.

2.3 Decision Support Systems

Making decisions is part of every person's daily routine. While for some, it is simply a question about the best possible means of transport or the choice of clothing on a particular day, far more complex decisions can have severe and long-term consequences. Especially in an organizational context, where business goals have to be achieved, complex processes come to light that requires far greater forethought. In addition to a clear understanding of the actual problem, the criteria to be considered, and the actual implementation, possible consequences must also be weighed. Hence, the role of a manager who is often confronted with such kind of decision can sometimes be sophisticated and cumbersome. In the early 1960s Herbert Simon delivered the *intelligence-design-choice* paradigm to approach such complex decision-making processes in a structured way [Sim77].

Within the initial *intelligence* phase, the considered problem is thoroughly investigated, including, among other things, the identification of the environment, data, and primary objective. After that, in the *design* phase, a model is created that denotes a simplified version of the reality, concurrently neglecting complex relations and constraints. Here, essential decision variables are considered to describe the potential alternatives the decision-maker may choose. Eventually, a possible solution is selected within the *choice* phase and tested regarding its viability. Depending on the type of the model the choice can be differently complex and structured. For instance, normative models attempt to provide the best alternative for a potential problem, representing *optimization*. At this point, thorough investigations and comparisons often have to be made, at which sophisticated procedures, such as multi-criteria decision-making methods, play a decisive role [SDT14; Sim77, 73–79]. An overview of this process, including also the additional implementation stage that was later inserted for solving the real-world problem, can be found in Figure 2.14.

Today, the presented paradigm is considered the basis of DSSs. In the early 1970s, the term was articulated for the first time by Gorry and Scott-Morton [SDT14, p. 43]. They describe DSSs as *"interactive computer-based systems, which help decision makers utilize data and models to solve unstructured problems"* [GS71]. Although there is currently no established and universally applied definition and most probably will never be, due to the content-free expression, such a system can be defined as a *"computer-based support system for management decision makers who deal with semi-structured problems"* [SDT14, p. 43]. Compared to decision-making systems, they provide potential solutions that can but ultimately do not have to be used [Alt76]. Hence, in conformance with this, the final output does not only provide the most suitable solution for an existing problem but, at the same time, the ranking of potential alternatives. Thus, apart from a user interface, for the representation of the results and interaction, a processing engine for the choice stage, as well as further elements, can be found in such a system. Those are further described in section 2.3.1.

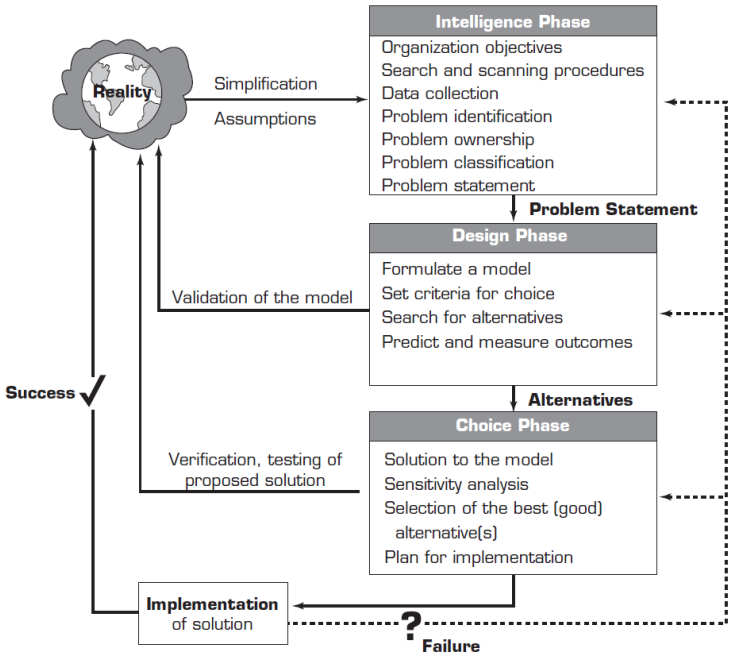


Figure 2.14: Decision making paradigm [SDT14, p. 74]

Based on the research conducted by [Pow08] various types of DSS that evolved over time can be identified. These are *model-driven*, *data-driven*, *communications-driven*, *document-driven*, and *knowledge-driven* DSSs. The first builds on the foundation for today's systems. In initial approaches, simple calculations and a small knowledge base were sufficient to support the decision-making (e.g., in the financial sector). With the shift of the focus on the underlying data, respective approaches emerged.

Data-driven DSSs observe current and historical data to facilitate elaborated decisions. By using database and data warehouse technologies as well as sophisticated methods,

thorough observations are feasible. Compared to those solutions, communication-based systems harness the capabilities of networking technologies with which people can work together. Consequently, the decision process can be seen as a group task here.

Document-driven DSSs, in turn, are used to retrieve and analyze documents. Often these are connected with search engines to provide the correct documents for a specific purpose. On the other hand, knowledge-driven DSSs suggest potential actions to managers by harnessing particular problem-solving skills. The problems to be solved are commonly related to a particular domain, which requires expert knowledge. Many of today's systems are developed with a web-based frontend, and sophisticated technologies to overcome recent problems, such as fraud detection or the scheduling in manufacturing operations [Pow08]. A similar overview of existing types was independently provided in [SDT14, 91–94]. However, additionally to the ones described here, further are introduced, such as compound DSSs, which denote a mixture of two or more of the aforementioned types.

Since such systems are often created for users who are not experts in this field, ease of installation and use is one of the main concerns [DHO+13]. In this sense, they are, for example, solutions that can be used regardless of location, such as web applications. Furthermore, the general support of semi-structured and unstructured problems is also addressed in this context. These support the interactive decision-making process rather than performing it autonomously [Pow02]. An exceedingly extensive collection of characteristics can be found in [SDT14, p. 90], extending those already mentioned. Together with a short description of the relevant facts, each of those can be found in Table 2.4.

No.	Characteristic	Description
1.	Semi-structured or unstructured problems	Support for decision makers, mainly in semi-structured and unstructured situations, by bringing together human judgment and computerized information. Generally, these problems gain structure as the DSS is developed. Even some structured problems have been solved by DSS.
2.	Support managers at all levels	Support for all managerial levels, ranging from top executives to line managers.
3.	Support individuals and groups	Support for individuals as well as groups. DSSs support virtual teams through collaborative web tools. DSSs have been developed to support individual and group work and individual decision-making and groups of decision-makers working somewhat independently.
4.	Interdependent or sequential decisions	Support of decisions that can be either made interdependently or in sequential orders, where multiple aspects need to be observed. The decisions may be made once, several times, or repeatedly.

Table 2.4 continued from previous page

No.	Characteristic	Description
5.	Support intelligence, design, choice, and implementation	Support in all phases of the decision-making process: intelligence, design, choice, and implementation.
6.	Support a variety of decision processes and styles	Support for a variety of decision-making processes and styles.
7.	Adaptable and flexible	The decision maker should be reactive, able to confront changing conditions quickly, and able to adapt the DSS to meet these changes. DSSs are flexible, so users can add, delete, combine, change, or rearrange basic elements. They are also flexible in that they can be readily modified to solve other, similar problems.
8.	Interactive, ease of use	User-friendliness, strong graphical capabilities, and a natural language interactive human–machine interface can greatly increase the effectiveness of DSSs. Most new DSSs applications use web-based interfaces or mobile platform interfaces.
9.	Effectiveness and efficiency	Improvement of the effectiveness of decision making (e.g., accuracy, timeliness, quality) rather than its efficiency (e.g., the cost of making decisions). When DSSs are deployed, decision-making often takes longer, but the decisions are better.
10.	Humans control the process	The decision maker has complete control over all steps of the decision-making process in solving a problem. A DSS specifically aims to support, not to replace, the decision maker.
11.	Ease of development by end users	End users are able to develop and modify simple systems by themselves.
12.	Modeling and analysis	Models are generally utilized to analyze decision-making situations. The modeling capability enables experimentation with different strategies under different configurations.
13.	Data access	Access is provided to a variety of data sources, formats, and types, including e.g. geoinformation system (GIS), multimedia, and object-oriented data.

Table 2.4 continued from previous page

No.	Characteristic	Description
14.	Stand-alone, integration, and Web-based	The DSS can be employed as a stand-alone tool used by an individual decision maker in one location or distributed throughout an organization or in several organizations along the supply chain. It can be integrated with other DSS and-or applications, and it can be distributed internally and externally, using networking and web technologies.

Table 2.4: Characteristics of decision support systems [SDT14, p. 90]

2.3.1 Basic Composition

As mentioned before, generally, those systems consist of different components, in charge of data management, processing as well as the communication with a user, and presentation of the results [LDW+10]. Sometimes different descriptions or additional components are given. However, all of them share a similar understanding of the overall conceptualization of the system. In [W H08], these are referred to as a *language*, *presentation knowledge*, and *problem-processing system component*. Indepdentelly Power et al. [Pow02] indicates that the use and composition of the components can be highly dependent on the type of the DSS. Some of the types, such as model-driven DSS, only require simple flat-file databases for the database components, whose structure is similar to a relational database. An example of this are comma-separated value (CSV) files. In turn, knowledge-driven and document-driven DSSs require specific, even more sophisticated solutions, e.g., capabilities to hold unstructured data. NoSQL databases, as they were presented in section 2.2.4, would be applicable for this. Apart from that, other concepts for the actual knowledge base can also be used, such as ontologies, [RS12]. These will be further discussed in section 2.4.

Notwithstanding that, database components, which hold the data, knowledge, and required documents, are just one of those system elements. Beyond that, generic and well-established approaches comprise a *user interface* component as well as a *model* component as further elements [Pow02]. The user interface component is mainly in charge of interactions with the user and is often seen as a critical success factor of the system. Hence, it takes not only the needed input and feedback from a user, results of the decision-making process are also presented. A well-designed graphical user interface (GUI) is often highly aspirational at this point [Pow02; SDT14]. The model component is responsible for the provisioning of the actual decision support. It can be rather seen as the inference engine, depicting *"the brain of the system"* [SDT14, p. 515], which takes the given information to

deliver potential decision support, for example, by using logical reasoning or complex calculations [SDT14, p. 521]. The communication component fulfills the role of the mediator as *"it refers to how hardware is organized, how software and data are distributed in the system, and how components of the system are integrated and connected"* [Pow02, p. 18]. Thus, it intends to connect the user interface with the remaining components, for instance, by a specialized architecture, networks, webserver, and client/server setups. Especially the popularity for the distribution and application as a web-based solution increased with the rise of the internet [LDW+10]. An overview of all components and their interconnection is, using a practical example of financial data, depicted in Figure 2.15

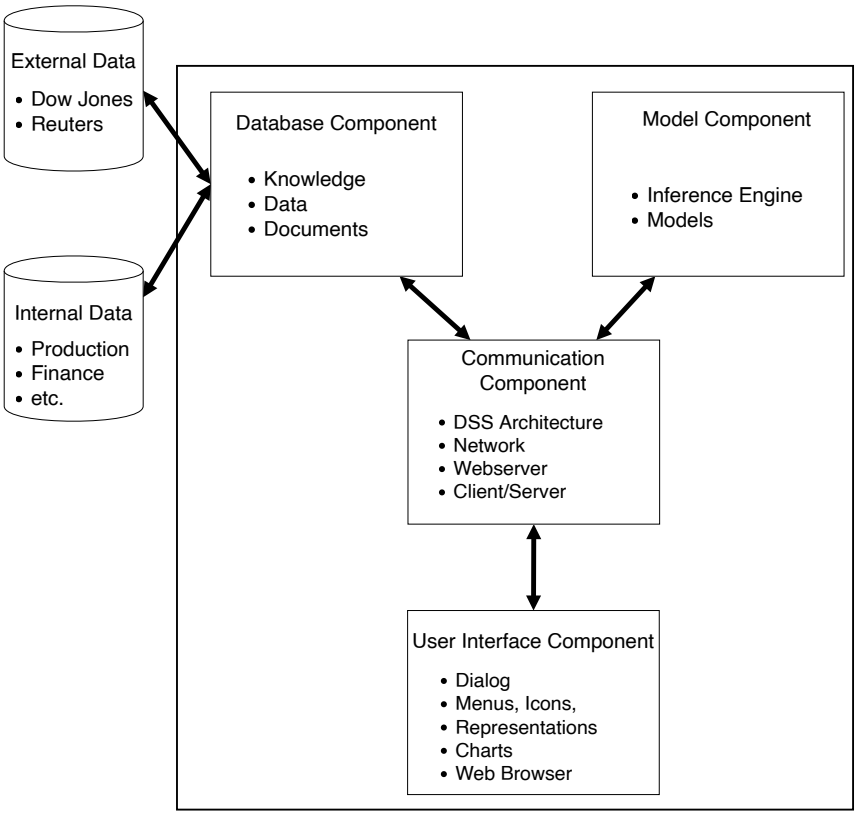


Figure 2.15: Structure of DSS according to [Pow02, p. 17]

Other authors extend this idea and introduce additional elements and connections, which observe those systems, especially in an organizational context. In [SDT14, p. 94] additional data storage, networks, and organizational specificities are considered. Data management and external models are then used for the connection with those supplementary elements. However, because the planned solution of this work is not exclusively used for an organizational context, only the basic components are highlighted as they were depicted and described before (cf. Figure 2.15).

DSSs have been the subject of research for decades. Although many of the fundamentals presented here are now considered standard, despite their disagreement on certain

points, research in this area has intensified precisely with the rise of big data and artificial intelligence. With the massive increase of the amount and complexity of data being produced and the corresponding commissioning of systems capable of processing those, many application scenarios emerged. In these, DSSs delivered meaningful insights and helped decision-makers from science and practice to overcome current problems, such as described in Ijadi Maghsoodi et al. [IRH+20]. Today, there are numerous contributions that examine not only the impact of the data to be processed in the context of big data [Fre18], but also the technologies themselves [PH15].

2.3.2 Multi-Criteria-Decision-Making (MCDM)

To ensure that the brain of the system delivers sufficient support for the decision-making process, often sophisticated methods are applied. Multi-Criteria-Decision-Making (MCDM), as one of the core disciplines in decision making and, thus, DSS research, *"deals with a general class of problems that involve multiple attributes, objectives, and goals"* [Eom08, p. 144]. Other authors further categorize existing approaches in multi-attribute decision making (MADM) and multi-objective decision making (MODM) that distinguish not only in terms of the number of attributes and objectives but also regarding further aspects, such as the interaction with the user [SER15].

Independent from the definition, classification, and application, several methods are frequently named and utilized when it comes to observing multiple criteria for decision making. This includes, inter alia, the *Analytical Hierarchy Process* (AHP), *Analytical Network Process* (ANP), *Technique for Order Preference by Similarity to Ideal Solution* (TOPSIS), *ELimination Et Choix Traduisant la REalité* (ELECTRE), and the *Preference Ranking Organization Method for Enrichment Evaluation* (PROMETHEE). Each of them is shortly summarized and described in the following sub-section. However, the AHP is specified in more detail, as already performed in previous research [VBB+19b]. This is not only due to its prominence in research, and practice [RA15; PP18a], especially in the field of big data [Lně15; HEE21; BHA+17]. Additionally, it forms one of the core concepts that are later on used within the created artifact of this work.

The AHP is presumably the most prominent approach for MCDM. The application is initiated by a decision matrix, similar to most of the MCDM algorithms. It depicts the pairwise comparisons of different criteria regarding the user's individual preferences. Each of the criteria will be compared to all others, resulting in a total of $(n^2 - n)/2$ comparisons, in which n is the total number of criteria of a single procedure. According to Thomas L. Saaty [Saa08] a value that ranges between one and nine for each comparison is recommended for the assignment. While a value of one means that both criteria are equally important, a value of nine reflects an *"extreme importance"*. In turn, intermediate ratings are intended to highlight slight preferences. An overview of all values and their meaning is depicted in Table 2.5.

Value	Interpretation of the given value for the compared criterion
1	Both criteria are equally important and contribute in the same way to the objective
2	The criterion is slightly more important than the other
3	The criterion has moderate importance compared to the other. Based on personal judgment and experience, one criterion appears to be a bit more important
4	The criterion has moderate plus importance compared to the other
5	The criterion has a strong importance, resulting in a strong favor for one of them
6	The criterion has strong plus importance compared to the other
7	The criterion has very strong importance, which also becomes visible by existing demonstrations in practice
8	The criterion has very, very strong importance compared to the other
9	The criterion is extremely important compared to the other. Literally, the other criteria could be neglected compared to this one

Table 2.5: Rating scale of the AHP according to [Saa08]

The calculatory equivalent, necessary for further computation, is expressed by the reciprocal value in the same matrix by taking the inverse position in the matrix. All comparisons, as well as the reciprocal values, are stored in a $n \times n$ identity matrix, called a comparison matrix. Basically, this is a zero matrix with ones on the diagonal. Each criterion is depicted by one row and one column, both with a similar index. All values above the main diagonal represent the comparison ratings and, in turn, all values below the reciprocal value (1).

After all comparisons have been made, a normalized matrix is generated, by dividing each value of a specific column by the sum of the same column. After normalizing the matrix, the average of each row is calculated, indicating the absolute priority of the targeted criteria. The higher an entry, the higher is the importance of the criteria. By using all of the obtained values, for each row, the weighting vector W can be created (2). In order to avoid an undeliberated decision, the consistency of the made comparisons can be further assessed. First, the initial comparison matrix needs to be multiplied with W to obtain the weighted sum vector Ws. Then the reciprocal is formed of each element of W that is followed by the calculation of the scalar product of both vectors: $Ws \circ \{W\}^{-} 1$. The number of compared elements then divides the resulting consistency vector, to obtain λ_{Max} as the eigenvalue (3). This value is required to calculate the consistency index (CI).

Finally, with the calculation of the consistency ratio (CR), a comparison of the CI with a randomized consistency index (RI) takes place, to measure the consistency of the given scores (5). If CR is less than or equal to *0.10*, all of the given ratings are consistent,

otherwise individual pairwise comparisons should be reviewed again. When all values of the criteria have been determined and W calculated, the alternatives are inspected in a similar way. Here for each criterion, an identity matrix is calculated where each row and each column depicts existing alternatives for the eventual decision. Hence, the procedure remains the same as described above.

Each alternative within a identify matrix is compared and rated, stemmed from one specific criterion. For each of the criteria, a weighting vector W needs to be calculated (6). This continues until each of them has been checked in terms of the available alternatives. Further, all W vectors will be brought together into one matrix, highlighting the connection between the alternatives and criteria. Finally, this newly created matrix will be multiplied with the initially calculated vector W from the criteria comparison matrix (cf. step 2.). The resulting vector describes the suitability of each alternative, based on the made comparison (7). An exemplarily realization is depicted in Figure 2.16, highlighting that the alternative A_1 appears to be the best solution for this particular case. For a better understanding, each of the previously described steps is linked to the figure via the corresponding number within the brackets.

Criteria Comparison Matrix (1)

a_{ik}	a_{i1}	a_{i2}	a_{i3}	a_{i4}
a_{1k}	1	5	4	1/3
a_{2k}	1/5	1	1/3	1/5
a_{3k}	1/4	3	1	1/5
a_{4k}	3	5	5	1
$\sum$	4.45	14	10.33	1.73

Normalized Matrix (2)

a_{ik}	a_{i1}	a_{i2}	a_{i3}	a_{i4}	W	Ws
a_{1k}	0.22	0.36	0.39	0.19	0.29	1.28
a_{2k}	0.04	0.07	0.03	0.12	0.07	0.27
a_{3k}	0.06	0.21	0.10	0.12	0.12	0.50
a_{4k}	0.67	0.36	0.48	0.58	0.52	2.33
$\sum$	1	1	1	1	1	

Eigenvalue Calculation (3)

$$\lambda_{Max} = \frac{(Ws \circ \{W\}^{-1})}{n} = 4.26$$

Consistency Ratio Calculation (4)

$$CI = \frac{\lambda_{Max} - n}{n-1} = 0.086 \quad CR = \frac{CI}{RI} = 0.096$$

$$\boldsymbol{CR \leq 0.1}$$

n	1	2	3	4	5	6	7	8	9	10
RI	0	0	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49

Random Consistency Index (5)

a_1	A_{i1}	A_{i2}	A_{i3}	W_{a1}
A_{1k}	1	3	6	0.61
A_{2k}	1/3	1	1/5	0.12
A_{3k}	1/6	5	1	0.27
$\sum$	1.5	9	7.20	1

Alternatives Comparison Matrix for Criterion a_1 (6)

	a_1	a_2	a_3	a_4	W	Result
A_1	0.61	0.32	0.18	0.40	0.29	0.43
A_2	0.12	0.29	0.54	0.12	0.07	0.18
A_3	0.27	0.39	0.28	0.48	0.12	0.39
$\sum$	1	1	1	1	0.52	

Final and Transposed Rating Matrix (7)

Figure 2.16: An exemplarily calculation of the AHP [VBB+19b]

The *ANP* is a modification of the AHP, which was likewise developed by Thomas L. Saaty [SV13]. While the classic AHP approach seeks to provide a linear hierarchy, structuring the problem into a superordinate goal, criteria, and sub-criteria to be observed as well as existing alternatives, the ANP permits interdependencies between the individual criteria in the form of a *network* to be taken into account [Saa16]. The required information from a decision maker does not change in comparison to the AHP. However, this does not apply to the execution of the procedure. In order to show and consider the dependencies between the criteria, a supermatrix, as well as iterative comparisons, are required. Instead of making only one comparison on the respective level, as in the case of the AHP, several comparisons are required. Here, mutual dependencies are taken into account for each comparison [SV13].

The *TOPSIS* algorithm was developed by Hwang and Yoon [Hwa81]. This method attempts to find *"an ideal and an anti-ideal solution and comparing the distance of each one of the alternatives to those"* [PP18c]. To achieve this, six consecutive steps are performed. First, a decision matrix is created, which is normalized to eliminate, for instance, any units. Then, in the second step, a weight is formed for each criterion multiplied by the normalized matrix afterward. The third step determines an ideal solution and an anti-ideal solution, representing the least recommended solution. In the fourth step, the distance of all alternatives to the best and worst solutions is calculated. The distance measure used here is usually the Euclidean, Hamming, or Manhattan distance. The relative closeness to the ideal solution is calculated for all values in the fifth step. Finally, in the sixth step, all alternatives are sorted according to their relative proximity [PP18c]

ELECTRE represents a family of MCDM algorithms whose origin started in the 1960s [FGR+13]. Over time various differentiations arose, such as ELECTRE I, II, III, IV or TRI [FMR16]. These methods can be used for four different reasons for which decisions need to be made. In particular, those comprise preference situations where (1) complete indifference between two actions exists, (2) a strict preference in favor of one action exists, (3) a weak preference is given, or (4) that any reasons are missing to justify any preferences [FGR+13]. Although they are very specific regarding the applicability, they all share a similar setup. They allow the outranking of different alternatives based on a given number of actions. These are *choosing*, *ranking*, and *sorting* [FMR16]. Compared to the AHP algorithm, used *weights* for the observation of single criteria are not bonded to given ranges. Here for indifferences and preferences, a pseudo-criterion model is built.

PROMETHEE is another collection of outranking methods, where different actions of an alternative are compared to provide a ranking of those. In 1982, the initial two approaches, namely PROMETHEE I and II were proposed by J.P. Brans. In the upcoming years, similar to ELECTRE, further approaches arose, resulting in a total number of six methods, each with a specific application purpose and properties [BS16]. The input is similar to other algorithms, such as TOPSIS [PP18b]. Especially for the first and second PROMETHEE algorithms, pairwise comparisons are used to create *outranking flows* of

the existing alternatives. Notably, while the first may deliver no particular preference for an alternative in specific cases, the second provides a complete ranking flow that highlights the position of each alternative [BS16].

A problem that can be noted in most algorithms, namely AHP, ANP, TOPSIS and PROMETHEE, is the rank reversal phenomenon, which creates a new ranking of the alternatives if a new option is added, even though the same preferences are given. Numerous studies dealt with suitable approaches to overcome this problem, as highlighted in [PP18a; BS16]. One frequently cited contribution by Wang and Elhag [WE06] provides the general suggestion to avoid changes of the criteria of all previous alternatives, in case of further additions or removals. Furthermore, if the selected criteria are sometimes neglected, the rank reversal problem should also not have a drastic impact and, thus, could be acceptable.

After highlighting the algorithms used within an inference engine, as they are frequently used for unstructured problems in knowledge-driven DSS (cf. section 2.3), another important semantic technology is introduced that can be used as a way to store and provide knowledge (knowledge base) within related systems. Namely, those are ontologies.

2.4 Ontologies

In this sub-chapter, essential information about the topic of ontologies is provided. As an integral component of the later artifact, necessary details are shared about the nature, structure, design and development, and application of those. First and foremost, this shall emphasize why an ontology, as a complex solution, was preferred over a flat-file format, a relational or NoSQL database (cf. section 2.3). Most of the basic information described here is taken from the previously published conference [VPT18] and extended journal paper [VSJ+20] that targeted the ontology created for this work.

Ontologies have been part of the scientific discourse for a very long time. Already Aristotle has philosophized about it and referred to this term as the primary attributes that belong together due their nature [GOS09]. In 1993, Gruber defined an ontology as an *"explicit specification of a conceptualization"* [Gru93]. According to Guarino et al. [Gua98], ontologies are used in the area of computer science for, among other things, the conceptualization of information systems or explicit knowledge. To understand the basic structure of those, one can imagine a modified form of a taxonomy, whose elements may contain further information, concrete instances, and most importantly, interconnections to other nodes within this taxonomy, independent from the location. In fact, such taxonomies form the first starting point of ontologies and describe their fundamental structure (cf. section 2.4.2). Taxonomies themselves are typically intended for classifications, and systematizations [NVM13]. An example could be a taxonomy of mobile computers. While on the zero level, the class (node) mobile computer is listed, on the next level, classes, such as laptop, smartphone, and wearable, are listed. Further specifications can be then made afterward (similar to the depiction in [VSJ+20] or cf. Figure 2.18). To illustrate

the usability and setup of a *taxonomy*, an example from previous research shall be given. In [SVG+20], *a taxonomy of existing taxonomies* in the big data domain was created, designed to facilitate a straightforward overview of existing approaches, the domain, and therefore increase accessibility. The taxonomy classifies all existing taxonomies provided in the literature. In doing so, according to their main scope, a hierarchical structure is built. Each node here addresses several of the found contributions. The specific mapping can be found in the respective contribution. Furthermore it is depicted in Figure 2.17.

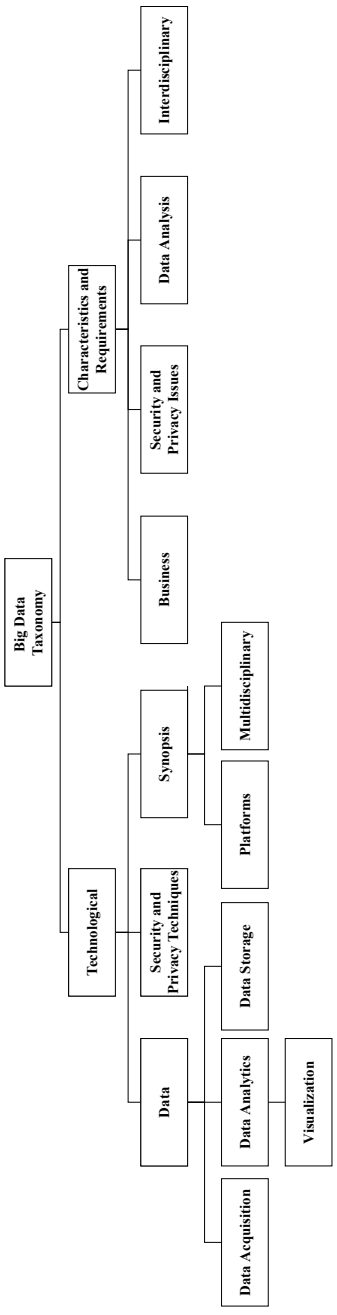


Figure 2.17: Exemplarily taxonomy of existing big data taxonomies [SVG+20]

Although this taxonomy is only implicitly used, the potential of structuring complex problems and domains, as well as the connection to available data (here the found taxonomies), is one of the main advantages of ontologies [RS12]. Based on this description, it becomes visible that taxonomies are typically forming the base for ontologies. With the rise of the *semantic web*, postulated by Tim Berners-Lee in [BHL+01], ontologies gained huge popularity in information system research due to these further advantages [RS12]. The semantic web describes an extension of the classic *World Wide Web* (WWW), in which machines can use and process data. For this intent, documents and information are put into a machine-readable form, for instance, by means of the Extensible Markup Language (XML), the Resource Description Framework (RDF), or the specially created Ontology Web Language (OWL). The OWL language's syntax that represents the ontology uses an RDF file with a XML-like structure. [Av09]. Here, the structure, relations, and relevant data can be mapped in the form of an instance. As an example, an instance of the class data storage, with respect to the previous examples, could be a particular taxonomy that has been found. In another context, this could also be a specific database solution, as they were shortly discussed in section 2.2.4. Hence, a comprehensive structure and mapping of related data are often required, which can be achieved using a single file.

In particular, the use of an ontology seems sensible due to the feasibility of hierarchizations, attributions, relationships, logical constructs, and reasonings. Because of these characteristics, a sustainable knowledge base can be created, enabling later reusability, extensibility, and adaptability that can be used standalone or in combination with a DSS, as highlighted by [RS12]. Despite the fact that most of the characteristics are also valid for various database types, many differences exist that delimit the usage of both databases and ontologies. Databases are commonly created from *scratch*, referring to the non-existence of previous concepts and approaches. In turn, during the creation of ontologies (cf. section 2.4.2) one of the initial steps covers the examination of already existing ontologies, providing a broader view on already investigated fields (cf. section 2.4.1). Furthermore, the focus of ontologies is rather put on the creation and presentation of knowledge. In combination with the aforementioned aspect of data persistence, sophisticated patterns are used instead of mechanics to prevent redundant data. Thus, while ontologies rather serve as a suitable solution to store, manage and share knowledge, databases are instead focusing on the persistence of the data [SBF15]. Notwithstanding that, the effort was already put into suitable methods to facilitate a transformation from one to another, and vice versa [MS18].

Even though the prominence of the research in the domain of ontologies and their interplay with information systems has changed, a multitude of different research articles is still published that discuss novel concepts, ideas, and techniques. For instance, in [ACD+19] a DSS is presented that harnesses the *Manufacturing Semantic Ontology*, built using the ontology tool Protégé, as the foundation of the knowledge base. Another article introduces a novel approach for the automated creation of ontologies, with a focus on

real estate [HZB+21]. An ontology covering comprehensive knowledge regarding tourist traceability systems, called *OntoTouTra*, is proposed in [MSF+21]. Through the additional use of big data analytics here, the ontology is automatically extended by new data in the form of classes and instances. Another research article that recently discussed the interplay between ontologies and big data was published by [SKL+21]. The authors propose that big data technologies and the *Smart Building Ontology* are used to save energy in smart cities. To obtain a thorough understanding of essential characteristics and methods established for the creation of ontologies, a short overview of existing types is given in the upcoming sub-section.

2.4.1 Types of Ontologies

For the construction and communication of ontologies, not only specifications regarding the used languages exist [Av09]. Furthermore, various types of ontologies have been established over the last decades. *Top-level* ontologies are commonly intended for a wide field of application areas to describe basic constructs [Gua98; NVB+13]. Therefore, these can also serve as a base for the construction of other ontologies. Famous ontologies that can be used as a foundation are, for instance, the *Suggested Upper Merged Ontology* (SUMO) [PNL02] or the *Descriptive Ontology for Linguistic and Cognitive Engineering* (DOLCE) [GGM+02]. The first mainly consists of 11 sections, covering various areas. Apart from basic distinctions of terms such as objects and processes as well as existing sub-hierarchies, also other detailed explanations of different areas are presented. This includes numerical operations, graph theoretic notions, and units of measure. All in all, even within the first version, *"roughly 1000 terms and 3700 statements involving those terms"* [PNL02] were included. A very small excerpt that highlights the general idea of this ontology is given within Figure 2.18. In here, a breakdown from the *entity* to the *science* domain is made. Starting with the first node, *Thing*, which typically initiates such a structure [GOS09], all further specializations are indicated by the relation `is-a`. These relations can be defined and assigned independently during the creation of ontologies. Unlike taxonomies, a relation can also be created between the individual nodes without creating a linear hierarchy. This is also shown in the two following examples.

Contrary to top-level ontologies, the *domain* and *task ontologies* are used for a specific scope (domain) or activity (task). To highlight the differences between these types, two examples are also given in Figure 2.18. Both of these are related to the area of mobile devices. Within the domain ontology, the class *device* functions as a generalization of a laptop, tablet, and smartphone, representing the conceptualization of mobile devices. While only the last two possess a specific Global System for Mobile Communications (GSM) module, they all use several other sensors. Compared to this, the task ontology focuses on particular activities, such as in the case of a telephone conversation. In here, a *Participant*, which is a *Human* is required for a *Conversation*, to realize this via telephone,

a suitable *Smartphone* and *Number* will be required. Application ontologies represent an intersection of both types and can be used for very complex structures in which individual entities of the domain can perform different tasks [Gua98]. In terms of the previously presented ontologies, an example for all types is shown in Figure 2.18. This includes parts of both the task as well as domain ontology. The class Smartphone forms the link between both approaches. Additionally, by using the class *SIM Card*, the connection between the needed *GSM Module* and the required Number is also defined. As a result, the entire ontology highlights the *application* of a smartphone device in terms of a regularly performed phone call.

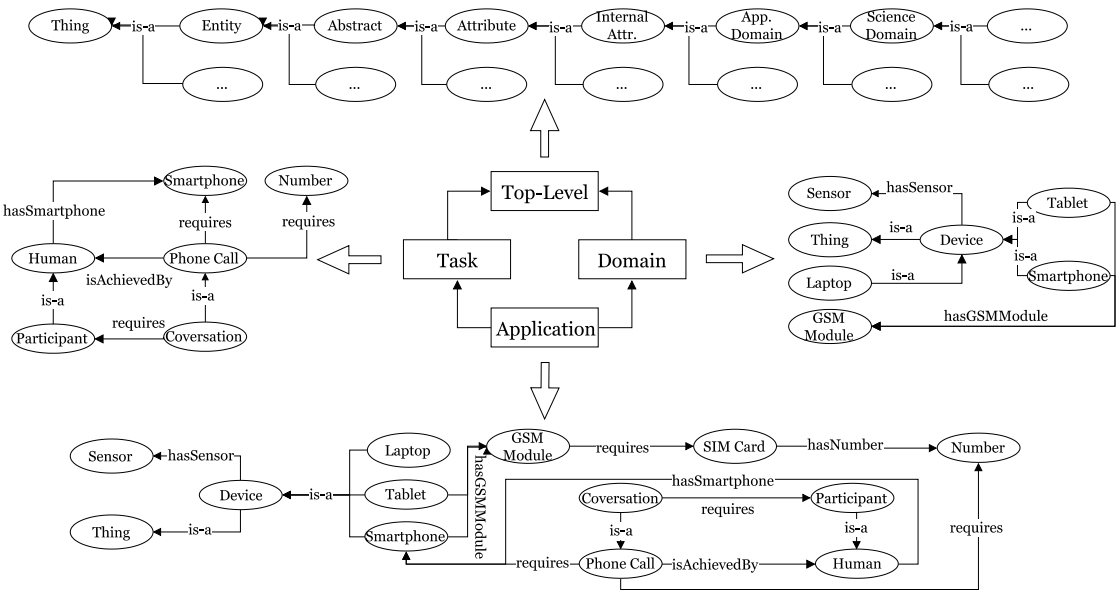


Figure 2.18: Types of ontologies [VSJ+20]

Another approach classifies ontologies into upper, mid, and lower (or domain) levels. While the first and the last are basically counterparts of the top-level and application types depicted in Figure 2.18, the mid-level ontology is described as a *bridge* between the other levels connecting abstract- with domain-specific concepts [Sem04]. According to Blomqvist et al. [BS05], further classification can be made into terminological, information, and knowledge modeling ontologies. These are, however, less widely used in the literature and therefore not considered in the further course of this work. For the identification of suitable ontologies, either for implementation or extension, numerous resources exist. Databases such as *BioPortal*, *Ontobee*, *OBO Foundry*, or *Ontology Lookup Service* provide a multitude of ontologies.

While most of the ontologies located in one of those databases are related to the medical or biological sector, only a small fraction deals with different concepts, such as software tools (*Software Ontology-SWO*) [MBL+14] or information artifacts (*Information Artifact Ontology-IAO*) [WB15]. In any case, as one can note, ontologies often repre-

sent just an excerpt of extensive knowledge bases and offer many opportunities to easily manage, extend, and implement them within complex scenarios.

2.4.2 Creation of Ontologies

As previously highlighted, the creation of ontologies strongly differs from the creation of databases [SBF15]. Today, a multitude of different methodologies, patterns, guidelines, and best practices exist that can be used to design and develop an ontology [CC05]. Similar to the creation of systems (cf. section 2.1), and big data systems in particular (cf. section 2.2.6), related principles and methods can be summarized by the term ontology engineering (OE). According to [IMM+13], it can be defined as a *"discipline that investigates the principles, methods and tools for creating and maintaining ontologies"*. Depending on the targeted application area, different approaches can be used [Gua98]. This is not only related to the overall process but also the single steps, through which an abidance of one approach seems insufficient [BS05]. Consequently, this exacerbates the structured creation procedure of an ontology. However, most of these follow a similar process flow, starting with a thorough planning phase that is often realized iteratively to overcome adaptations, extensions, and refinements easily.

As a starting point, a motivating scenario, an existing problem, or the overall planning is used (cf. examined methodologies [CC05]). Then, existing approaches are further checked and investigated in terms of their reusability, such as the DOLCE or SUMO. A first hierarchy is built, e.g. using the concept of taxonomies, by identifying further terms relevant to the targeted domain and the existing approaches. Depending on various factors, this can be achieved by the deduction from general to special concepts (top-down) or vice versa (bottom-up) [NM01]. Also, the middle-out method exists, where the *"basic concepts before the super and subordinate ones"* [Mik95] are formulated. Afterward, the relations and properties of the classes need to be further specified, defining the general nature of an ontology [NM01]. In the end, further evaluations are required that reveal potential shortcomings and needed refinements. An evaluation is performed to investigate the proposed solution's general applicability and validity. In case that refinements are required, further revisions will be conducted by reperforming the previous stages [NVB+13; NM01; UG96; SSS+01]. The general ontology engineering procedure is depicted in Figure 2.19 (based on [VSJ+20]).

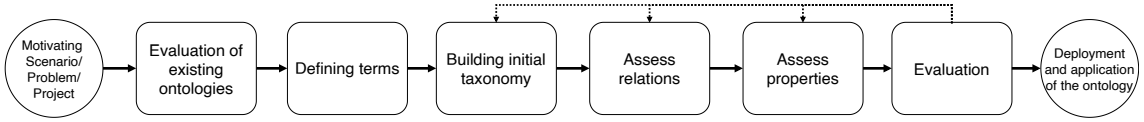


Figure 2.19: Generic ontology engineering procedure based on [VSJ+20]

As highlighted before, there is no standardized procedure, as in the case of the other domains, however, it is noteworthy to highlight that some researchers, such as [CC05],

subdivide the last mentioned steps into more detail, namely *conceptualization*, *formalization*, *integration*, *implementation*, and *evaluation*. For reasons of simplicity and because ontologies are only a small part of this work, used as a knowledge base, the depicted baseline procedure is followed.

In general, ontologies span a very wide field, with logic, representations, and as in any other field, guidelines and best practices exist. Due to the comprehensiveness that has resulted from decades of research, this section should only present a brief overview. Further, even more extensive information can be found in the following literature recommendations, which draws from [SS09; NM01; NVB+13; UG96; CC05].

3 Design of an End-to-End Procedure Supporting the Realization of Big Data Projects

As highlighted in detail in the previous chapters, the implementation of big data projects is not a trivial undertaking. It requires attention to a wide range of activities from different areas. To further approach the second hypothesis and answering of the research questions described in 1.2, this chapter deals with a potential *end-to-end* procedure that supports the realization of big data projects based on previously discussed BDE activities. Before, it was already indicated that not all relevant steps could be effortlessly implemented by a potential decision-maker. Due to the specificities of the domain of big data, various particularities during the comprehensive realization from *end-to-end* need to be taken into account. Most importantly at this point are the novel technologies and everything related to them, inter alia, targeting the *project planning*, *requirements engineering*, *SE*, *system modeling*, and their *respective deployment*. To better understand existing solutions and their applicability within the related steps, preliminary research in this direction is thoroughly investigated and described within the first section. Based on the made findings there and the previously given observations (cf. chapters 1 and 2), the general interaction with a computer-aided solution is proposed in the second section of this chapter.

In particular, a DSS capable of taking care of knowledge-intensive and cumbersome activities is introduced, together with the FRs and NFRs that such a system must fulfill. While the first predominantly focuses on the essential functionalities, the latter are derived from characteristics a DSS shall commonly satisfy. A exploitation table is created by an adhering comparison of the discussed contributions described in the first section with the requirements developed in the second. By observing this table, it becomes clear that specified approaches exist but cannot fully eliminate the open gap. For this reason, the inner activities of a potential system, as well as the interaction with a user, are further elaborated and discussed, supporting the full *intelligence-design-choice* decision-making paradigm and delivering auxiliary assistance for the system creation. Eventually, an end-to-end process is developed and represented by means of a *Business Process Model and Notation* (BPMN) model that depicts the interaction between user and system, which harness all of the aforementioned information. After that, all activities identified as non-trivial are bundled into components. These will afterward be discussed in the following chapter

4, in which the main design and development of the DSS framework is presented, forming the foundation of a prototypical implementation. The chapter itself consists primarily of research efforts that have already been published in previous contributions [VSP+19; VSB+20a; VSB+20b; VSI+22]. The articles that are discussed in the subsequent section were almost all found and discussed throughout all of the contributions described in section 1.5.

3.1 Related Research

During the presentation of the research background, numerous works were directly or indirectly presented, which support the realization of such projects. However, this was described rather selectively for certain areas in order to convey the necessary fundamentals. Today, there is an enormous number of contributions in the field of big data. A multitude of them were already thoroughly investigated, for the overarching goal of this work, in previous research activities (cf. section 1.5). In order to obtain an overview of relevant papers that contributed in either way to the goal of this thesis, they are presented successively in the following sub-sections, related to the project planning and design and development (cf. the BDE process - Figure 2.13). Their naming and description depend on the respective step within the BDE to which the achieved research outcome can be assigned. Furthermore, most of these were found through complex literature research procedures, referring to the methodologies mentioned in section 1.3. For more details about their finding and use, almost all contributions listed in Table 1.1 can be read.

3.1.1 Project Planning for Big Data

A thorough literature review was performed by Poleto et al. [PHC17] that investigates relevant elements for the application of big data in decision making. Related setups or systems, in general, require most of all components capable of handling, storing, and visualizing the data. The latter is mainly necessary for the interaction with the user. Similar to a DSS, the functionalities that are required here are not limited to simple inputs and outputs, instead, further search and analysis activities are to be expected, resulting in reoccurring feedback cycles. Finally, they propose a model for the decision-making process that also includes the use of a DSS. Despite the fact that this presented approach addresses the overall decision-making in general, not focusing on the realization of big data projects, the authors emphasize the relevance of incorporating and treating all of the data characteristics. In previous research, they similarly motivate an integrated view of DSSs, big data, and business intelligence to assist managers with problem-solving tasks as well as with the identification of novel opportunities. However, again they highlight the complexity of a related setup and integration in an organizational context. Thus, it requires a thorough understanding of the application domain and big data itself.

Initial steps that provide evidence for the sensibility of a big data technology application are worthy of investigation to prevent unnecessary planning activities because already existing solutions could already be sufficient, and no complex systems are required. In the work of Barham [BD20] a comprehensive approach is introduced that shall assess the readiness of a big data project. Originating from a similar discussion, as performed at the beginning of this work, the authors emphasize the high rates of big data project failures. Particularly this is the case for smart city environments. Hence, a hierarchical decision modeling MCDM approach, with which 18 different criteria can be examined, is proposed to identify the overall organization´s readiness for a big data project. Instead of using this one time only, a recurring application is suggested as long as the city status may fail. Since the focus is put exclusively on smart cities, usage in a broader scope is questionable. This also includes the applicability by non-skilled persons that might be capable to decide on any of the requested information. In general, this is the case for almost all of the discussed solutions. Although comprehensive information about the proposed solutions is almost always provided, required background information is not always listed.

Another work by [PLS16] seeks to provide a similar solution. The authors highlight that the implementation of big data could result in huge investments that can be dangerous for the related companies if the projects fail. This is presumably due to the equal consideration of the business opportunity missing here. In fact, most of the projects are instead focusing on the technological perspective, neglecting the initial planning to understand whether their realization is sensible or not. This leads, in turn, to similar project setups as in the case of business intelligence, with a structured procedure to reach predefined indicators for a project success. A similar problem description was also highlighted at the beginning of this work. The mixture of new paradigms and technologies requires an initial precheck, such as the *Big Data Complexity Framework* (BigDAF) that the authors postulate for the initial realization of the sensibility. It considers various data characteristics, their severity as well as the individual weighting of each of them within the final calculation. In particular, the weighting for volume is 60%, velocity 10%, and variety 30%, based on the fact that *"volume has higher importance to Big Data projects and consequently it was attributed a higher weight"* [PLS16]. Unfortunately, no further information are shared on how the weighting was created, similar to other aspects, such as the choice of the data characteristics or the final assessment value that determine the recommendation. The work itself comes close to the approach of Doug Laney without further highlighting it.

In a presentation about the *Information Economics, Big Data and the Art of the Possible with Analytics* in 2012 [Lan12], Doug Laney presented, inter alia, a possible way to identify the applicability of technology generations in big data projects. Using the three Vs, namely velocity, variety, and volume, and a layered model indicating the severity of each of in them a potential project, an assessment value, the so-called data magnitude index (DMI), can be determined. The higher the values are, the more relevant an application

of big data technologies is. Unfortunately, similar to the previous approaches, essential information about the creation is missing.

Other researchers further contemplate those data characteristics in their requirements definition to support the overall requirements engineering in big data. Norwali et al. [NAM16] describe in their research article a possible way to connect quality attributes, often found in NFRs, with data characteristics. In doing so, a specific format is chosen where, according to the authors, three elements are connected with each other. In their notation, the quality attribute and characteristic are connected via the multiply symbol $\times$. An example that addresses velocity and performance could be *"velocity $\times$ performance requirement : the system shall use a stream-processing engine with a latency of 0.5 – 2.0 seconds (e.g., Storm, S4, Spark or Samza) to process data in real-time between global earthquake sensors and the data center"* [NAM16] (see Table 1). In case one of the elements connected via the $\times$ cannot be described, NULL is used instead, indicating that further investigations in this regard might be required.

Comprehensive literature research regarding the RE was performed in [AM18]. Apart from essential activities of this discipline that need increasing attention, important challenges and existing approaches in the domain of big data were investigated, besides the already discussed approach, highlighted before [NAM16], further solutions are presented. Among other things, two comprehensive software tools are introduced, one of them as an extension of Microsoft Visio, focusing on the visualization of privacy requirements, and the other as part of a greater project. The verification tool, called D-VerT as part of the DICE project, was proposed by [BMR+16] and allowed the verification of unwanted configurations using annotated UML diagrams.

In another article provided by Arruda and Madhavji [AM19], *a tool for modeling quality requirements for big data applications*, called QualiDB, is introduced. By focusing on goal-oriented requirements, all relations between the overarching goal, the data characteristics, and related big data technologies on an *operationalization* level are formulated and modeled. In doing so, the requirements engineering of big data projects and related systems shall be assisted by the models that visualize dependencies and relations between all of the relevant information. Most of these articles were also brought into an overarching cumulative dissertation. Hence, for further requirements engineering details in this regard, the work by Darlan Florencio de Arruda can be used [Dar20].

3.1.2 Technology Selection and System Creation

Apart from the planning and related activities within the engineering of related systems, further design and development are required to conceptualize and create the system (cf. section 2.2.6). One of the most critical aspects in this regard is selecting specific big data technologies. In almost all found cases, only NFRs were considered. However, this applies not to the contribution by [MBB+20]. Here, *a big data processing tool naviga-*

tion diagram is introduced. According to them, the visual knowledge summary helps researchers and practitioners to reduce the number of available big data processing tools for their respective problems. The approach itself works similar to a decision tree. Based on essential functionalities, the first decision leads either into stream processing, machine learning, batch processing, SQL, or graph processing technologies. After that, essential non-functional aspects, such as the performance or latency of the solutions, are stepwise observed, leading to further branches or the recommended tool.

Another comprehensive approach is presented by [LFV16]. Here, the authors propose a layered reference framework that comes with a method for selecting big data technologies. While the first layer attempts to distinguish and classify technologies similar to the approaches described in section 2.2.4, the second supports the selection procedure. Within the description of each layer, distinct properties are highlighted, and exemplarily technologies are presented. The method itself, called *Strategy Time Analytics Data Technology* (S.T.A.D.T) the selection framework, thereby represents a multi-stepped procedure. Compared to the approach provided by [MBB+20] the first can be a structured process rather than a decision tree. A tactical plan is formulated starting with the strategy, decomposing a use case into storage, processing, and analytics. Afterward, each element is aligned to the layers and further process steps of selection framework, which describe the main building blocks in combination. In the adhering steps, the remaining requirements of the use case are investigated and integrated. By identifying the processing types, used methods, and the handled data, a technology selection is refined.

While these works tend to succeed by ordered processes and structures, numerical approaches are harnessed in other contributions as they are often used in DSS [RA15] (cf. section 2.3.2). Sachdeva et al. [SSK+16] describe in their work a multi-criteria fuzzy group decision analysis approach for the selection of an appropriate cloud solution in a big data project. In particular, the TOPSIS algorithm is harnessed to obtain the most suitable cloud platform, such as Google Cloud or Microsoft Azure. For the application of the algorithm, three criteria are observed: *security*, *business continuity*, and *e-governance*. However, they highlight that further criteria, such as implementation, cost, and confidentiality, are imaginable. Apart from the management or specific functionalities other relevant aspects are not considered here. Another approach related to the selection of cloud technologies in big data projects was thoroughly described in [BHA+17]. The authors use a fuzzy AHP and fuzzy TOPSIS algorithm to analyze ten different criteria, sorted into three overarching categories.

In Lněnička et al. [Lně15], an AHP for the selection of the most appropriate technologies is implemented. Besides some of the data characteristics, further important big data properties are formulated as requirements and used as *comparison criteria*. For instance, the computational *complexity*, *data security*, *maintainability*, or *cost* are viewed. Despite the fact that the authors thoroughly highlight the specifics and required background information for an application, only spare details about the aligned values for each technology

and each criterion are given. The same applies to the general setup of the approach.

In the paper provided Helmy et al. [HEE21] a fuzzy AHP for the selection of the most suitable *big data framework* was proposed. Based on the idea and criteria presented in [Lně15], the choice of the alternatives Spark, Hadoop, Flink, Storm, or Samza is supported using the fuzzy AHP. Although relevant insights are provided, especially in terms of the calculation, no details are shared on how the rating of the alternatives was made, similar to the work described before. Beyond that only NFRs are recognized, as they can be found, for instance, in the ISO/IEC 25010 [ISO17]. Certain functionalities provided by the final system are not further considered here.

Within the work by Alpoim et al. [AGP+19], another approach is proposed to identify big data integration tools, specifically *Talend* and *Informatica*. Instead of providing single functionalities, as in the case of big data tools, these rather represent entire platforms. Through the use of multiple criteria, covering functional and non-functional aspects in a compound manner, a weighting for each category is made. These weightings, in turn, were extracted by the authors from other literature. Afterward, presumably, an AHP is used to calculate the suitability of each of the alternatives. Notably, here, the defined characteristics were extracted from the work of [Lně15], thus making it an alternate version for the tool selection, or in this case, *data fabric tools*. Notwithstanding that, the construction of the solution and its application are hardly described. Essential information is hidden from the user, hampering the use and further extension.

Another approach that attempts to investigate entire big data platforms is introduced in the work of Haoud and Hali [EH22]. Within this work, the authors discussed the selection of *big data analytics platforms* on the base of different criteria emerging from recent literature. For the actual identification of the most suitable solution, an AHP is used, similar to the research mentioned in the above articles, which are also referenced here. However, no details about most of the information, the actual application, and the evaluation are shared. Similar to the other approaches, a potential real-world application and extension are limited.

The research article [EJQ17] discusses the relevancy of semantic technologies in the domain of big data, especially with a focus on ontologies. The authors stress the importance of sophisticated concepts to handle the complexity of this domain. In this regard, especially SME that may benefit from potential solutions capable of storing and displaying relevant information are addressed. Among other things, ontologies are described as sensible solutions in this context. For instance, when using those pre-defined structures and concepts, limited options for data integration can be given. However, even though valuable insights and motivational aspects are provided, no specific solution is proposed.

A concrete DSS for the selection of technologies is proposed in [FJJ+18]. Although not directly focusing on big data, it delivers complex insights about the process of choosing the most appropriate database technologies, including also big data-related solutions, such as the NoSQL database MongoDB. The overall setup is similar to a knowledge-driven DSS

as described before, containing an inference engine and comprehensive knowledge base amongst other components. The latter consist of a collection of decision models that denote grouped sets of *rules and facts*. For the inference engine, a MCDM approach is used that considers NFRs as they emerge from the related [ISO17]. The authors highlight that the prioritization of non-boolean criteria can be cumbersome and a complex task. Hence, a specific technique is used for the prioritization, called *MoSCoW*.

BDAs, as described above, are complex structures. They are usually composed not only of a large number of technologies but also have unique properties tailored to the application area. Therefore, many use case descriptions exist, providing detailed information and concrete architectures. These approaches are mostly so specific that conformist applicability is only possible to a minimal extent. However, reference architectures try to bundle these general concepts to a certain degree and counteract this problem partially with the same results. Often these also exhibit specifications, as they are only applicable for separate problem definitions. An example of this is the SOLID architecture, which focuses primarily on handling semantic data in real-time [MCA+15].

Notwithstanding that, also broader architecture templates exist that come very close to general concepts, highlighting critical components, categories, and functionalities. Two of these overly comprehensive approaches, as presented earlier, are the NBDRA [CG19b] as well as the architecture postulated by [PP15]. Based on existing architectures, both allow extensive insights and provide recommendations for possible instantiations. Especially in the case of substantial system approaches, which are to be integrated into complex structures, a consideration before and during the system conceptualization is beneficial. It should be mentioned here that although these can be used for almost anything due to their non-specific design, in-depth knowledge is again required to determine details.

However, since the setup of the perfect architecture is not the ultimate goal of this work but to give general technology recommendations and initial sandbox systems, such reference architectures will be considered rather implicitly. Nevertheless, the mentioned work can be used as a first starting point for users regarding further developments. Also, the provision of concrete architectures instead of individual technologies appears further meaningful. Therefore, solutions that are used for the immediate selection and provision of individual technologies and consider general structures and dependencies, make more sense here.

Even though, at the current moment, a vast number of publications in the domain of big data deliver promising insights into some of the targeted areas, only a few computer-supported solutions exist, especially dealing with the provisioning of related technologies. Against initial expectations, these were not found using classical keyword-based literature research approaches, such as [LJ06]. Instead, further forward- and backward-search procedures revealed them, as they are proposed by [WR02]. In particular, those are Apache BigTop, found in [OBA+17] and the DICE project, as indicated by [AM18]. According to the official project page of the former, BigTop *"is an Apache Foundation project for*

Infrastructure Engineers and Data Scientists looking for comprehensive packaging, testing, and configuration of the leading open source big data components" [Apa]. The offered components are packaged, delivered, and maintained by the community behind the project. According to its declarations, the scope of this solution mainly covers but is not exclusively limited to, big data technologies from the Hadoop ecosystem. For the actual deployment of the components, Docker is used, and for their internal configuration Puppet. While BigTop offers a wide range of functionalities, configurations, and testing capabilities, many technologies outside the Hadoop ecosystem are excluded here, presumably due to complexity and integration efforts.

In contrast to the Apache project, the DICE framework originated from an EU project funded under the Horizon 2020 program, which seeks *"to deliver a quality-driver DevOps toolchain for Big data applications that natively support these Big data technologies"* [CL18]. In doing so, a comprehensive plugin for the integrated development environment (IDE) Eclipse is provided that helps step-by-step with the implementation of data-intensive applications. Through the chained integration of UML diagram profiles and technology-specific peculiarities, comprehensive and detailed activities of BDE can be performed, including the planning, design and development, testing, and deployment. Here, the deployment of related technologies is realized using the configuration management tool Chef, which fulfills similar functionalities as Ansible and Puppet. Through the additional use of the cloud industry-standard Topology and Orchestration Specification for Cloud Application (TOSCA), cloud deployments of related prototypes and continuous delivery and testing are facilitated here. Unfortunately, the DICE project ended in 2018. Since then, no major extensions or updates have been performed. Hence, long-lasting usage cannot be recommended because changes in the big data ecosystem will no longer be considered.

Despite the fact that the testing plays a vital role in the successful construction of software systems and big data, in particular, this specific activity lies behind the main scope of this research. Here, the developed solution is rather used as an essential starting point for decision-makers to drive their big data projects. In case further integrations of the developed system are planned or specific strategies for skilled practitioners are worth considering, further contributions by Staegemann et al. can be recommended that, inter alia, discuss test-driven development and microservice concepts to create, test, and deploy BDAs [SVD+20; SVJ+20; SVP+21; SVS+21; SVT21]. Notwithstanding that, an overview of all previously discussed articles and a short description of those is depicted in Table 3.1.

Ref.	Brief Description of the Approach
[BD20]	Investigation of a big data project readiness using 18 different criteria and an MCDM approach.
[PLS16]	Identification of the BigDAF for the assessment of a big data project sensibility using weighted data characteristics.
[Lan12]	Propose the DMI and a layered framework for the identification of a big data project sensibility.
[NAM16]	Formulation of an approach to combine data characteristics and quality attributes to specify quality requirements in the domain of big data.
[AM18]	Comprehensive literature review, highlighting essential activities, novel approaches, and existing challenges for performing RE in big data projects.
[AM19]	Introduce the QualdiDB tool for assisting and visualizing the interconnection between big data tools as well as related requirements and data characteristics.
[MBB+20]	Description of the big data processing tool navigation diagram that can be harnessed to select related technologies through a visualized approach.
[LFV16]	Presentation of the S.T.A.D.T Selection Framework that denotes a multi-stepped procedure for the selection of the most suitable big data technologies.
[SSK+16]	Through the use of the fuzzy group-based MCDM method, here the TOPSIS algorithm, the selection of a potential cloud solution in a big data project is described.
[BHA+17]	Use of a fuzzy AHP and fuzzy TOPSIS algorithm for the selection of the most suitable cloud solution in big data projects.
[Lně15]	An approach for the selection of big data technologies using various (non-functional) requirements and the AHP.
[HEE21]	Use of the fuzzy AHP and different NFRs for the selection of the most suitable big data framework.
[AGP+19]	Here, different requirements are observed, and an AHP is used to select one out of three big data integration tools.
[EH22]	Another approach is discussed that uses an AHP and several aspects to identify big data platforms.

Table 3.1 continued from previous page

Ref.	Brief Description of the Approach
[EJQ17]	Description of a potential idea to sufficiently handle big data-related information through the use of ontologies.
[FJJ+18]	A concept for a DSS is introduced that helps with the selection of the most suitable database technologies using a specific MCDM method.
[CL18]	Introduction of the DICE project that covers a multitude of tools, accessible via an Eclipse-based plugin, for realizing big data projects.
[Apa]	Apache deployment tool, especially focused on big data technologies that can be used, among other things, for testing.

Table 3.1: Overview of the described research articles.

The presentation and discussion of the selected contributions in this sub-chapter highlighted that various approaches exist, addressing single parts and activities of BDE (as shown in section 2.2.6). By providing specific support for some of those, related projects and the engineering of corresponding systems can be facilitated and assisted to some extent. Notwithstanding that, as one may notice, no comprehensive end-to-end approach was found to help decision-makers realize their projects and the construction of related systems. Even though one may argue that only a low number of papers were presented, most of them were identified and discussed in the course of the published research articles for this work (cf. Table 1.1). For this reason, they not only show how individual scientific contributions support big data projects but also provide food for thought to enable structured decision support. This includes the overall identification of the sensibility of a big data technology application [PLS16], the formulation of specific requirements, or the modeling of those in combination with potential technologies and the addressed data.

Generally speaking, increased attention should be paid to the data characteristics, as described in section 2.2.2. Furthermore, existing projects should be observed and compared to the personally planned endeavor to receive a comprehensive view of these, either serving for initial ideation or the identification of domain-specific information, also in the context of the implementation area. Besides that, technology selection as the core activity for a related engineering procedure requires increased awareness.

Numerous approaches were identified that either focus on an argumentative procedure [LFV16; FV15; MBB+20] or make use of qualitative numerical approaches [Lně15; HEE21; FJJ+18; SSK+16]. However, almost all of them solely stick to NFRs. Hence, this shortcoming shall be solved in the intended solution. Therefore, a potential scenario is described in the following course that harnesses the discussed ideas in a generalized and structured way. Based on the complexity and the required knowledge for some of those activities, a computer-supported solution, in the form of a knowledge-based DSS

is discussed. As it seems to be a valuable addition to the scientific discourse [PHC17; FJJ+18].

3.2 Scenario-based Requirements Engineering

In order to approach the investigation of the second hypothesis and to find a possible answer to SRQ3, the following section will discuss a computer-based solution in the form of a DSS that enables the semi-automated selection and provision of big data technologies. The approach will be similar to the one already proposed in section 2.1.2. Proceeded from a scenario that was introduced there and generally *"tell a story of how various elements might interact under certain conditions"* [Pau95], a potential interaction with a system is described that supports the realization of a big data project and thus the application of the proposed process in section 2.2.6. Beyond that, a use case diagram from the UML is used to comprise and highlight the main activities, their relations, and the relevant stakeholders. Both solutions are suitable for the identification of initial requirements in information systems research [PR21; Sut03; OMG17]. In compliance with the relevant recommendations, a narrative description for a potential scenario may look like the following:

A potential decision-maker would like to conduct a big data project without having little to no knowledge about methods, technologies, architectures, or other background information. In order to get a first idea about big data and how it can be harnessed, a DSS shall be used that provides the necessary information. During the inital contact, the user is provided with the first hints for the interaction with the system. In particular, this concerns important input parameters that are related to the planned project. An extensive project planning step may be required by the user, which, in addition to an initial brainstorming, also includes a requirements analysis with an increased focus on the nature of big data. Specifically, this addresses the data and all necessary operations in dealing with it. In the case of a lack of ideas or the ability to define requirements concretely, existing project descriptions for big data projects can be analyzed and compared.

Once the user has gained a first understanding, a recommendation for a possible system should be provided originated the input information, allowing him to gain detailed knowledge, without extensively informing himself about the whole domain. Since the objective varies depending on the user's role and can range from the pure provision of information on technologies to be used, through their visual combination, to the provision of test environments, appropriate selection options should be available. Before these options can be implemented, the system must first check the given inputs in order to provide an initial assessment as to whether the use of big data technologies is sensible. If this is the case, an identification of the necessary system elements can be carried out under consideration of complex decision-making processes. The user then can use these further and thus conceptualize an architecture or automatically provide a system for testing purposes.

Based on this description and thus the interaction of potential stakeholders with the system, a model can be created in the next step, which, in turn, can form the basis for the requirements engineering [Sut03]. For this purpose, a use case diagram from the UML is used, which is able to represent the dependencies and interrelationships visually. All previously mentioned stakeholders were generalized and referred to as *users*. Regardless of the role or motivation (see Figure 1.1) of their system usage, they are in contact with the DSS. The *actor*, who is labeled as an expert, constitutes a skilled user that is not excluded from the overall user group. For instance, these could use such a system for improving related processes, further inform themselves, or create relevant and tailored architectures with relatively low effort.

The maintenance of the DSS is not explicitly considered here, only how a potential user can interact with it to receive decision support. The required activities and provided functionalities are depicted through ellipsis, typically referred to in use case diagrams as *use cases*. The connection can be made via directed or undirected edges. The direction of the reconnections also depends on potential conditions, which might be fulfilled, the extend (`<<ext.>>`) relation expands an existing use case by further ones. So-called *extension points* define when these could be triggered. Hence, the use case occurs not always. Contrary to this, include relations (`<<inc.>>`) always incorporate subsequent use cases and do not require further extension points. Necessary details about the notation can also be found again in [OMG17].

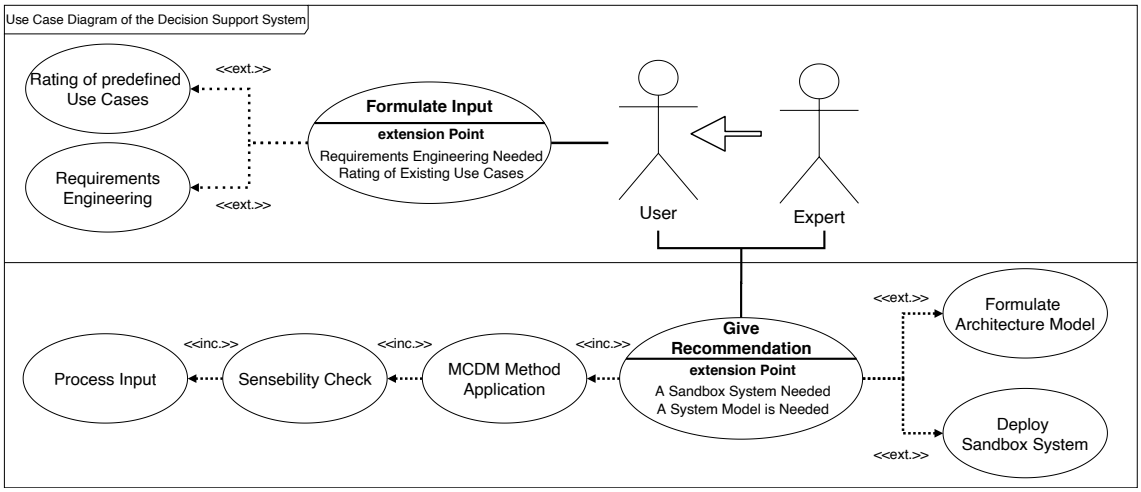


Figure 3.1: Developed use case diagram of a DSS and its interaction [VSB+20b]

In particular, stemmed from the observations made up to this point, the system's structure looks as follows. For every user interaction with the system, an input needs to be formulated that may require further investigation. Based on the scenario description, this comprises a thorough understanding of the project, covering, inter alia, the requirements engineering. However, if the user could not formalize those details, existing standardized use-cases should be investigated and compared in relation to the *planned*

project. Hence, a user may not only identify whether it resembles a real big data use case, furthermore, additional insights could be derived. Independent from the origin of the project specificities, the required information needs to be passed to the system, which can be realized using the GUI of the DSS.

The recommendation of a related system predominantly requires various big data technologies, each fulfilling different functionalities. Hence, other criteria need to be observed and checked through an MCDM approach. But before this can be realized, the overall endeavor should be evaluated and general sensemaking conducted. The latter commonly refers to *"structure the unknown so as to be able to act in it"* [Wei95, p. 4]. Typically it is applied when environments are getting to complex and additional help is required to provide clarification and to reduce confusion [Anc12; WSO14]. To start this and adhering activities, input information are required. If the user wants further assistance in the form of an architectural model or a blank system, related functionalities by the DSS are triggered. By observing each of the *use cases*, one can note essential functionalities the user and the system need to fulfill. As highlighted by the Object Management Group (OMG), in charge for the UML, these can be directly used *"for specification of the (external) requirements on a subject and for the specification of the functionality offered by a subject"* [OMG17, p. 640].

Originating from that, the following FRs were derived from the system. The *Technology Recommendation* (TR) indicates the identification of a single technology, while the overall proposition of their composition is described by a separate requirement (RA). Both are highlighted by the use cases *Give Recommendation* as well as the included *MCDM Method Application*. To conform to the general nature of the DSS, providing decision support and, thus, required knowledge, a *Technology Presentation* (TP) was added. The same reason justifies the addition of a *Use Case Comparison* (UC) requirement, with which the DSS should be able to gather essential details about existing (standard) use cases. Especially people without extensive knowledge in the big data domain shall be assisted with the selection of a suitable foundation of their project. An overview of all FRs a potential system may fulfill are listed in Table 3.2.

ID	Name	Description
RE	Requirements Engineering for Big Data Projects	The system defines inputs that require an initial requirements engineering through the user. In doing so, FRs and NFRs are necessary that need to be elicited. Indirectly, the required input fields provide sufficient information for the requirements the user may need to think about.
BC	Big Data Technology Application Check	The system is able to identify the overall sensibility of a big data technology application, in a combined way, before realizing a concrete decision support routine.

Table 3.2 continued from previous page

ID	Name	Description
UC	Use Case Comparison	The system contains an overview of existing use cases to either determine further specifics of the planned project or perform the ideation of the personal scenario.
TP	Technology Presentation	The system offers a presentation of condensed information of single big data technologies.
TR	Technology Recommendation	The system is able to identify single technologies for a targeted use case, based on the given input, through the observation of multiple criteria (e.g. MCDM).
RA	Recommendation for Architectural Compositions	The system is able to recommend and rank particular technology combinations for a planned project.
AM	Architecture Modeling	The system provides a solution to model or rather visualize BDAs that can be used for communication with further stakeholders and decision makers.
AD	Architecture Deployment	The system is able to deploy single and combined technologies in a targeted environment for rapid prototyping or testing to enable experiments with current setups and configurations.

Table 3.2: Derived FRs of the potential system derived from the use case diagram as proposed in [VSB+20b]

While the FRs were explicitly derived and developed, on the base of the use case diagram from Figure 3.1, further NFRs for the overall setup of the system are required to describe how the system may act (cf. section 2.1.2). To do this on an elaborated foundation, the characteristics of a typical DSS were investigated and suitable requirements derived (cf. Table 2.4).These are described in Table 3.3. Apart from FRs, also NFRs can be identified, similarly to the approach conducted by [DHO+13].

ID	Name	Description
RC	Role Concept	The system contains a role concept, with the different permissions and restrictions that can be managed, allowing users to manipulate resources in a regulated environment. Access management to critical tools, which should not be available for specific user groups, is regulated.

Table 3.3 continued from previous page

ID	Name	Description
MT	Multi-Tenant Support	The system is able to store and handle various users in one application.
KE	Knowledge Base Editor	The knowledge base can be read, altered, and extended using the system.
TE	Inner Tool Encapsulation	The system is able to solve semi and unstructured problems by providing an encapsulated provisioning of core functionalities. Each of them can be used in combination, standalone, and in a repetitive way.
ID	Information Distribution Among User Groups	The system is able to distribute information among individual users and user groups, allowing them to collaborate with each other.
KA	Knowledge Access	The system provides access to relevant information for the interaction, modification, and extension with the system, depending on the underlying role of the user.

Table 3.3: Functional requirements developed from basic DSS characteristics

Additionally, various standard NFRs were examined that are commonly used for software systems [ISO17]. Based on various aspirations of a good DSS and the different phases of decision making (cf. section 2.3), those were derived and transferred into requirements. In doing so, it is worth mentioning, at this point, that both the general characteristics of a DSS and the typical NFRs from ISO/IEC 25010 [ISO17] are not congruent but have a lot in common. For instance, the ISO deals with flexibility, graphical representation, and simplicity of use, similar to the points mentioned in section 2.1.2 and Table 2.4. In this context, web technologies are frequently highlighted that may greatly impact those. Using them means an application is not bound to an installation, location of use, or the device for accessing. This also applies to the GUI in charge of the overall appearance and interaction with the DSS. Due to this, those were combined into a single NFR for the DSS, named GUI and portability. This also applies for all of the other NFRs for the planned DSS. A list of all of them, including the ID for further comparisons, the name, and the description, is shown in Table 3.4.

ID	Name	Description
PE	Persistence of intermediate results	The system is capable to store and share intermediate results in the targeted environment, to prevent data loss and concurrently allow users to retrace former operations.
R	Reliability	The system provides the required FRs without any downtimes, referring to a constant availability. In case downtimes or errors occur, the system attempts to get back in a valid state to provide further operations.
GUI	Graphical User Interface	The system has an appealing and easy-to-use GUI that can be harnessed without any prior knowledge. It shall facilitate a consequent interaction with the user without requiring auxiliary installations. Documentation provides required guidelines for the use.
PO	Portability	The system can be used as a stand-alone tool either in one location or distributed throughout an organization. By using networking and web technologies, basic functionalities can be utilized without installing further tools.
E	Extendability	End users can further develop the system. Hence, the code should be written in an easy-to-understand programming language, sufficiently commented, and based on established best practices. The same applies to all components that are directly related to the system.
CS	Consistent Support	The system provides necessary documentation and assistance to support all groups of users, regardless of their level of knowledge.
CD	Customizable Decision Support	The user can fully control all steps during the decision-making. Most importantly, this includes the customization of single steps and parameters that result in the final decision support.
CI	Continuous Interaction	The system is focused on the decision support not on the decision making, requiring a constant interaction at each point in time while using it. Hence, input and related fields are named and described by each functionality.

Table 3.4: Non-functional requirements of the planned DSS

After having provided the individual FRs and NFRs for such a system as well as the interaction with it, the papers presented in section 3.1 are to be compared with it.

Concerning the FRs, only those from Table 3.2 were considered since all additional ones are instead related to typical DSS-related functionalities. However, as one may note by checking the requirement KE and KA, a comprehensive knowledge base (KB) is not directly covered within the derived FRs, even though these form an essential part of a DSS. Therefore, it was added to the considerations and comparisons. An overview of all found articles and their degree of fulfillment regarding the developed FRs is depicted in the simplified comparison located in Table 3.5. As soon as an FR is fulfilled, even if this is only slightly the case, either explicitly or implicitly, a • was used otherwise a ○.

Approach	**RE**	**BC**	**UC**	**TR**	**TP**	**RA**	**AM**	**AD**	**KB**
[BD20]	○	•	○	○	○	○	○	○	○
[PLS16]	○	•	○	○	○	○	○	○	○
[Lan12]	○	•	○	○	○	○	○	○	○
[NAM16]	•	○	○	○	○	○	○	○	○
[AM18]	•	○	○	○	○	○	○	○	○
[AM19]	•	○	○	○	○	○	○	○	○
[MBB+20]	○	○	○	•	○	○	○	○	○
[FV15]	○	○	○	•	○	○	○	○	○
[LFV16]	○	○	○	•	•	•	○	○	○
[SSK+16]	○	○	○	•	•	•	○	○	○
[BHA+17]	○	○	○	•	○	○	○	○	○
[Lně15]	○	○	○	•	○	○	○	○	○
[HEE21]	○	○	○	•	○	○	○	○	
[AGP+19]	○	○	○	•	○	○	○	○	○
[EH22]	○	○	○	•	○	○	○	○	○
[FJJ+18]	○	○	○	•	•	○	○	○	○
[PP15]	○	○	○	○	○	•	○	○	○
[CG19b]	○	○	○	○	○	•	○	○	○
[EJQ17]	○	○	○	○	○	○	○	○	•
[CL18]	○	○	○	○	○	○	○	•	○
[Apa]	○	○	○	○	○	○	○	•	○

Table 3.5: Comparison table that highlights the fulfillment of the FR by each approach

However, this applies not for those stated in Table 3.3 and the NFRs from Table 3.4. This is mainly resulting out of the absence from required information and testable software solutions provided by the artifacts of the individual contributions. As found out, currently, no solution exists that achieves all of the given requirements and, thus, provides comprehensive decision support along with the planning, creation, and deployment of a big data project and related architectures, in a computer-supported way. On the one hand, this is related to the form of the solution itself, where in most of the cases, no computer-supported approach was discussed. On the other hand, the presented contributions often focus only on a specific aspect and not on comprehensive end-to-end decision support, where system design and thus the selection of technologies play a significant role. Considering this, one could assume that the potential chaining of those would be a suitable idea in a timed workflow. This include, for instance, those which were proposed in [LFV16; PLS16; NAM16; CL18]. However, most of the previously described approaches heavily lack in the level of detail. This decreases not only the overall implementability but also the actual use. Especially for people with little to no knowledge in the domain of big data, a practical setup would not be possible. This is related to the unique characteristics of each of the presented solutions and the complexity that may result from a concatenation. Whenever input and output parameters or any other values in between may change, thorough revision and modification processes might be required. A solution covering an *end-to-end* procedure would provide the required functionalities in a structured way. In doing so, beginners who require thorough guidance may profit from this. The same applies to experts who may look for particular and distinct features a system may fulfill. In the following sub-section, an *end-to-end* procedure is proposed and meticulously described to overcome the existing paucity.

3.3 The Proposed End-To-End Procedure

After the basic requirements have been identified, it is further necessary to get a more in-depth understanding of the interaction of the relevant user(s) with the system. Even though use case diagrams present a suitable way to show the provided *use cases* and in which way actors are related to those, they do not reveal a specific workflow [OMG17, 639–641]. To highlight the communication between the potential *actors* and the planned system in more detail, the *Business Process Modeling and Notation* (BPMN) was used to create a *Business Process Diagram* (BPD). It is a sophisticated process modeling solution that is understandable amongst different user groups, ranging from business analysts to technical developers [OMG14, p. 1].

The entire process is depicted in Figure 3.2, dividing all *activities* into *user* and DSS–related ones, similar to the presentation of the use case diagram (cf. Figure 3.1). In order to be able to follow the subsequent procedure better, *annotations* are placed above or below the single activities. For simplicity and convenience reasons, the specific

symbol was not used; instead, the number was directly set. Concurrently, these are used in parentheses at the corresponding positions in the text. The process consists of two separate pools, one for the actors from the use case diagram, called *users*, and another for the intended *DSS*. The main reason for this is the potential logical and geographical distribution of both, as indicated by one of the NFRs of the system (cf. *PO* in Table 3.4). In general, the BPD utilizes almost all information, as they were provided in the previous chapters, including the described aspects of SE, project management, BDE, and DSSs. However, not all of them were modeled in the highest level of detail. Due to the focus on the technology selection and thus the general support of a project realization, activities that are not mandatory were not integrated. Furthermore, activities potentially covering further sub-processes, which are highly dependent on the environment and the future implementation, were covered as *collapsed sub-processes*. To increase the overall understanding, the official BPMN guideline can be recommended [OMG14].

The process of realizing a big data project from an end-to-end perspective, focusing on the engineering of related systems, can be started by a *user* with the formalization of the general project idea, at which the main aim and goal should be defined (1). The *user* within the process represents a single person or group, independent from the profession, level of knowledge, or job description. This first step serves more as a guideline to achieve a better overview of the main objectives and the later RE procedure. At the same time, it conforms to the typical way of planning a project (cf. section 2.1.2), creating a system in general (cf. section 2.1), and big data in particular (cf. section 2.2.6) as well as the interaction with a DSS from a management perspective in a decision-making process [TAL05]. Then, further concretizations need to be performed, where the *user* clarifies additional key partners, key activities, value propositions, and other details required for the specification of the big data project (2). In doing so, for the *user*, a multitude of existing guidelines and best practices could be harnessed, as they are thoroughly discussed in today's literature. At this point, the relevant activities commonly performed in the context of the respective organization of the *user* or the individual experience can also be conducted. Here, for instance, the clarification of the business model could be beneficial but not strictly required, especially if widely accepted approaches, such as the *business model canvas*, are considered [GH13].

Subsequently, when the user finishes the concretization of the project goal, either the requirements engineering procedure (3) or a use case comparison (4) should be triggered within the next step. The choice heavily relies on the required level of detail the current project has, as well as the knowledge the user brings with the application. In general, the complete procedure would involve the examination of existing use cases to obtain a thorough understanding with an adhering requirements engineering procedure (5a). However, this might not be required for every project to the same extent. In any case, getting a thorough understanding of the targeted big data environment and, thus, the specific requirements that need to be fulfilled is mandatory.

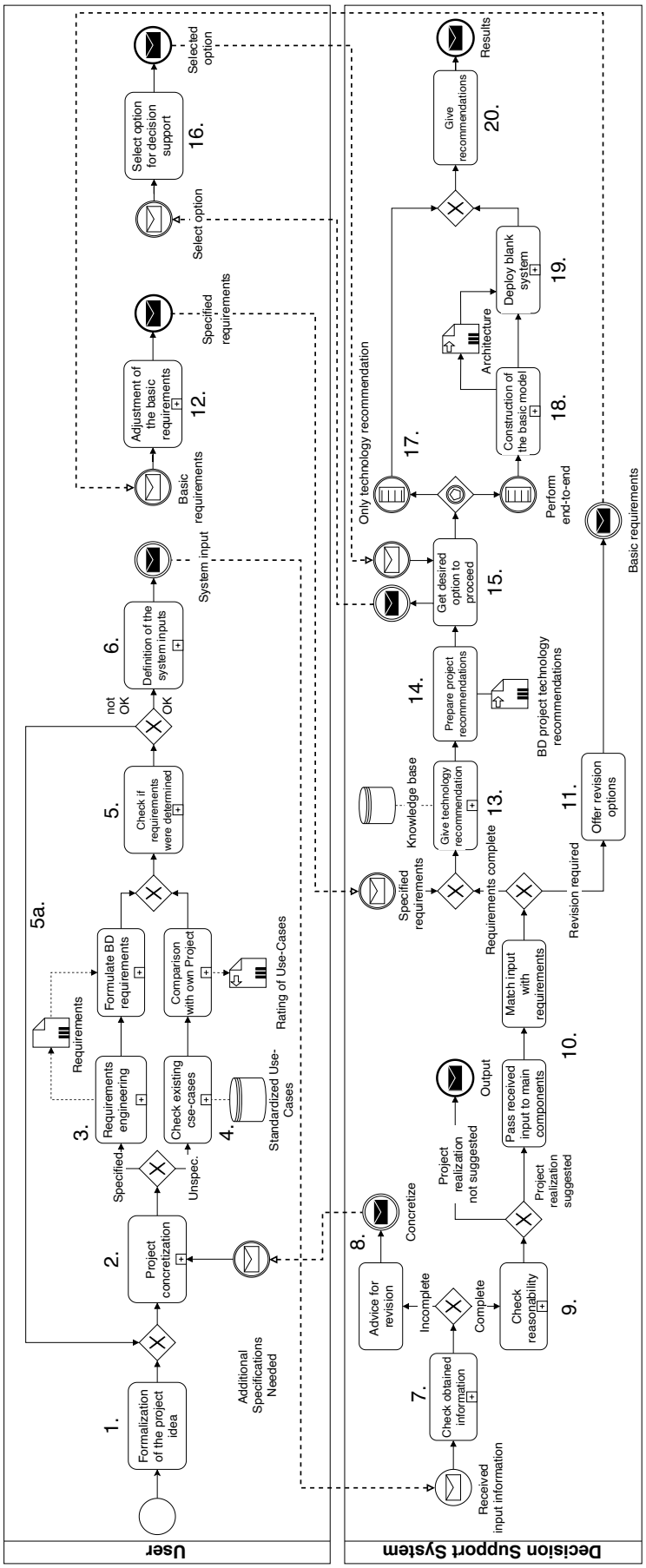


Figure 3.2: BPMN diagram of the end-to-end procedure

For the first case, if the user already understands the project and its specifics, the requirements engineering should be conducted (3). If no clear idea about the project and its specifics was elaborated in the second case, a comparison to existing SUCs should be performed. In particular, this would include the stepwise comparison to descriptions and requirements known from prior existing big data projects (4). For both cases (3 & 4), the result should be a clear understanding of the FRs and NFRs, the targeted system for the big data project must be capable of fulfilling (5). As described, if those are not defined, essential project planning steps need to be retaken (5a). Although it is generally recommended to do both, the requirements engineering and the use case comparison are as detailed as possible, and no all-encompassing documents or requirement collections are needed. Especially for untrained users, the comparison to existing use cases could be beneficial for obtaining initial insights into other big data endeavors. Notwithstanding that, as per defined on the base of the characteristics of a DSS, they naturally offer capabilities to refine given inputs during their use (cf. Table 3.4). This also applies to later stages, where, independent of the rigor of the performed steps, a revision can be performed to sharpen the technology selection, modeling, and deployment.

After the input information is determined, it can be submitted to the system (6). Depending on the offered interface, thorough checks might be required to examine the given input information regarding their applicability (7). This includes especially the resulting requirements from the requirements engineering procedure or the use case comparison. If the validated input information is insufficient, further steps need to be taken by the *user* (8). Otherwise, the process continues with the investigation of the general sensibility. Based on the determined information of the planned project, a potential suggestion can be given that highlights an isolated or combined application of big data-related technologies would make sense or not (9). If the latter applies, the process could be aborted (9a). Otherwise, it is continued with the transmission of all input information to the respective system components (e.g. the inference engine).

Then, the given input information shall be matched with the requirements needed as input information for the system processes (10). Depending on the level of automation, this can be done by the system itself or via the *user*. If further revisions are necessary (11), caused by system routine checks, or induced by the *user*, at least one additional feedback loop is needed, where the initially provided requirements need to be adjusted (12).

When all requirements are complete and usable, the main procedure regarding the decision support of relevant technology and their selection can be started (13). Within the entire BPD, presumably, this activity is one of the most complex ones, as already denoted in the previous use case diagram in Figure 3.1. Besides the already mentioned algorithm and user input information, technology-specific information also needs to be covered and used. Based on the number of existing technologies, their range of functions, the degree of fulfillment, and the interdependencies between them, here, for the potential inference engine of the DSS an additional knowledge base is needed that contains all of

the details (cf. section 2.3.1).

Subsequently, after the technology recommendations are determined, those need to be prepared for further steps. The system waits for the decision of the user (15), whether they only want to get the ascertained results or further activities to be realized. As soon as the user makes his decision (16), the system acts and either solely provides a technology recommendation (17) or continues with the end-to-end procedure, denoting the overall BDE process (cf. section 2.13) and use case diagram (cf. Figure 3.1). In particular, this comprises the creation of an architectural model (18) as well as the deployment of the selected technology recommendation, which is depicted here as a *blank system* (19). Finally, all results are shared with the user (20).

Although not explicitly highlighted, as one may note, some of the performed activities by the intended system may require an additional interaction with the *user*. This is the case, inter alia, by click events, collecting results, and other simple feedback tasks. For simplicity reasons, those were not further modeled. Instead, the implementation-specific interactions are the subject of a later prototypical implementation (see chapter 5).

3.4 Specification of System Components

As described in previous sections, today's information systems have, in many cases, a loosely coupled or layered architecture instead of a monolithic approach (cf. section 2.1. Especially for DSS, a layered architecture consisting of a presentation, the computation, and the persistence layer, or rather the knowledge base, is followed (cf. section 2.3.1). However, when a system is only visualized in those schemes, by setting up those, not always interaction points, potential dependencies, or other specifics can be identified. This is predominantly the case if some of the functionalities shall be used or provided in an isolated way. This includes not only the knowledge base that can be attributed to the persistence layer but also different activities from the designed end-to-end procedure, which can be either used optionally or in an isolated way.

Based on this description and the benefits resulting from such a component-based approach, the encapsulation of different functionalities, as they were denoted in Figure 3.2 tends to be highly aspirational. Hence, a component-based approach is targeted, with which an end-to-end process can be defined while offering the possibility to use each of the components also in an isolated way, conforming to the formulated NFRs and thus characteristics described in Table 2.4.

Generally speaking, there are various approaches to determine components [BO09]. If an already existing system is taken as a starting point, then already present components must be considered and the general range of application taken into account. New or changed components may not be disturbing [Zah03]. In contrast, systems that are newly created have more possibilities. Basically, it is considered with the identification of components that these exist, if possible detached from each other. The data to be

used, functions, as well as processes that such a system illustrates are always in focus. Decisively the components build thereby on the specified requirements to the system. The individual components group in each case always individual process steps and activities to fulfill a specific function. Accordingly, models, as they were pointed out before, both by the use case and the process diagram, serve as a suitable foundation [AKT+03; BO09; Tur03]. The latter, in particular, shows the general process flow, important tasks, and functionalities while allowing an insight into the data to be processed, e.g., through inputs and outputs. For green-field projects, as in the contribution at hand, top-down approaches are often used [BO09]. In doing so, the overarching system is decomposed into smaller elements that couple basic functionalities.

Based on this assumption, six essential components were identified that are mandatory for an end-to-end procedure (cf. Figure 3.2) to set up big data projects and thus implement the BDE workflow to create related systems (cf. Figure 2.13). Each component, with its planned role and functionalities as well as the further discussion throughout this work, is described in the following Table 3.6. Due to the complexity of each of them, the related sections of the performed research and the prototypical implementation are highlighted.

Name	**Functionality**	**Research Artifact**	**Proto. Impl.**
Technology Application Check Component	Based on the given input information, this component identifies the general sensibility of a big data project realization, referring to the combined use of big data technologies. It provides a general recommendation on whether further specifications appear to be reasonable or not. In doing so, it shall sensibilize potential decision-makers and allow them a quick-check for their planned endeavor.	4.1	5.4.1
Use Case Comparison Component	The processed input information by the user is used for acquiring knowledge about a possible implementation of big data technologies. If a user has no concrete idea about the project or its structuring, this component can be used in terms of necessary requirements. By means of a simple comparison of a short description of established use cases and their requested functionality, a rough understanding of the size and complexity of such an undertaking can be emphasized.	4.2	5.4.2

Table 3.6 continued from previous page

Name	Functionality	Research Artifact	Proto. Impl.
Multi-Criteria Decision-Making Component for a Technology Selection	The selection of a technology can depend on a variety of information that must be taken into account. In addition to simple binary criteria with regard to the functionalities the underlying system shall fulfill, these include NFRs. By means of a multi-criteria approach, the latter can be considered and compared with each other. As a result, different recommendations are provided in a ranked order that describes sensible constellations of single or multiple big data technologies.	4.4	5.4.3
Modeling Component	The visualization and representation of the components of a system architecture is a major challenge, especially for decision makers. Particularly in the context of big data, special aspects must be taken into account that increase the size and complexity of possible diagrams. With the help of this component, it should be possible to create and distribute big data diagrams in a simple manner in order to provide a visual overview of an imaginable implementation.	4.5	5.4.4
Automatic Deployment Component	The goal of discussing individual big data technologies and their combination is, in most cases, not only of a conceptual nature. In the aftermath of the decision support, the recommended technologies are often to be implemented and tested to get a better impression of their actual usability and functionality. With the help of this component, individual solutions are deployed automatically without the decision maker having to demonstrate a profound understanding of the installation.	4.6	5.4.5

Table 3.6 continued from previous page

Name	Functionality	Research Artifact	Proto. Impl.
Knowledge Base	All the necessary information for dealing with big data and its technologies must be stored in a comprehensive knowledge base. In addition to the pure functionality of being able to provide information in a targeted manner, modifications must also be permitted that allow sustainable and extensive use.	4.3	5.3.2

Table 3.6: Overview of the main components, their functionalities, relations, and further occurrences

Many of the components described in this table were identified as essential and non-trivial parts of the DSS, for which encapsulation is worthwhile. Noticeably, by the comparison of these with the described process is that, above all, the *Use Case Comparison* component comprises activities that were aligned to the user. However, since here supposedly complex comparisons are carried out under the same constraint of a large information density, it becomes logical to have a computer-supported solution in the form of a separate component that reveals information about different use cases. In this way, comparisons with the planned projects can be carried out both in advance and in retrospect. The lack of suitable approaches for each of those components, except for connectivity and presentation (cf. knowledge-base handler and technology presentation), led to the necessity to perform additional research for all of them before creating a DSS. In the next chapter, every component is discussed separately, culminating in an overarching framework presented in 4.7. This is then used for the prototypical implementation and use of the system.

3.5 Summary

This chapter described the general idea of an end-to-end procedure supporting the realization of big data projects. As a foundation for the construction of the intended artifact of this work, first, a general procedure for creating successful big data systems was required that may assist the realization of related projects. In doing so, the developed BDEP, consisting of established methodologies from adhering domains, was used (cf. 2.2.6 - Figure 2.13) together with insights presented in section 2.1. After that, essential activities were highlighted that could be covered and supported by a potential DSS. In conformance with best practices from the SE domain itself, a use case diagram was created, and FRs and NFRs were developed. These were afterward compared to existing solutions that could contribute in either way to the overarching goal. Even though many comprehensive ap-

proaches exist, none of those fulfills all of the derived FRs and NFRs shown in Tables 3.2 and 3.4. Hence, in conformance with the described BDEP, an end-to-end procedure was designed that constitutes a complete solution to overcome this paucity. Using the BPMN, a big picture of the entire process is given while simultaneously revealing the interplay between a user and the system. It became apparent that some of the functionalities are not strictly included in a workflow; thus, general components can be derived. Hence, contiguous activities were brought into single components that are worthwhile to be implemented to eventually form a suitable approach that is feasible to cover the presented process in a loosely coupled way that also facilitates an isolated use of those. Each of those elements is further investigated and described in the following chapter, ultimately forming a DSS framework.

4 A Decision Support System Framework for Big Data Projects

In the following chapter, all of the previously derived components are described, which eventually culminate in the main artifact of this work. All of these were thoroughly discussed in various published research articles, namely [VHB+16; VJT17; VPT18; VSP+20; VST21; VSJ+20; VSI+22; VST+20; VSS+22]. To obtain a thorough impression of the ideation, conceptualization, and evaluation, essential information about each of them is stated. Hence, every sub-section will briefly introduce the component's role, the related articles, the overall setup, use, and the conducted evaluation. Especially the latter represents one of the cornerstones of the applied eval patterns, which will be described in the overall evaluation chapter (cf. chapter 6), emerging from Sonnenberg and vom Brocke [SV12a].

Based on the aforementioned observations, for every component, a separate design science procedure was conducted (cf. section 1.3). However, at this point, one should note that only the main elements are conceptualized and described. Specific connectors, interfaces, or auxiliary modules, which are important for the implementation in targeted environments, used for maintenance, are omitted. Although some of the artifacts were programmed, the focus is rather put on the scientific research performed. Notwithstanding that, relevant implementation details are described during the prototype presentation and can also be found in the digital version of the documentation. Furthermore, for more detailed information, further insights can be found in the respective articles.

The overall flow of the presentation of the components conforms to the order of the BPMN process depicted before (cf. Figure 3.2) as well as the table of identified non-trivial elements (cf. Table 3.6). As described in section 4.1, the first component covers the general sensemaking functionality at which the reasonability of a big data project realization is assessed, including most of all the specific technologies. In doing so, further aspects are highlighted that deliver insights about a sufficient requirements engineering procedure. Afterward, SUCs, usable for comparing existing UCs, are described in 4.2, with which initial ideation, investigation, or further comparison and specification of potential big data projects can be conducted. Then, the required research for the sophisticated knowledge base component is addressed. In particular, an ontology-based approach is proposed in section 4.3 that delivers a comprehensive view of single technologies in the domain of big data, and their interplay and applicability. After the foundation for the

technology specification is described, an approach for selecting and recommending big data technologies and their combination is provided in section 4.4, using a two-stepped MCDM method. Other components can then use the given recommendations for modeling and deployment. Those are addressed in sections 4.5 and 4.6. Eventually, the shared results and highlighted aspects are brought together in section 4.7 to create the *Decision Support for Big Data Projects* (DECIDE) framework.

4.1 Technology Application Sensemaking for Big Data Projects

As it has been described extensively several times before, the identification of the general sensibility of an implementation of a big data project is a reasonable investment in terms of time and money, with which not only significant savings can be achieved but also general failures can be reduced or even prevented [PLS16; BD20]. Recent work addressing this issue is scarce and handles this task only in a limited way. For instance, although the work of [PLS16] has been discussed, which provides a helpful approach to calculate an assessment value used for the identification of big data technology applications, there is a lack of detail in many places. In comparison to the specially created and, apparently parallelly published work [VHB+16], furthermore, no reference is made to existing approaches. Specifically, this refers to the DMI created by Doug Laney, which is intended to assess a big data technology application solely based on the processed data. In [VHB+16] this approach and its general applicability were discussed in detail. The most important findings are summarized in this sub-section here.

As already found out before (cf. section 2.2.2), Doug Laney is known as the originator of the three core characteristics of big data, which are known today under the acronym of the 3Vs [Lan01b]. In his presentation about the *"Information Economics, Big Data and the Art of the Possible with Analytics"* [Lan12], he further harnessed those with respect to the applicability of various technology generations in big data projects. Under the simultaneous consideration of the three core characteristics, velocity, variety, and volume, and their severity, the DMI can be calculated. Using a layer-based framework in the shape of a triangle, a potential decision-maker can perform a mapping of the characteristics. This is depicted in Figure 4.1.

Every layer contains a classification number that employs a corresponding region or limit for each characteristic of the data created or processed. Depending on the markedness of the underlying characteristics, these values can be between zero and three. The identification of technology generations occurs according to their sum, which forms the DMI. When the result of the calculated value is situated between one and three, established technologies from the previous generation are sufficient to solve the addressed problem. If the value is over six, not even current big data technologies might be adequate. The

meaning of the DMI is shown in Table 4.1. However, as mentioned before, a critical observation should be placed towards the construction of the framework. No information about metrics nor the definition of any threshold values for the three layers were presented.

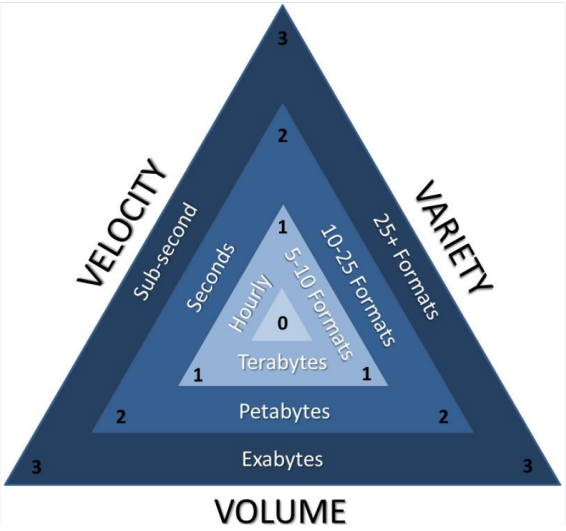


Figure 4.1: Doug Laney's framework to the DMI [Lan12]

Furthermore, it is not visible how the creation process was performed. This includes the identification of the suitable characteristics, as well as their classification. As a result of these findings, the applicability and reproducibility of the approach appear to be challenging. Additionally, this is reinforced by the time of the presentation. The framework was developed in the early stages of big data research (cf. section 2.2.1), which can also be noticed based on the Gartner Hype Cycle of 2012, where big data was just on the way to the *peak of inflated expectations* [Gar12].

Necessary Technology Generations	**DMI**
Technologies of the previous generation (e.g.,traditional database, data warehouse, etc.)	1-3
Technologies of the current generation (e.g.,In-Memory database, Hadoop, HBase, NoSQL, etc.)	4-6
Technologies of the future generation (e.g.,Quantum Computing)	7-9

Table 4.1: Distribution of the DMI according to Doug Laney [Lan12; VHB+16]

Beyond that, many aspects changed, including new characteristics and big data technologies. Even though the latter may not greatly affect the framework, new data characteristics, on the other hand, appeared to be worth looking at. Due to these problems, a direct implementation into the DMI was not sensible. Laney's framework was the first quantitative approach identified in the literature that does not require a huge investment

by a user. Only the data need to be somewhat identified and assessed. The limited information that is required for a user to use, led to further revisions and modifications.

In particular, further investigations were performed to check which of the newly discovered characteristics and their threshold values are suitable. This includes the examination of the original framework and its underlying characteristics as well as the used metric. To check, revise, and extend the existing approach, for instance, by additional characteristics and threshold values, thoroughly described use cases were analyzed, based on the methodology of Robert K. Yin [Yin09].

4.1.1 Examination of the DMI

Various big data projects should be reviewed to examine the existing DMI framework and its applicability, and the potential of further extensions. All information regarding the data characteristics should be similarly comprehensive and available. In doing so, the 51 cases were used that are thoroughly described in the third volume of the NIST Big Data Interoperability Framework called *Use Cases and General Requirements* [CG18]. Here, a very comprehensive template is used for the description of each use case that provides domain experts with a quick overview of the relevant aspects. In addition to some general information, the reports contain details about the processed data and their characteristics, particularly the Vs: *volume*, *velocity*, *variety*, and *variability*. All 51 cases were analyzed in terms of the DMI and thus the three core characteristics.

These focused primarily on the amount of the data but not on the size. We found out that 27 cases had an amount of data in the range of terabytes and only 12 of them in petabytes. According to the latter, only one commercial use case reached a size of exabytes [CG18]. In the context of velocity, specific numerical data could rarely be found. Only a few cases provided detailed information. Typically, velocity refers to the speed of data operations, like generation, storing, and processing [KNH+19; PSK17; DGL+13; CG19a; AA19]. Therefore, it is most uncommon to express it in such detail, as currently given by hours, seconds, and sub-seconds. Moreover, terms like batch processing, real-time, or streaming were used, as found out during a previous structured literature review and the mentioned use case analysis (cf. [VHB+16]). In 60 percent of the cases, real-time or streaming was addressed by the velocity. It should be noted that sometimes a distinction between generation and processing was made, similar as noted by the definition itself (cf. section 2.2.2). In this case, the faster expression was chosen. The analysis of the last core characteristic, variety, appeared to be more difficult. Generally, it addresses the diversity of the data. Therefore, expressions like structured, semi-structured, or unstructured are used [Erl16; GSS+16; GH15].

During the subsequent investigation of the cases, it was found out that the used terminology was inconsistent. Sometimes the typically mentioned expressions were used, and sometimes detailed data formats and types. We tried to interpret these statements and

derive information about the processed formats using the *data type* field of the template. Hence, it was found out that 24 percent of the cases processed approximately 5-10 data formats, 16 percent used around 10-25, and another 16 percent used vast quantities of formats. In doing so, it was implicitly noticed that in most cases, the ranges described the typically used structure types, e.g., unstructured data. However, distinctions exist, for instance, when only one format is used and, of all types, this is related to unstructured data, such as video files.

After analyzing the 3Vs, we compared the results found to Doug Laney's framework and his metric. We found out that not only the characteristics and the threshold values of his approach are applicable to identify the use of big data technologies. Furthermore, most of the characteristics' checked distributions were balanced, except the *velocity*.

To not only deliver an updated version regarding the original framework but also in comparison to existing approaches that are solely utilizing the 3Vs [PLS16], other characteristics were analyzed, as they were described in section 2.2.2. In particular, these are the previously described: *variability*, *volatility*, *veracity*, *value*, and *consistency* (cf. Table 2.2).

In terms of the variability, during the analysis, it was noticed that in certain cases, structural changes were explicitly addressed by this characteristic, also with a view on the applicability of specific methods and algorithms. For this reason, we assigned these occasionally appearing cases to their respective origin, which is, per definition, the volatility. In each of the related fields, similar terms were used to differentiate the severity. The amplitude was denoted using *low*, *medium*, and *high*. The others with a detailed description were interpreted considering these keywords. We found out that 70 percent of the cases dealt with the variability, and the remaining 30 percent were instead focusing on the volatility.

At its first glance, the veracity of the data appears as a suitable extension of the original DMI because possible technologies or analyzing methods, in the context of big data, are necessary to increase the overall reliability and trustworthiness. This could be done, among other things, by cleaning or preprocessing the data originating from different sources. For this reason, an inverse classification seems natural, in which the lowest veracity could indicate use of big data technologies. In the case of a banking application, which processes lots of structured data, the veracity could be very high because the sources should be credible. Nevertheless, this would lead to a negative effect on the overall assessment. Therefore, finding a suitable metric and threshold value for the extension of the existing approach appears to be challenging.

Furthermore, isolated handling, decoupled from the idea of the DMI, would not only unbalance the applicability. Additionally, this might appear as a hurdle for the early identification of a combined big data technology application for non-experts. The existing dependencies between veracity and other characteristics were also highlighted by [Gha21a]. Here, the link to value is also given, indicating the comprehensive dependencies of both of

these. Hence veracity and value tend to be unsuitable for further consideration in such a framework.

Big data is defined, inter alia, by the horizontal scalability and the parallelization [CG19a]. As a consequence, the data is not only persisted through multiple duplicated storages, also concurrent read and write operations occur. This leads, in contrast to traditional systems, to a possible forfeit of a strict consistency when the availability or partition tolerance will be guaranteed. In general, this reflects the consistency of Eric Brewer's CAP theorem [Bre00a]. Consequently, the general requirement for this kind of consistency could lead to a recommendation of already established technologies, where no distributed systems are needed. In contrast, an eventual consistency up to a general weak consistency appears more suitable concerning distributed systems and, therefore, the term big data. Although no clear indications and connections could be uncovered throughout the template and filled cases, evidence was found in other fields of the templates.

4.1.2 Development of a Big Data Project Application Framework

Based on the results of the structured literature review, the conducted use case analysis, and the given assertions, a framework for the identification of the sensibility of a big data technology application was constructed [VHB+16]. Fundamentally, it can be visualized as a hexagon, an amended and extended version of the previously described approach of the DMI. The essential properties of this framework, as it is depicted in Figure 4.2, are the layer-based structure and the two types of characteristics.

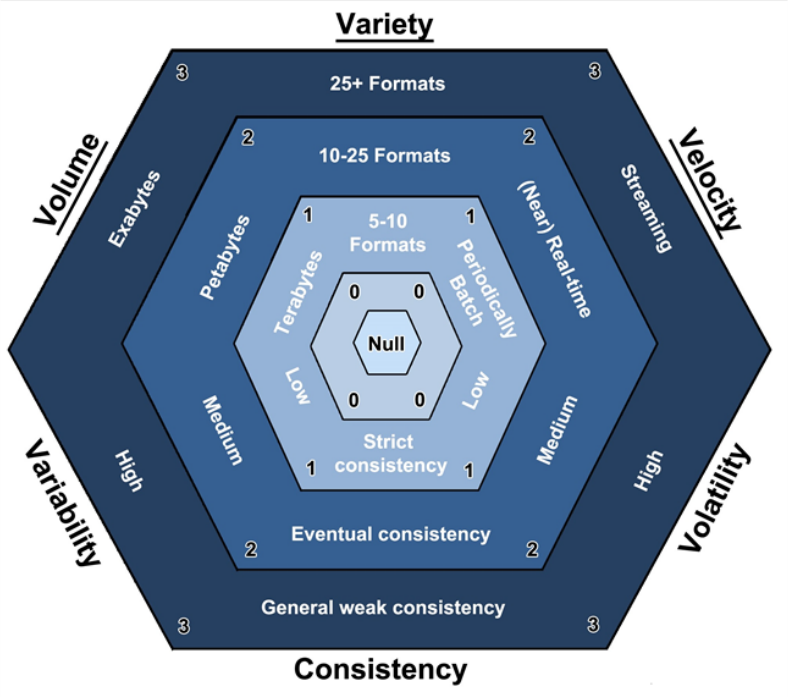


Figure 4.2: The proposed big data technology application framework [VHB+16]

On the one hand, the three underlined core characteristics can also be found in the 3V model and the DMI. Caused by the origin of the framework and the significance highlighted throughout the work, these are crucial and should always be determined by the user. Furthermore, the discovered and investigated volatility, variability, and consistency, from the use case analysis, were added as supplementary but not essential characteristics. However, it is recommended to use these to improve and strengthen the given results.

The layer-based structure is needed for appropriate representation of the application scenarios and the decision support in the context of the processed data. For this purpose, one possible metric was applied, which was derived from the results of the use case analysis and the subsequent discussion. Every layer represents a specific markedness of the various characteristics and contains a classification number. The higher these numbers are, the more sensible big data technologies are for this part of the application scenario. Nearly all encountered forms of the 3Vs were retained for parts from Laney's approach. Despite the initial doubt, his classification has proved to be suitable. At this point, only the time-based interval of the velocity was changed due to the differences in the general understanding of this characteristic and the given severities. The expressions of the additional characteristics volatility and variability were derived from the examination results, in particular low, medium, and high. Only those of the consistency were inferred by deduction.

As far as already described, Laney used the DMI to determine the suitability of certain technology generations with the help of the classification numbers from the core characteristics and their sum. Therefore, a specific classification was created, where the summed-up numbers can be sorted (see Table 4.1). While the simple sum of the values was sufficient, in the updated framework, the additional characteristics of variability, volatility, and consistency are optional. When the user of the framework cannot determine one of these in the input information, the classification number should not affect any result. Therefore, the additional NULL layer was introduced. Considering this, the arithmetic mean was used to calculate the assessment value, as shown in Figure 4.3.

$$\text{Assessment Value } = f(x) = \frac{\sum_{k=1}^{n} x_k}{n} \text{ with } \begin{cases} x = \text{assignment of the characteristics} \\ n = \text{length of the vector} \\ x_k = k_{th} \text{ entry of the given vector} \end{cases}$$

Figure 4.3: Calculation of the assessment value [VHB+16]

Nevertheless, an equivalent segmentation for the assessment value was chosen as in the DMI, with an adjusted range through the corresponding re-use of the arithmetic mean. Similar to the DMI, a value higher or equal to 1.33 indicates a potential sensibility of a big data technology application solely based on the handled data. The categorization of this assessment value is depicted in Table 4.2.

General Decision Support based on the Value	Range f(x)
It is not recommended to use big data technologies.	$0 \leq f(x) < 1.33$
It is recommended to use big data technologies.	$1.33 \leq f(x) < 3$

Table 4.2: Categorization of the assessment value [VHB+16]

4.1.3 Stepwise Application of the Developed Framework

In order to be able to define the severities of each data characteristic and apply the proposed framework, some preparatory steps are required. In particular, this encompasses all actions that are typically needed in planning a big data project, namely the initial project ideation, concretization, and related RE procedure (cf. Figure 2.13 and Figure 3.2). While many different methods and techniques exist for the initial steps, which are equally applicable for IT and big data projects, the latter RE may require special attention [MMK15; NAM16]. The previously discussed approach, provided by Noorwali et al. [NAM16] appears to be a sensible addition to general RE techniques. A combination of quality (non-functional) requirements and big data characteristics was discussed in their work, as already described in section 3.1.1.

A particular mapping between both is realized using the symbol $\times$, which is typically used to calculate the cross product. However, it should be noted that not all quality requirements are always related to the data characteristics and vice versa. According to the authors, for this particular case, NULL assignments may exist, which can be traced back to missing, or insufficient quality attributes, and thus a shortfall within the requirements engineering process. A similar setup was independently created for the previously described framework, also harnessing a NULL assignment in case of missing information (cf. Figure 4.2). Even though, the full adoption of this approach tends to be sensible, it neglects crucial aspects of the basic functionalities of a potential system, which influence the overall identification of relevant technologies.

As generally recommended, a combination of FRs and NFRs appears to be suitable, resulting in a combination of data characteristics and compound requirements. Instead of focusing on single NFRs, the complete compound requirement is considered. Examples for those are described in Table 4.3, following the principles of [NAM16] and the delivered guidelines in section 2.1.2. With respect to the developed framework, only the core characteristics are mandatory. All other are supplementary and can be attributed with NULL.

Based on those considerations and findings, the planning of a big data project can be realized by creating an initial idea. Before performing the RE, it is prepared by a concretization. Similar to the steps conducted for this work, scenarios, and use case diagrams can then be used to obtain an understanding of the compound requirements. Those are used and extended by the given data characteristics from the framework.

Relevant Characteristics		Characteristic × Compound Requirement
Core	Volume	The system shall store petabytes of data, which need to be processed for the positioning of the particles.
	Variety	The system shall analyze the data, which are present in 15 different formats, for fingerprint identification.
	Velocity	The system shall analyze the sensor information in real-time processing speed.
Additional	Consistency	The distributed system shall adapt to a consistent state from a general weak consistency after a certain event.
	Variability	The system shall scale out during high changes in the data flow while collecting sensor information.
	Volatility	The system shall react to medium changes in the data format while identifying new patterns.

Table 4.3: Exemplarily identification of the requirements [VJT17]

If only one of them is mappable, this is sufficient, since only the highest expression should be selected and assigned. For instance, as mentioned before, the velocity can be influenced by both the acquisition of the data and its analysis. Thus, possible future adjustments, which might affect later suitability or certain technologies, are considered beforehand. In case any of the necessary core characteristics are missing during the assignment of the specific information, the user will have to revisit the RE process step and determine the missing requirements. Afterward, the actual calculation can take place. Here, the requirement with the highest value of data characteristic is used. The complete process is depicted in Figure 4.4.

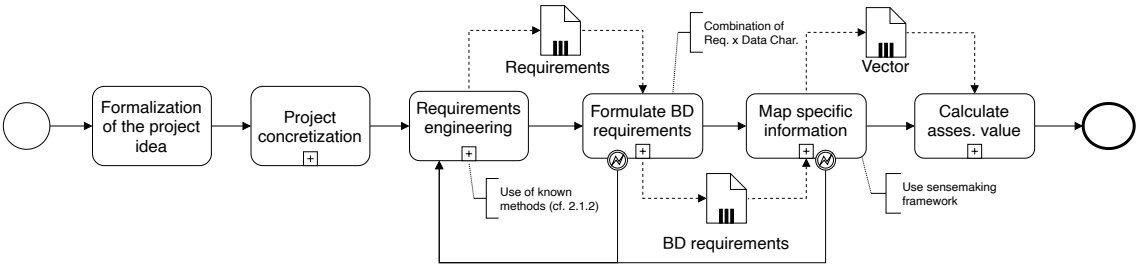


Figure 4.4: Instantiation of the big data project planning by the user, based on [VJT17]

4.1.4 Evaluation of the Stepwise Application Check Framework

A two-stepped evaluation was performed to examine the proposed solution's applicability. This includes the developed framework (i) for the identification of the sensibility as well

as the stepwise procedure, including the definition of the requirements and their mapping to the framework (ii). While for the first, two specific use cases were investigated, which were provided by Europe's biggest IT consulting company, with almost 700.000 employees. A real-world application scenario was used for the latter, performed at one of the biggest European business-to-business trade companies. The project's main idea was to analyze large quantities of clickstream data connected with user-specific information and purchase history to ensure fast adaptations of the complex category structure and give the user specific product recommendations. Further information related to this particular case can also be found in a separate publication, see [VSJ+17].

The evaluation for the initial framework (i) took place as follows. First, the processed data of the already implemented application scenario were identified in retrospective. After that, an appropriate mapping of these to the various expressions of the framework was conducted. The next step was to calculate the recommendation, for which the formula of the assessment value from Figure 4.3 and the classification from Table 4.2 technologies of the application scenarios were compared to the extracted information about the decision support from the framework use. In both cases, the processed data was assigned to the proposed framework. Therefore, the results of both cases look as shown in Figure 4.5. As one can quickly notice, there is a big difference between the individual expression, especially in terms of the spanned area.

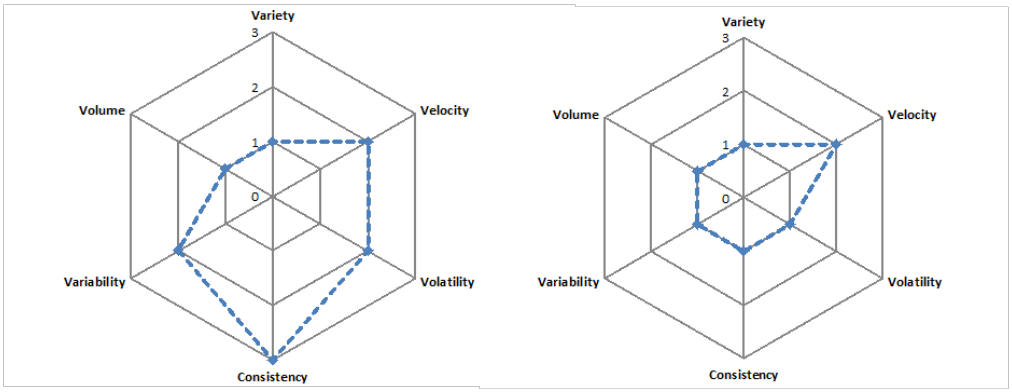


Figure 4.5: Evaluation results of an initial application check [VHB+16]

The results of the two cases were for the first 1.83 and the second only 1.16. Thus, the decision support of the first case indicates that a general usage of big data technologies is appropriate here, whereas this applies not to the last. However, following initial assumptions, this does not mean that certain big data technologies are not applicable for the scenario. Considering the higher classification number of velocity, specific technologies, which have a beneficial influence, should be deliberated. In fact we found out that various of those technologies were used in the application scenarios, especially in the first. While this case revealed a combination of multiple big data technologies, like Hadoop, Horton-

works, HBase, and HDFS, the second used only the SAP High-Performance Analytic Appliance (HANA). The main benefit of SAP HANA, or in general in-memory technologies, is the capability to reach a much higher processing speed [LLV+11; Pla12]. In connection with the higher classification number of velocity, this indicates the emphasized correlation between the characteristics and the applicability of certain big data technologies, even if the boundary was not exceeded. However, this cannot be fully demonstrated in the first application. Certain technologies used in the context of this refer to various characteristics. More specifically, they have a beneficial influence in different ways. Therefore, it is much more the appropriate combination of technologies, for which the determination requires sophisticated methods that are later discussed. After successfully evaluating the framework, the presented BPMN workflow was followed to assess the stepwise procedure (ii). First, the initial project idea was formulated, then the requirements engineering procedure was carried out. Using the described framework, all requirements classified as relevant were summarized accordingly and converted into a Table as shown in 4.4.

Relevant Characteristics		**Value**	**Requirements**
Core	Volume	1	The system shall store terabytes of data, consisting of user-specific information, clickstream data, and already made purchases.
	Variety	2	The system shall store and analyze the semi-structured data for the product recommendation system, presented in more than 10 different data formats.
	Velocity	3	The clickstream data shall directly being streamed into the system.
Additional	Consistency	NULL	NULL
	Variability	2	The system shall scale out during irregular changes in the data flow (medium), which will occur during sales, events, and changes in the range of products.
	Volatility	1	The system shall be adaptable and extendable in terms of planned changes in the data structure (low), which will occur after adding new data sources and analyzing methods.

Table 4.4: Mapping of the requirements to the respective characteristics

More specifically, the first two columns provide information about the type of the affected characteristics. The third column contains the extracted values of the single-layer, and the fourth column lists the highest requirement of these specific data characteristics. The values determined from the requirements were visualized for better clarity, similar to the evaluation of the framework, as shown in Figure 4.6. After that, the assessment value was calculated. The result of 1.8 revealed that the suitability of a combined big data technology use in this potential project is given. In fact, big data technologies such as Sqoop, Hadoop, and Hive on Spark were used.

Although the structured implementation is not always possible in this form in every project and often also depends on the custom of the individual or the organization, at least an approximate adherence to such a procedure for a requirements determination seems reasonable. This includes, above all, the integration of the framework, in which the expressions of the data characteristics are checked either by the determination of requirements or through elaborated guesses. In any case, thorough planning, as described in BDE and SE, seems to be recommendable.

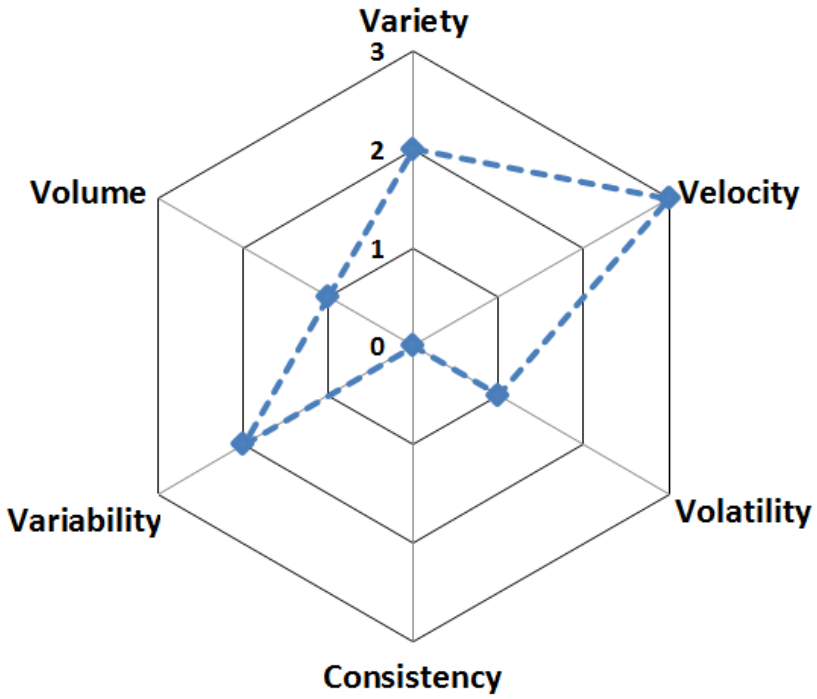


Figure 4.6: The illustrated mapping of the targeted layers

4.2 Standard Use Case Identification of Big Data Projects

If no detailed information about the planned project or a particular idea is available, the assessment of best practices can be a useful way to get started with a future project. Although many use case descriptions in the big data domain exist, these differ strongly in terms of their level of detail and overarching scope. For untrained persons, using them

as an orientation for project ideation or further concretization can be a complicated and time-consuming task. Hence, an intermediate level is required that offers generalized information that can easily be understood while covering specific details of relevant use cases. As found out by previous research, no such approach exist. Therefore, a potential solution to this problem is proposed in the following sub-chapter. The sub-chapter is primarily based on the two papers [VST+20; VSS+22]. In the upcoming sections, further details for the overall understanding of the conducted methodologies here, especially regarding the performed literature reviews, use case analysis, the built SUCs, and their evaluation, are presented. For a detailed description of each step, the referenced contributions should be read.

4.2.1 Identification of Existing Big Data Projects

In order to provide potential users of big data technologies with an initial idea of what existing projects and their implementation might look like, as well as what information and steps are essential for their own projects, existing descriptions shall be identified. For this purpose, comprehensive use case descriptions from business and academia are used, in which made decisions, the reasons for those, and their implementation are thoroughly described [Yin09].

Generally speaking, case studies tend to be a suitable solution to guide novice users with a particular problem. Either way, to provide maximum value, they must be written according to pre-defined standards [Kha17], since it was presumed that these case studies thoroughly describe the usage of big data technologies and the related processes in their context.

To identify relevant big data use cases that have been published between 2015 and 2021,two structured literature reviews were performed. Again, the methodologies according to Levy and Ellis [LJ06] as well as Webster and Watson [WR02] were used. Thus, a keyword-based search was conducted at the beginning, focusing on the terms *"case study"*, *"use case"* and *"case description"* in combination with *"big data"*. Afterward a forward-backward search was conducted. As a reliable (meta-) literature database that indexes relevant contributions from other databases as well, Scopus was chosen. However, this literature database covers only articles published in an academic context.

The Google search engine was used for further descriptions originating from industry, harnessing the same inputs. Further, additional inclusion and exclusion criteria were formulated and applied to refine the search results during the review process. Notably, while almost all of the criteria were applicable for publications from research and industry, the peer-reviewed inclusion criterion was only relevant for actual research articles. A document was accepted for further investigation if all of the inclusion criteria were fulfilled while none of the exclusion criteria was applied. A complete list of all criteria is given in Table 4.5.

Inclusion Criteria	Exclusion Criteria
The document focuses on the presentation of a big data use case.	The paper focuses mainly on introducing, developing, or evaluating new technologies.
The document must be written in English.	The paper does not provide any information about the data to be used.
The document was published between 2015 and 2021.	The paper does not provide any information about the previous data processing and analysis approach.
In the case of a research article, the paper must be peer-reviewed and published in either conference proceedings, a book, or a journal.	The paper does not highlight the main objective and expectations for the adoption of big data.
The papers mentions the used data sources.	The papers do not describe any requirements for the planned project.

Table 4.5: Literature review of existing use case descriptions in big data, based on [VST+20]

The literature review was performed two times, following the exact same procedure. The first, covering the completed years from 2015-2018, for the initial creation of the SUC, was outlined in the journal publication [VST+20]. The addition of the missing years from 2019-2021 is described in detail in [VSS+22]. An existing use case template was adopted to verify the comprehensiveness of each of the found-out use case descriptions. For this purpose, the template from the previously addressed use case collection of the NIST was utilized [CG18]. This extensive template consists of eight categories with 57 big data project-related questions. Apart from the general project description and the situation before the project realization, the relevant big data characteristics, applied techniques, and other information is stated here. Not all of the template's fields were of major interest. Hence, modifications to the original version were performed. This includes, for instance, the last two parts and questions like *"do you foresee any potential risks from public or private open data projects?" or "under what conditions do you give people access to your data?"* [CG18].

In total, the material collection resulted in 48 different cases, 43 (1.-43.) out of them were found in [VST+20] for the years 2015-2018 and another five cases for 2019-2021 (44.-48.). An additional forward-backward search did not bring any further contributions for both iterations. A complete list of all of them, concerning their origin as well as an ID, used for the further alignment, is given in Table 4.6. While most of the related databases are directly addressed, *others* comprises the databases MDPI, Taylor and Francis, Gesellschaft fuer Informatik, ACM, IADIS Portal, Scitepress, and the use cases originating from company resources.

Database	Paper References
Science Direct	**1.**[YSH+18], **2.**[ACD+18], **3.**[HIE+18], **13.**[GMG+17], **19.**[YMR+17], **27.**[IS16], **29.**[MVM16]
IEEE Xplore	**4.**[RFP+18], **5.**[ZZW+18], **8.**[TAG+18], **9.**[OBO+17], **10.**[KE17], **12.**[SPM17], **14.**[LAM+17], **15.**[AGP17], **16.**[HRA16], **17.**[AAR+16], **18.**[WCK+17], **23.**[PA16], **24.**[SKC16], **25.**[DG16], **30.**[SSJ+16], **35.**[PGS+16], **37.**[AGM+15], **38.**[ASH+15], **39.**[PTJ+14], **46.**[IZ19]
Springer	**7.**[SCN+18], **11.**[SPC+18], **21.**[SPT+17], **22.**[AGH+16], **31.**[SVS+16], **32.**[KZV+16], **33.**[FLL+16], **44.**[JJL20], **45.**[WAS+20]
Other	**6.**[MB18], **20.**[EJA17], **26.**[ZWS+16], **28.**[Che16], **34.**[HVN16], **36.**[HCJ+15], **40.**[XSQ15], **41.**[CSK+17], **42.**[SS16b], **43.**[Kha17], **47.**[JPA+20], **48.**[GAC+20]

Table 4.6: Results of the performed literature reviews [VST+20; VSS+22]

4.2.2 Use Case Analysis

After identifying many potential use cases, these should be further analyzed, correlations found, and generalized. For this purpose, general descriptions provide inexperienced users with a high-level picture of potential use cases. By concurrently categorizing the corresponding contributions found in advance, further detailed information can be extracted, which in turn are useful for concrete project planning. The creation of the intermediate level, which is understood here as *standard use cases (SUCs)* in the area of big data, is now further described as follows.

The manual comparison of all use cases may result in a great effort. Especially in the beginning, when a solid foundation in the form of a SUC is required, a computer-supported solution appears to be beneficial. Because of that, and to increase the comprehensibility of this research, a more objective analytical approach was chosen. In particular, document clustering was identified as suitable here. For this particular approach, numerous methods were compared with each other in terms of their applicability, as they are intensively investigated in various contributions [SKK00; ZKF05]. In doing so, hierarchical clustering was chosen, as it avoids the need of starting parameters, specifying the strict number or size of the clusters [Kan11] while it concurrently *"provide[s] a view of the data at different levels of abstraction"* [ZKF05]. Particularly for the creation of the SUCs an agglomerative clustering approach was used that assigns each object to one cluster and merges them until a whole tree is formed. The first step requires the calculation of a proximity matrix between the objects. Following that, the two closest clusters with the lowest distance are merged, and the proximity matrix is updated for the new cluster. The procedure is repeated until only one cluster remains [TSK+19; Kan11; ZKF05].

The basic steps needed for the clustering are the definition of the feature set for all

use cases, the creation of the input matrix, the examination of the hierarchical clustering, the definition of the cluster structure, and the determination of the intercluster distance. After everything is defined, the clusters need to be reviewed and modified in case they are not correctly assigned. By the end, those will be defined as SUCs. To summarize all of the aforementioned steps described before, all related actions for the initial creation of the SUCs are depicted in the BPMN diagram in Figure 4.7. As one may note, a distinction between manual and computer-assisted activities is made, indicated by a *human* and *gear wheel* symbol.

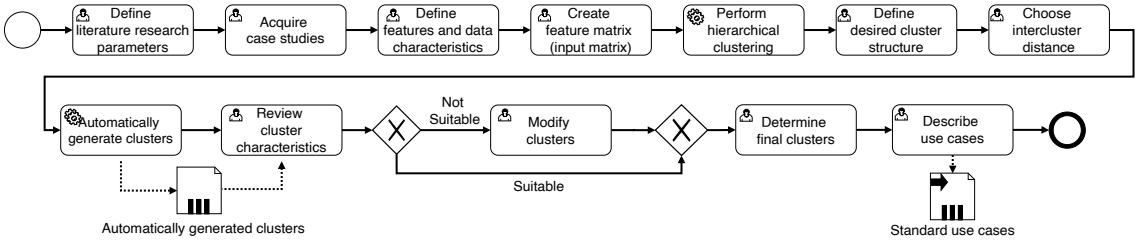


Figure 4.7: Procedure for the creation of the SUCs as a BPMN diagram.

With respect to this process, the collection of the material was already performed, and a suitable algorithm was already identified. Based on that, in the adhering step, the features and data characteristics are required, thoroughly describing each use case on a common foundation. In doing so, each of the cases was mapped to the modified template, describing the current situation (e.g., represented by the aim and data characteristics) as well as the obtained solution (used methods and technologies).

Although the template formed a promising starting point for the description of the feature matrix, it was not possible to use it as a direct input for the clustering algorithm. This is not only due to some unnecessary descriptive fields, such as *title*, *author*, or the rough *description* of the use case but also for needed information that can be manifoldly expressed like the variety of the data or the used algorithms. After an additional examination of the filled templates, 30 binary features were identified. Each of these and the number of occurrences can be found in the Appendix in Table A.2.

As one may note, some of the listed features were more frequently identified compared to the others. In descending order, this includes *Dynamic Data, Data Fusion*, *Unstructured Data*, *Heterogeneous Data*, *Statistical Calculations*, *Multiple Sources*, *Big Data Analysis*, *Real-time Data*, *Hadoop*, and *Batch Processing*. For the construction of the input matrix, the detailed mapping of the formulated feature set, with respect to the individual cases, was needed. This matrix can also be found in the Appendix, in Table A.3.

While one column represents one feature, each row stands for one use case. The IDs from the use cases and the features were used for better visualization. If a particular use case (row) fulfilled one of the formulated features (column), a filled dot (•) was noted, whereas for not related features, an empty dot (◦) was used. The created table was trans-

formed into a binary matrix, transposed, and used for the input for the actual algorithm. Eventually, the hierarchical tree structure (dendrogram) was created, as it can be found in Figure 4.8.

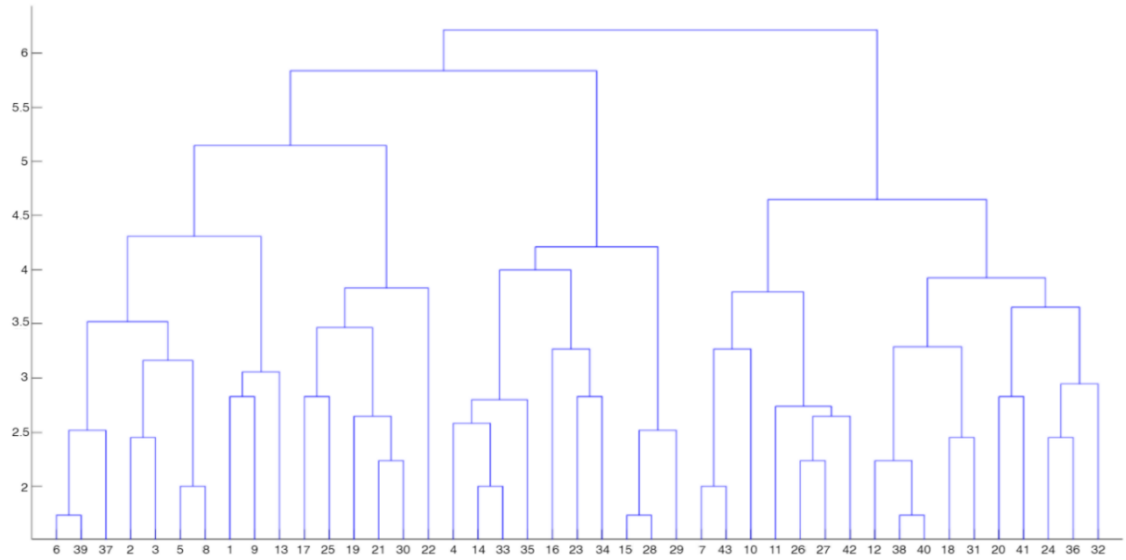


Figure 4.8: Dendrogram of the cluster analysis

The x-axis lists the ID of the use case (cf. Figure 4.8). In turn, the y-axis describes the distance between the various use cases and aggregated clusters. Within the figure, the elements and their interconnection are presented. The relation of individual nodes in the tree structure indicates a cluster. As mentioned, no strict number of contained elements is needed for the cluster definition. During the investigation and formulation of the features, huge differences between the cases were noticed in parts, which directly influenced this upper boundary definition. By having a low distance, too many clusters would have been created that differ only slightly from each other. Consequently, a decrease in the usability for later SUCs was expected, diminishing the general idea of such an approach.

With respect to the project planning, apart from the time-consuming planning steps needed beforehand, detailed knowledge about specific features would also be required to make further distinctions. For that reason, the inter-cluster distance and the number of agglomerated clusters were examined to understand at which point all cases were assignable to several overarching clusters. As one can note in the depicted diagram, only six clusters consisting of multiple cases are built at a distance of two, whereas 31 cases remain as one separate cluster. At the level of 3.5, only one case remained unassigned, and 13 clusters were formed in total. A distance of 4 resulted in seven distinct clusters, which comprise all of the 43 use cases. At a level of 4.5, only five agglomerated clusters exist. Considering the aforementioned disadvantage of too few cases the achieved seven clusters at a distance of 4 were chosen for the initial 43 use cases, ascertained in the first iteration [VST+20]. A better overview is given in Table 4.7.

Cluster No.	Paper References
1	**2.**[ACD+18], **3.**[HIE+18], **5.**[ZZW+18], **6.**[MB18], **8.**[TAG+18], **37.**[AGM+15], **39.**[PTJ+14]
2	**1.**[YSH+18], **9.**[OBO+17], 13.[GMG+17]
3	**17.**[AAR+16], **19.**[YMR+17], **21.**[SPT+17], **22.**[AGH+16], **25.**[DG16], **30.**[SSJ+16]
4	**4.**[RFP+18], **14.**[LAM+17], **16.**[HRA16], **23.**[PA16], **33.**[FLL+16], **34.**[HVN16], **35.**[PGS+16]
5	**15.**[AGP17],**28.**[Che16], **29.**[MVM16]
6	**7.**[SCN+18], **10.**[KE17], **11.**[SPC+18], **26.**[ZWS+16], **27.**[IS16], **42.**[SS16b], **43.**[Kha17]
7	**12.**[SPM17], **18.**[WCK+17], **20.**[EJA17], **24.**[SKC16], **31.**[SVS+16], **32.**[KZV+16], **36.**[HCJ+15], **38.**[ASH+15], **40.**[XSQ15], **41.**[CSK+17]

Table 4.7: Build clusters from the hierarchical clustering [VST+20]

After that, each of the clusters was qualitatively assessed, and modifications were manually performed. It was discovered that some of the use cases did not match properly with each other, even though they were assigned to one agglomerated cluster. Presumably, this can be traced back to the uniqueness of a big data project. Even though similarities between the features of a use case existed, differences were ascertained. Hence, modifications were performed on each of these, including insertions, deletions, and consolidations, to better highlight the data characteristics, used methods, and aim of the use case. Above all, this was required to ensure that key indicators, such as the distance, are used to create the SUCs and qualitative assessments are realized. For instance, it was noticed that the first identified groups of clusters three and four, containing cases no. 19, 21, 30, and 4, 23, 35, share similar interests.

Besides the general focus on smart cities, also the same characteristics are shared, except for one case (no. 23) that uses near-real-time processing instead of real-time-processing [PA16]. Hence, both of them were merged into one new cluster. All remaining cases of the initially calculated third cluster, focusing on sensor analysis, became the new second cluster.

Furthermore, the cases 16, 33, 34, and 40 appeared to be outliers, not only for the respective fourth cluster but also for the entire dataset. This does not represent an error in the qualitative analysis but rather how heterogeneous the individual use cases can be. For instance, in case no. 16 [HRA16], the main goal was to improve the query performance of a library information system. This was the only case that solely handled structured data within the given collection. Case no. 33 [FLL+16] exclusively used graphical processing of the data efficiently handle queries on multiple integrated bioinformatics databases. Those use cases were removed or assigned to another cluster to prevent misleading information.

Case no. 14 [LAM+17] proposes a smart clinical workflow that aims to increase the volume of medical data that can be processed. In doing so, health data from different sources are integrated and used to facilitate predictive therapy and improve the patient's well-being. Due to the similarities to the first cluster and the goal to generally improve the quality of the performed analysis, this case was assigned to the said cluster.

In cluster three, the case no. 17 [AAR+16], 22 [AGH+16] and 25 [DG16] revealed no real interconnection to the overall features. By comparing those, it was found that all of them aim to optimize existing processes using big data. Eventually, a new cluster was manually built (cf. Table A.1 – Cluster 9). The separate groups, which were identified within the initially calculated sixth and seventh clusters, were extracted and declared as a separate one. Hence, out of both clusters, two additional ones emerged (cf. Table A.1 – Cluster 5-8). An overview of all clusters is depicted in the Appendix in Table A.1.

There, also the use case descriptions from the second iteration are already recognized that was described in [VSS+22]. The contributions, from the years 2019-2021, namely [JJL20; WAS+20; JPA+20; IZ19; GAC+20], were investigated and manually assigned to the most suitable clusters. By having those, a potential stakeholder willing to perform a big data project may collect thorough insights regarding feature descriptions and further references. However, the overall applicability could be hindered due to the comprehensiveness and the level of knowledge that might be required to understand each of them. This is especially the case when additional comparisons of the clusters shall be conducted.

4.2.3 Definition of Standard Use Cases

As stated before, the sole consideration and possible comparisons of the build clusters could be a cumbersome task. Hence, out of each cluster, a SUC was formed to deliver understandable information, even for *non-experts* in this domain. To do so, a generally applicable description for each SUC was added. Furthermore, a thorough explanation that roughly comprises and describes the overarching goal of all contained use cases were added.

SUC 1 – Data Analysis Improvement By adopting big data technologies, an improvement in the quality of the data analysis is pursued. A significant step to achieving this aim is to make sense of massive amounts of unstructured data coming with high speed and exploit sophisticated methods, such as deep learning. Additionally, statistics and classification methods are often used to increase the quality of the analysis. The described characteristics of this general case and the used methods can be mapped to different cases, coming from healthcare, transportation, manufacturing areas, and social media. Details of the particular cases can be viewed in [ACD+18; HIE+18; ZZW+18; TAG+18; AGM+15; PTJ+14; MB18; JJL20; WAS+20].

SUC 2 – Batch Mode Sensor Data Analysis One of the reasons for harnessing big data technologies is to enable the processing of large amounts of (IoT) sensor data to

obtain new insights. Key factors in this use case are integrating different data sources, such as sensors and devices, and enabling the data exchange between users and applications. The data commonly does not exist in a structured format. Thus, processing unstructured data plays an important role. Real-time processing is not required, as the data is first gathered and processed in batch mode. To uncover different types of patterns, clustering approaches are used for the analysis. The visualization of the processed data is crucial to represent the findings. Based on those, strategies, for instance, to improve the user experience, resource allocation, process costs, and others, can be developed. Concrete specifications for this SUC are explained in [YSH+18; OBO+17; GMG+17].

SUC 3 – Smart City This category deals with the challenges of smart cities by involving various resources in real-time data analysis. The concept itself utilizes data from various devices, sensors, and human actors to improve the quality of life for citizens. For this purpose, structured, unstructured, transient, and permanent data can be used as analysis input. In order to turn a large amount of heterogeneous data into value, deep learning algorithms are used. In this case, a robust storage solution, such as a NoSQL database, should be used for massive amounts of differently structured data. Due to the nature of this domain, personal information has to be recognized, and privacy-preserving techniques are applied. All related cases are comprehensively described in [YMR+17; RFP+18; SPT+17; SSJ+16; PA16; PGS+16; JPA+20].

SUC 4 – Multi-Level Problems In this SUC, complex multi-level problems are stated, requiring thorough planning from different perspectives, covering the system and the data being processed. Organizations facing those problems are confronted, particularly, with the growing amount of data coming from various institutions, such as the healthcare sector. Apart from the required high reliability of the targeted solution and the ability to efficiently search, query, and store the data, privacy-preserving techniques also have to be considered. Moreover, processing unstructured data, such as handwritten documents or images, needs to be enabled. To analyze the data, different data mining approaches, which analyze stored data (e.g. on an HDFS) in batch mode, can be considered. This SUC originates from the following contributions [AGP17; Che16; MVM16].

SUC 5 – Expand Data Sourcing In this case, data coming from various resources need to be combined into one functioning system. As the considered data originates from different sources or instances, the structure and the data itself can be highly volatile. Due to this reason, not only sophisticated storage solutions for those various types of data (e.g., NoSQL), but also pre-processing techniques are needed. After the initial collection and cleaning, various statistical methods can be used. The data is usually processed in batch mode. Concrete details of all relevant use cases can be found in [SCN+18; IS16; SS16b; Kha17].

SUC 6 – Data Connection Adopting big data technologies in areas with widespread collections of information can improve decision-making by incorporating a larger information basis. As wrong decisions, especially in domains like healthcare, can have enormous

consequences, guaranteeing the correctness of the analyzed data is a significant step, necessitating extensive pre-processing. Depending on the application area, this can additionally require special processing steps like anonymization or classification. For the analysis, data mining techniques can be used, and efficient querying and searching over the data in real-time should be enabled. Further information is provided in [KE17; SPC+18; ZWS+16; GAC+20].

SUC 7 – Decision Support Real-time analytics on differently structured data are used in those use cases to facilitate decision support for data-driven problems. Previously unused data are converted into valuable information through basic statistics, classifications, and other analytical methods. For a better presentation of the obtained results, visualization techniques are highly important. This use case can be characterized by the phrase turn volume into value. Details can be observed in [WCK+17; SVS+16; ASH+15].

SUC 8 – High-Speed Analysis Within this use case, the input data comes in a structured and unstructured format and needs to be processed in (near-) real-time to ensure that all functionalities and results can be immediately provided. In addition, complex solutions are required to maintain, search, query, index, and analyze all data. Visualization techniques are paramount for an understandable representation of the results and the performed calculations. For particular insights, the following contributions can be used [SPM17; EJA17; SKC16; KZV+16; HCJ+15; CSK+17; IZ19].

SUC 9 – Process Optimization Big data technologies turned out to be an enabler for the general optimization of existing processes. Usually, the data is incoming with high velocity and needs to be processed in real-time. However, also batch-processing mode should be available either as a backup solution or for specific analytical tasks. In this case, both structured and unstructured data are considered. Clustering techniques support the identification of recommendations with which existing processes can be optimized. Various visualization techniques allow for a better presentation in an appealing way. Further details are described in [AAR+16; AGH+16; DG16].

4.2.4 Requirements Catalogue for Standard Big Data Use Cases

Especially for potential decision-makers that use those SUCs to plan their potential big data projects, more sophisticated information for their creation are needed. In particular, this refers to the FRs and NFRs, as they are essential for the design and development of related systems and thus the successful realization of big data projects (cf. section 2.2.6). The investigation of these was one of the main artifacts in [VST21]. In the context of the identification of a potential MCDM application for the selection of big data technologies, as it will be described in section 4.4, a structured literature review was performed. One more, well-known methods were applied in the keyword-based search procedure with an adhering forward and backward search, as they are described in [LJ06; WR02]. As a result, 23 publications were identified that deal with requirements either implicitly or explicitly.

The found NFRs are: *user interface (UI)*, *installation and maintenance effort (IM)*, *documentation and support (DO)*, *flexibility and scalability (FS)*, *fault tolerance (FT)*, *cost (C)*, *computational complexity (CC)*, *regulations (RE)*, *storage capacity (SC)*, *security (SY)*, *availability (AV)*, *sustainability (S)* and *reliability (R)*. A complete list of the articles as well as the addressed requirements is shown in Table 4.8. The description for each of the NFRs is stated in the Appendix in Table A.4.

Reference	**UI**	**DO**	**IM**	**FS**	**C**	**CC**	**RE**	**SC**	**SY**	**PM**	**AV**	**S**	**R**
[AGP+19]	●	●	●	●	●	●	○	○	●	○	○	○	○
[AM18]	○	○	○	●	○	○	○	●	●	●	●	○	●
[Bra19]	●	○	○	●	○	○	○	○	●	●	○	●	●
[CGD15]	○	○	○	○	○	○	●	○	●	●	○	○	●
[CLC+15]	○	○	○	●	○	●	○	●	○	●	○	○	○
[Con14]	○	○	○	○	○	○	○	○	○	○	●	○	○
[DGL+13]	○	○	○	●	○	○	○	●	●	○	●	●	●
[DLM14]	○	○	○	●	○	○	○	●	●	○	●	●	●
[Don17]	●	○	○	●	○	○	○	○	○	●	○	○	●
[FB19]	●	○	○	●	○	○	○	●	●	○	●	○	○
[GRS17]	○	○	○	●	○	○	○	●	○	○	○	○	○
[LFV16]	○	○	○	●	○	●	○	●	○	●	○	○	○
[Lně15]	●	●	●	●	●	●	●	●	●	●	●	●	○
[NSB+16]	○	○	○	●	○	○	○	○	●	○	○	○	○
[NAM16]	○	○	○	○	○	○	○	○	●	○	●	○	○
[OBA+17]	○	○	○	●	○	○	○	●	●	○	○	○	●
[PLH15]	○	○	○	○	●	○	○	○	●	○	○	○	○
[PZ14b]	○	○	●	○	●	●	○	●	○	●	○	○	○
[SS16a]	○	●	●	○	○	○	○	●	●	●	○	○	○
[SSC17]	○	○	○	●	○	○	○	●	○	●	○	○	○
[SZG+15]	○	○	○	●	○	○	○	●	●	○	○	○	○
[YHL+17]	●	○	○	●	●	●	○	●	●	●	○	○	●
[MBB+20]	●	●	●	●	○	●	○	○	●	●	●	●	●

Table 4.8: Overview of the found NFRs [VST21]

Most of the depicted requirements were already described in section 2.1.2 and reveal a high similarity to those stated by the ISO/IEC 25010 [ISO17], describing the quality requirements for software and systems. In fact, in most of the contributions, non-functional aspects were considered and discussed that rather focus on the overall effectiveness of a solution (cf. section 2.1.2). In turn, with respect to FRs that describe the comprehensive functionalities the system may fulfill, only the processing method (PM), in terms of a stream or batch processing, was found. It was assumed that most functionalities are rather unique for a project, similar to the characteristics and the technology combination. Consequentially, additional effort was put into the investigation of *general* FRs, as they could be found in various further research articles, use case descriptions, or technology documentations. Hence, an adhering search was performed to obtain additional requirements that are not only focusing on the *constraints of the service or functions of the system*, but rather the required functions themselves. Thus, after examining further research articles [KKA20; DVv+20; TS19; JBB14], which focus on technology-specific information, related documentation, wikis, and webpages of frequently used technologies, as well as the use cases harnessed in the context of the created SUC, a multitude of FRs were investigated that can be rather seen as basic functionalities. Eventually, the following FRs were identified:

automation acting (AA), *batch processing (BP)*, *cluster management (CM)*, *consistency preservation (CP)*, *data aggregation (AG)*, *data classification (CF)*, *data cleaning (CL)*, *data clustering (CU)*, *data formatting (F)*, *data mining algorithm support (DM)*, *data pipelining (P)*, *data selection (DS)*, *data streaming (ST)*, *data visualization (V)*, *event-data processing (EP)*, *machine learning (ML)*, *message handling (MH)*, *monitoring (MO)*, *near real time processing (NP)*, parallel processing (PP), *real time processing (RT)*, *recovery mechanics (RC)*, reporting (RP), resource management (RM), *store semi-structured data (SS)* (e.g., tagged data – xml), *store structured data (SD)* (e.g., table), *store unstructured data (SU)* (e.g., audio, video), *streaming processing (SP)* and *support scripting language (SL)*.

All of the identified FRs and NFRs are described in the Appendix in Table A.4. Supplementary insights from [VPT18; VSJ+20] were considered, as they are predominantly used in the adhering sub-chapter regarding the foundation of the targeted knowledge base. In particular, this refers to the overall structuring of the FRs on the basis of the individual process steps known from data mining processes, such as the CRISP-DM or KDD, which data-intensive projects are commonly following (cf. section 2.2.3 & Figure 2.13). To do so, five overarching categories were built to which those functionalities can be aligned, namely data ingestion (DI), data preparation (DP), data analysis (DA), data result delivery (DD) and system operation (SO) as it should be typically performed for the alignment of requirements, as highlighted in section 2.1.2 and later introduced during the construction of the knowledge base in section 4.3.1.

As one may note, FRs may either be requested by a specific SUC or not, leading to a binary decision. Compared to this, NFRs may be hard to formulate since they can conflict with each other in terms of shared resources or potential trade-offs. Instead of having a similar binary alignment, individual ratings might be required that highlight the severity of each of them [FJJ+18; Som16]. Due to this, to facilitate a similar differentiation and highlight the individual importance of each of those, a rating from 1 to 5 is given. Similar to the commonly known Likert scale [Boo12], it ranges from very low (1) to very high (5). A value in this range was defined for every single use case, considered for the creation of a SUC, individually highlighting the importance of each NFR. Whenever the requirements were neither considered nor implicitly or explicitly described, a score of one was given. A score of five was allocated if a requirement was explicitly addressed. The values in between were used if the NFR was implicitly highlighted (4), indicated by additional information (3), or only slight hints were recognized (2). Then, for each of the defined SUCs, the median value of all included use case descriptions was calculated. An overview of the importance of each NFR for each SUC and the required FRs is depicted in Table 4.9.

SUC	**Non-Functional Requirement (NFR)**														**Functional Requirement (FR)**				
	UI	**IM**	**FS**	**C**	**CC**	**RE**	**SC**	**SY**	**DS**	**FT**	**AV**	**S**	**R**	**DI**	**DP**	**DA**	**DD**	**SO**
1.	5	3	5	4	4	3	5	3	3	3	5	2	5	EP, MH, P, ST, SD, SS,SU	AG, CF, CL, CU,F	MH, NP, PP, RT, SP, DM, ML	RP, V	CM, MO, RC, RM, SL
2.	5	4	5	2	2	2	5	2	2	2	3	2	5	SD,SU	AG, CF, CL, U,F	BP, PP, DM, ML	RP, V	-
3.	5	5	5	4	3	4	5	4	3	4	5	3	5	P, DS, SD,SS,SU	CF, CL, CU, F	MH, NP, RT, ML	RP, V	CM, SL
4.	3	3	5	3	3	3	3	5	3	5	4	3	4	EP, P, ST, SU	-	BP, RT, DM, ML	-	MO, RC
5.	5	5	5	3	4	2	5	5	3	5	5	2	5	DS, SD, SS, SU	CL,F	BP, NP	RP,V	CM, MO, RC, RM, SL
6.	5	5	5	3	3	3	5	2	3	2	5	4	5	EP, P, DS, SD, SU	CF, CL, CU, F	DM, ML, BP	RP, V	AA, RC
7.	5	4	5	3	3	3	5	5	4	5	5	2	5	DS, SU, SD	CF, CL, CU,F	BP, RT, DM, ML	RP	-
8.	5	4	5	3	5	4	5	5	3	5	5	3	5	DS, ST, SD,SU	CF, CL, CU,F	PP, BP, RT, DM, ML	RP	-
9.	3	5	5	5	5	5	5	5	2	5	5	2	5	EP, MH, DS, SS, SU	-	BP, RT, DM	-	MO, RC

Table 4.9: An overview of all NFRs and FRs of the SUCs [VSS+22]

4.2.5 Evaluation of the Defined Standard Big Data Use Cases

Similar to the research foundation of the other components, DSR was carried out to. Consequently, as an essential step at the very end, the validity of the artifact needs to be verified [PTR+07; HMP+04]. Hence, for the proposed SUCs, a thorough evaluation was carried out in the first article [VST+20]. This comprises the procedure itself as well as the mapping of the SUC catalog. To assess the coverage with a practical orientation, an approach that is inspired by machine learning's division into training and test data is utilized. For this purpose, yet unused publications, emerging out of another iteration for the year 2019, were examined. Three additional use cases were found and harnessed for the evaluation. Since those were not involved in creating the SUCs, they function as the equivalent of a test data set. In doing so, another time, the procedure for collecting and verifying the material was performed, including the comprehensiveness check through the altered template.

The first case study used for the evaluation emerges from the area of online retail [AIS+19]. It provides an approach for a recommendation system that can be realized in an online store, requiring a user to sign up. Besides harnessing historical and transactional data, it also uses the customer's browsing history. Additionally, structured and unstructured data processing is required in real-time. All information was compared with the overall description of the SUCs discussed in section 4.2.3, the defined features highlighted in Table A.1, and the discussed requirements in Table 21. Hence, regarding those observations, and the key role of the recommender engine, within the data analysis, a mapping to the ninth cluster could be made. The second case study presents a system that uses real-time social media data for an analysis in the area of tourism [VPK+19]. The analysis comprises the main steps: data gathering, cleaning, storing, querying, filtering, and the visualization of the results. The data emerges from different social media sources, including Instagram, Flickr, Foursquare, and Twitter. Based on the data types, the content comes in an unstructured format in the form of posts, reviews, images, or videos.

By checking Table A.1, Table 4.9 and considering the case study's aim to involve real-time social media data in the analysis, this use case could be aligned to the eighth cluster, which targets real-time data analysis, incoming with high speed. The last use case description [MBR+19] deals with the area of smart transportation. Compared to the already existing approaches, which deal with single issues like congestion avoidance or environmental-friendly driving, the considered case study shows a system that proposes a solution to multiple problems. It aims to track vehicles, suggest optimal routes, and realize a smart parking concept, utilizing predominantly unstructured data from various sources like sensors, cars, or navigation systems. With regard to Table A.1 and 4.9, this example fits into the third general use case.

In conclusion, the successful categorization of the three cases used for the evaluation to one of the defined general use cases suggests that adequate coverage was achieved. The

degree of fragmentation, in turn, is based on the intended application scenario. While a more general approach might increase the coverage further, it offers no clear orientation in selecting potentially similar case studies. Vice versa, every case as its own category would effectively negate the idea of categorization. For that reason, the current number constitutes a trade-off that allows for a choice of relevant properties while still providing several example cases as a knowledge base for the aspired endeavor.

4.3 An Ontology for the Classification of Big Data Technologies

Knowledge regarding a specific domain is an essential factor for many projects, not only in connection with solutions, methods, and measures that can be used but also under which conditions this is done. This also applies to the description of big data, existing use cases, related technologies, and their use. Especially for the latter, it has already been shown beforehand that there are a large number of such technologies, and initial approaches exist to describe, classify (cf. section 2.2.4), and even select them (cf. 3.1.2). Nevertheless, these approaches are usually very limited concerning their further application and only allow a use and modification with extensive expenditures. Moreover, a computer-based utilization is permanently excluded due to the pure presentation of tables, facts, and generic algorithms (cf. Table 3.6 – no computer-supported solution regarding that functionality). A similar circumstance was ascertained for the previous identification of SUCs. The manual comparison of the information listed in the various table, such as the SUC description, their FRs, and NFRs, can be a sophisticated endeavor when selecting one. The managing of new and complex information was also discussed in an independently conducted research presented in section 3.1.2. In [EJQ17], the lack of a suitable approach to *"capture, manage and present the complex information"* is described. In this regard, the authors highlight the suitability of semantic technologies, especially an ontology-based concept. Although a thorough motivation is given, the authors miss delivering a particular solution.

Within this section, an ontology is to be described as a suitable knowledge base that focuses specifically on big data technologies and can thus provide extensive information on them. At the same time, all other domain-specific information that has been identified and developed in the context of this work should be included to develop a comprehensive knowledge base that the envisaged solution can use. This includes, inter alia, the investigated SUCs, their specific use case descriptions, FRs, NFRs, and their relation to all other elements. Hence, the content discussed here is based not only on the previously identified findings and decisions but primarily on the conference contributions [VPT18; VBT17], and the journal contribution [VSJ+20] that arose from it. While a basic version of the ontology was already presented in the journal article, more far-reaching changes had to be

made in the further course of the development of the artifact of this work. This pertains to those that arose during the extension of the knowledge base by new classes and those that resulted from the implementation and use of the other elements.

4.3.1 Preliminary Considerations of the Ontology Setup

The creation of an ontology is similarly performed to the development of requirements and systems, via specific engineering activities. OE, as discussed prior (cf. section 2.4.2), denotes various procedures that can be used to create ontologies. However, these are influenced by the aspired type (cf. Figure 2.18). Although a broad knowledge needs to be covered by the intended ontology, everything is related to the *domain* of big data. By performing a closer comparison to the existing types, accordingly, the type of a domain ontology was identified here. Due to the previous identification of many existing concepts in this regard, comprising technologies, SUCs, and requirements, the middle-out approach was chosen for the creation. Compared to the top-down and bottom-up approach [NM01], it focuses first on the definition of basic concepts, with an adhering identification of super and subordinate ones [Mik95]. In doing so, the previously introduced OE procedure was instantiated (cf. Figure 2.19), requiring at first the evaluation of existing ontologies. Here, the main concepts were defined as a foundation.

At first, famous top-level ontologies were investigated, including DOLCE and SUMO. Unfortunately, non of these contained relevant classes, properties, and relations. Due to this, various ontology databases were searched, such as BioPortal, Ontobee, or Ontology Lookup Service. In doing so, the SWO [MBL+14], IAO [Ceu12], the Ontology of Data Mining (OntoDM) [PSL08], and OntoDM-KDD [PSD13] were identified. After further investigations, it has been observed that these are already interconnected in parts. OntoDM(-KDD) describes different analytical steps of data mining processes, data types, and processing. The multi-component ontology extends both of the found mid-level ontologies. While the SWO describes tools and algorithms used in the area of bioinformatics, the IAO focuses more on general objects to serve as a bridging ontology. Terms, relations, and specific characteristics, which could be applied to the domain ontology, were extracted and used within the developed approach.

The building of the foundational taxonomy was started using the results of the previous sections. Most of all, existing categorizations and terms description were of major interest. Hence, potential categorizations, as already described in section 2.2.4, were incorporated here. Especially in terms of data science processes, such as the KDD and CRISP-DM, great synergies were identified in the context of big data projects in general but also with respect to the OntoDM [PSD13; PSL08]. Although these approaches differ in their respective field of application and level of detail, all of them contain a procedural sequence of how a data mining, data analytics, or data science problem could be solved stepwise. This includes the identification, preparation, and analysis of the data as

well as the visualization of individual results. In a project context where the engineering of related systems plays a dominant role, the operation of those could be considered as well. Generally speaking, in terms of an ontology with a focus on big data, unambiguous wording is required. The problem related to this was already addressed in section 2.2.4, revealing that even the term technology is often incorrectly treated.

Based on that, in conformance with the second step of the OE and the used middle-out approach, founding terms were determined, and further sub- and superordinate ones were formulated. All classes were determined, named, and described iteratively according to their definitions within the literature and the naming convention of currently existing ontologies. Besides the concept of a technology and its delimitation to other terms, such as paradigm and tool (cf. Table 2.3), also FRs, NFRs and SUCs were chosen as a foundation. While the aforementioned concepts were already specified, the bridging process step that offers potential for mapping individual technologies to not only each step but also different functions was not yet specified.

This encompasses the general steps of a data science process: data ingestion (DI), data preparation (DP), data analysis (DA), data result delivery (DD), and system operation (SO). Although not exactly named the same, these were discussed in the work by Pohl et al. [PBT18], who discussed a *data-science-as-a-service model*. While the first step includes the connection as well as the integration of data sources and the storage of data, the second focuses on data cleansing and transformation. Within the third step, data analysis, statistical modeling, machine learning, data mining, and various other activities are performed to facilitate data description, pattern recognition, and knowledge generation. In the form of reports and graphical representations, the results are then described by the data result delivery. All of the related functionalities or the results themselves can be used in the system operation stage. An overview of all of these general steps can be found in Table 4.10.

General Step	**Definition**
Data Ingestion (DI)	Linkage and integration of data sources and persistence of raw data
Data Preparation (DP)	Cleansing, transformation, and structure modeling of data
Data Analysis (DA)	Recognition, modeling, mining, clustering and processing of the data using specific methods
Data Result Delivery (DD)	Implementation of the achieved results, their delivery, and visual representation
Data Operation (SO)	Integration and application of the results in a system context

Table 4.10: General steps of a data science process, based on [VSJ+20]

Additionally, the approach of the ontology development process by Neuhaus et al. [NVB+13] was implicitly followed, in which competency questions were asked, and continuous evaluation was conducted to ensure a high *fidelity*, *craftsmanship*, and *fitness* of the identified classes, their relation and thus the artifact of the section. Most of these questions were derived from the initially highlighted problem situation, such as: *"Which manifestations are performing data analytics operations?"*. For the prevention of any misunderstandings and inconsistencies, care was taken not to misinterpret or overlap with other terms [NM01]. This is, for instance, the case with synonyms, homonyms, and meronyms. In the case of existing synonyms, the annotation `has_synonym` was used.

For example, the class `tool` was annotated with the synonym *Software* as it originates from the SWO. Following the construction of the initial taxonomy, every step was substantially and equally performed to identify the missing structure of relationships and properties. Creative techniques or further methods to determine other elements of the ontology were not used, as proposed, for instance, by Uschold and King [Mik95].

Because of the focus on the domain instead of the top-level ontology, only relevant information described in the context of this work was partially included here. As a result, the ontology will cover even more information as they will be relevant for the later implementation context. Hence, in the case of standalone usage, a comprehensive overview of existing concepts and their relation is provided. An excerpt of the created taxonomy is shown in Figure 4.9, by using the visualization plugin OWLViz [Hor19] in Protégé.

In fact, the entire ontology was created using the open-source editor Protégé [Mus15], which represents the de facto standard for creating and maintaining ontologies, mainly due to the availability of various plugins, widgets, reasoners, and graphical visualizations. The main goal of the developed Big Data Technology Ontology (BDTOnto) is to depict technology-relevant information and the involvement of those along with the realization of big data projects. In doing so, the relevant technologies, the fulfilled FR and NFR, SUCs that might deliver useful orientation for unplanned projects, and various fundamentals are included. Emerging from the root node, called BDTOnto, the top-level class `Data Science Process` is used. The class consists of the sub-classes `General Step` and `Functionality`. While the first contains all of the steps as they have been described in Table 4.10 and throughout the work, related functionalities of each phase are covered by the second sub-class.

Superordinate classes such as `Information Content Entity` and `Material Entity` were reused from existing ontologies to achieve a better segmentation and comprehensibility, especially for those who are familiar with the used mid-level ontologies SWO and IAO. Subordinate classes, such as `License` and `Organization`, were formulated for the sake of completeness. These are intended to hold relevant information for the further specification of big data technologies. By following the previously formulated definitions of related terms, Table 4, the latter are included under the `Information Content Entity`, similar to IAO. Here, each tool related to the domain of big data, seen as a kind of software

component, is included within the same-named class. The path to this class is derived from the structure of the formulated definitions.

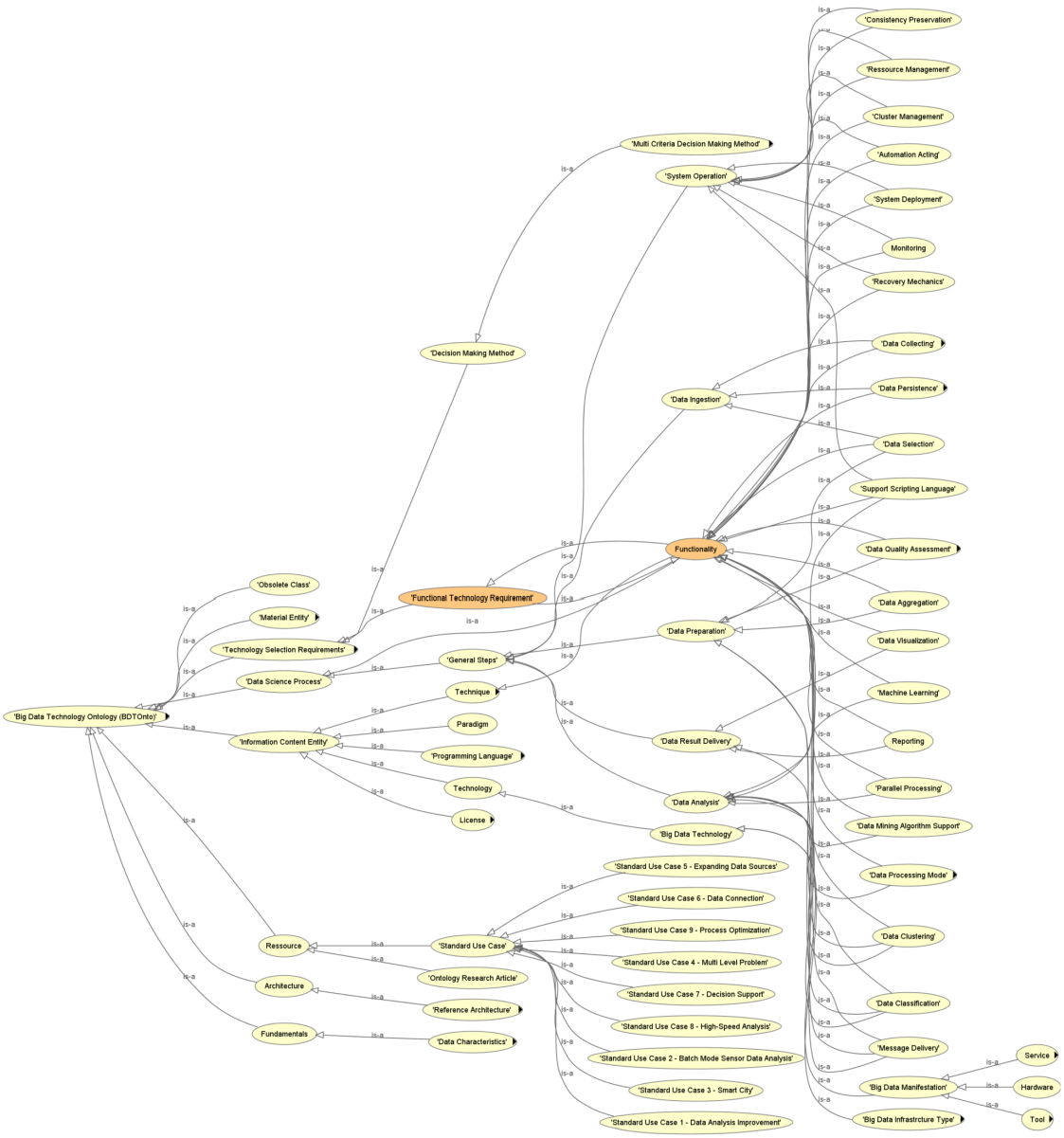


Figure 4.9: Excerpt of the basic taxonomy of the developed BDTOnto

Relevant requirements identified for selecting the SUCs and the technologies themselves, as described in section 4.4, are included under the class `Technology Selection Requirements`. Here, the sub-classes `Functional Technology Requirement`, `Non-Functional Technology Requirement`, and `Decision Making Method` are contained. In each of them, distinctive sub-classes are formulated as they emerge throughout this work.

4.3.2 Design of the Big Data Technology Ontology (BDTOnto)

After all key entities, their primary relationships, and properties were gathered and the baseline taxonomy created, the development of the BDTOnto was continued. Every class, relation, and property was described by various annotations, including self-defined as well as commonly used RDF Schema (RDFS) and Dublin Core (DC) properties, such as `rdfs:label`, `rdfs:comment`, `dc:creator` and `dc:description` to provide as much information as possible. This also includes the planned usage, synonyms, and the original source. Certain concepts are oriented towards the previously investigated ontologies SWO, IAO, and OntoDM. This applies to both, the classes and their relationship structure, which were further denoted by the annotations imported_from and `dc:source`. The definition of the used resources and profiles of the created OWL file are depicted in Figure 4.10.

Apart from essential entries, such as the required namespace identifier `xmlns:rdf`, `xmlns:owl` or `xlmns:dc`, further specifications regarding the ontology were made. This includes the used namespace for the self-created properties and relations within the ontology, using `xlmns:BDTOnto` as well as descriptive information, denoted, e.g., by the `owl:versionIRI`. Further information regarding the instantiation of the related files can be found at the Word Wide Web Consortium (W3C)[W317].

```
1  <?xml version="1.0"?>
2  <rdf:RDF xmlns="http://www.mrcc.ovgu.de/mrcc/team/matthias-
       volk/BDTOnto#"
3        xml:base="http://www.mrcc.ovgu.de/mrcc/team/matthias-
           volk/BDTOnto"
4        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
5        xmlns:owl="http://www.w3.org/2002/07/owl#"
6        xmlns:BDTOnto="http://www.mrcc.ovgu.de/mrcc/team/
           matthias-volk/BDTOnto#"
7        xmlns:xml="http://www.w3.org/XML/1998/namespace"
8        xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
9        xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
10       xmlns:dc="http://purl.org/dc/elements/1.1/">
11     <owl:Ontology rdf:about="http://www.mrcc.ovgu.de/mrcc/
         team/matthias-volk/BDTOnto">
12     <owl:versionIRI rdf:resource="http://www.mrcc.ovgu.de/
         mrcc/team/matthias-volk/BDTOnto-1.1"/>
13     <dc:identifier rdf:datatype="http://www.w3.org/2001/
         XMLSchema#string">BDTOnto</dc:identifier>
14     <dc:description xml:lang="en">A thorough description of
         the artifact. </dc:description>
```

Figure 4.10: The baseline definition of the created OWL file

The already mentioned top-level class `Data Science Process` and its sub-classes `General Step` and `Functionality` define the baseline of the ontology, serving as a bridge between the others. Each of the `General Step` class elements describes one of the aforementioned overarching process steps (cf. Table 4.10). Through the use of the `direct_followed_by`

relation, originating from the SWO, the realization of each of those is described in a structured order. All functionalities typically required within a big data project, as described in section 4.2.4, are listed within the Functionality class. In order to point out the connection between a specific process step and the typical functions located here, each function serves as a sub-class of `Functionality` and the respective `Data Science Process` sub-class, such as `Data Analysis`. An excerpt of this structure is shown in Figure 4.11. This and the following structural overviews, were created using Protégé and OntoGraf [Fal10].

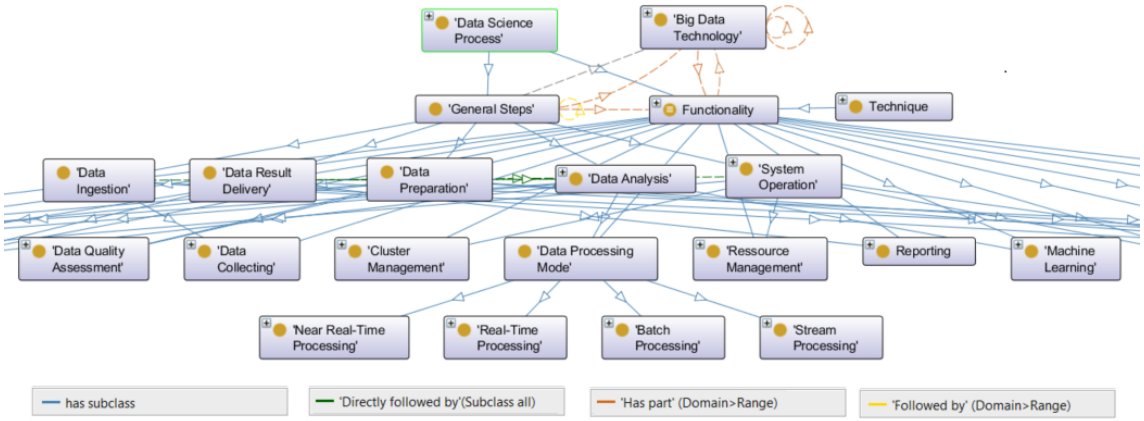


Figure 4.11: An excerpt of the ontology showing the functional hierarchy and dependencies using Protégé and OntoGraf

For an in-depth description of each of the functionalities and the implemented definitions within the ontology, Table A.4 in the Appendix can be checked. As one can note, based on the given Table 4.10 and the made inferences up to this point, the functionalities are not only part of each step within the general data science process. At the same time, they depict an equivalent of the FRs, which should be considered for setting up big data projects. As described before, the linkage is made with another superordinate class that holds the information of FRs and NFRs, namely `Technology Selection Requirements`. Similar to the FRs, the information for the NFRs emerged from previous research, as shown and described in section 4.2.4. Notably, MCDM algorithms and their specificities, regarding the later consideration, are included here. As highlighted by various research contributions (cf. section 3.1.2), the consideration of NFRs tend to be sensible for the selection of related technologies. Especially the approach by [Lně15] was often used as the foundation for other approaches. In their paper, further specifications were made regarding the NFRs. Hence, a similar categorization to increase the level of detail, as they recommended, using the classes `Expense`, `Social` and `Technical`, was implemented here. The relations between these classes can also be seen in Figure 4.12. The superordinate class containing relevant SUCs called `Ressources` holds appropriate knowledge regarding the successful realization of big data projects. Besides all defined SUCs and their specificities, all use cases are described in section 4.2.3.

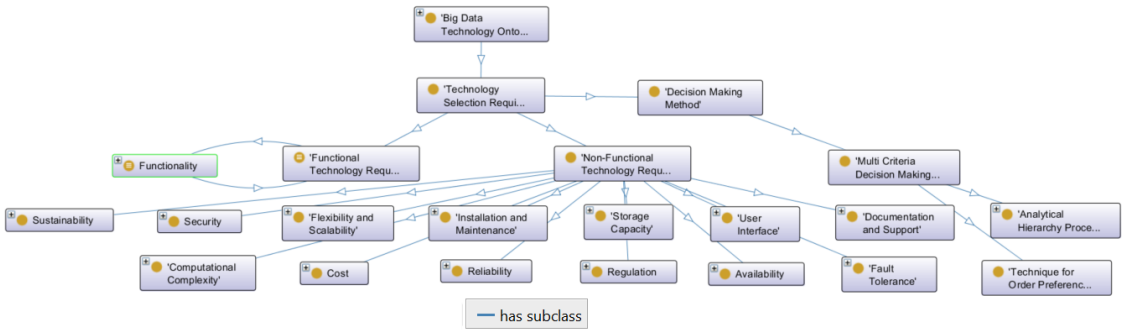


Figure 4.12: Overview of the functional and non-functional requirements mapping

Due to the natural deduction from the general to the specific, the SUCs were denoted as separate classes and the use cases as *individuals*. The same was done for the weighting of each NFR. For each SUC, a mapping regarding the described severity of each NFR, ranging from 1-5, was realized by using individuals. For example, for the first SUC and the availability, the individual `Weight_SUC_1_Availability` was created, containing the data property assertions `has_weight_value` and `has_maximum_weight_value`. While the first described the average value, calculated by the individual rating of each included case, the latter denoted the absolute maximum that was ascertained. Because of the potential influence of outliners, those were added for the sake of completeness. Another relation was made between the single functionalities and SUCs. Whenever one of the included functionalities was fulfilled, as described in Table 4.9, the `has_participant` relation was used, as the IAO provides it. Apart from those SUCs, relevant published articles that emerged in the context of this thesis are included in the `Ressource` class, predominantly to increase the comprehensibility of the developed artifact. The class `Ontology_Research_Article` contains all relevant articles, as they have been described earlier in Table 1.1. A complete overview of the managed resources, primarily focusing on the individual SUCs, is given in Figure 4.13. In particular, SUC 4 is further specified by showing the relevant individuals (third level) as well as the fulfilled functions on the last level.

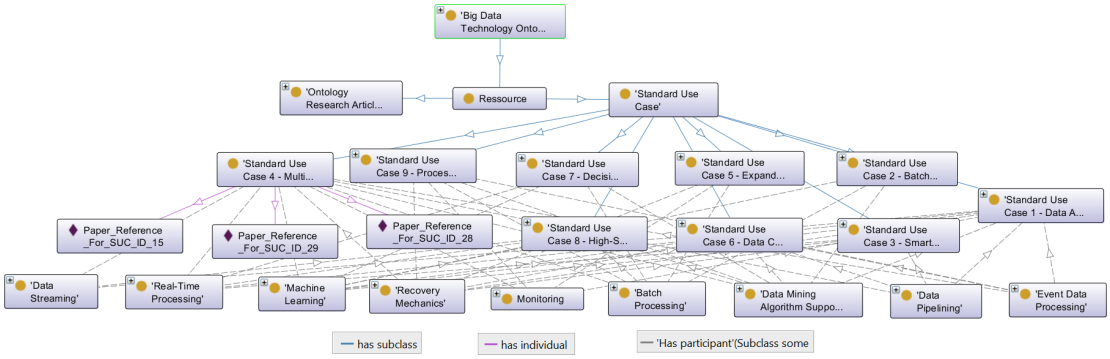


Figure 4.13: Excerpt of the stored SUCs, revealing the mapped use case and requested functionalities

As highlighted before, definitions and existing approaches were used for the categorization of the technologies as they emerged in the context of this work. As the core and main driver of the ontology and its creation, the class `Tool` covers big data technologies. This, in turn, represents a specification of `Big Data Manifestations` that incorporates other elements, such as `Hardware` and `Service`. The main idea behind this was mainly driven by the absence of a distinct definition and differentiation of the terms, as stated in section 2.2.4. Each specific tool placed as a leaf offers extensive information like the origin and exemplarily use, as it is later discussed in section 5.5.3. A complete list of all of these can be seen in Figure 4.14. Notably, based on the given depiction, each tool is related to single sub-classes of the `Non-Functional Technology Requirement` and `Functional Technology Requirement` (or its equivalent `Functionality`) class, which are located in the `Technology Selection Requirements`.

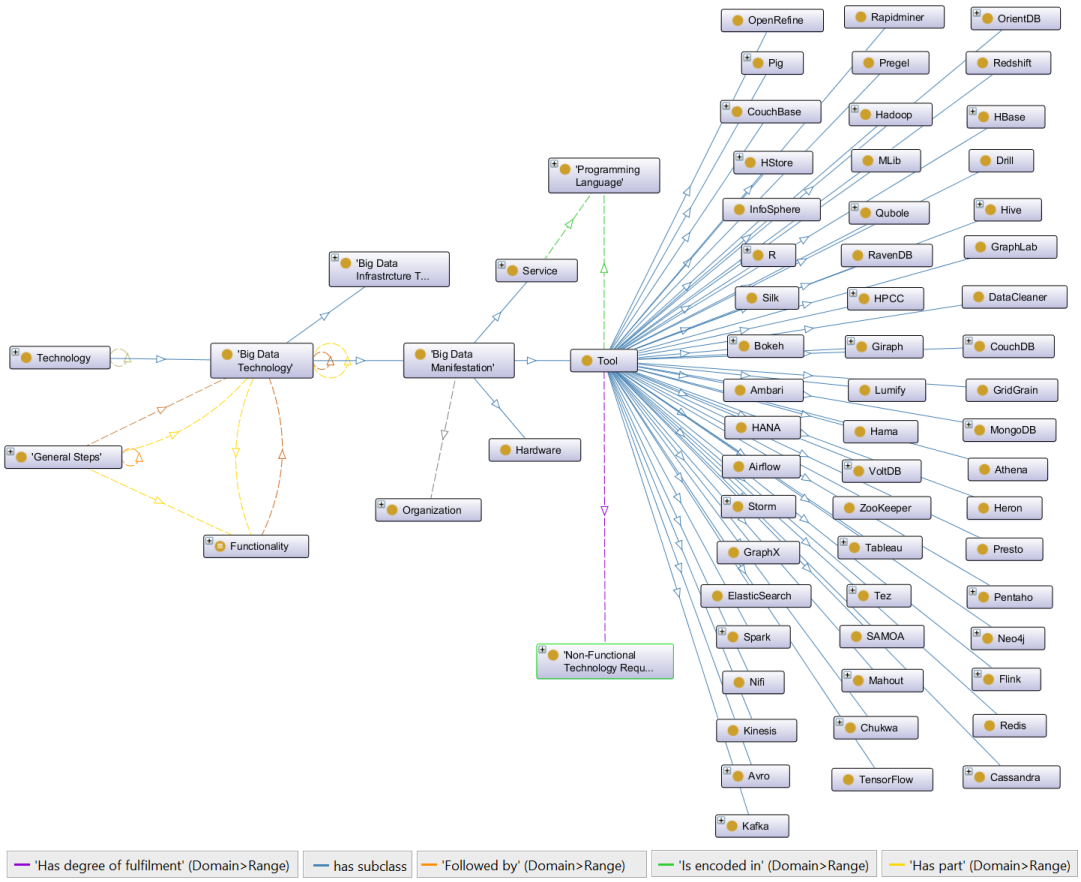


Figure 4.14: Overview of the included technologies using Protégé and OntoGraf

For both of them, a similar relation is made as in the case of the SUCs. While the functionalities are connected via the `has_participant` relation, the latter uses a separate individual for each tool as well as the relation `has_degree_of_fulfillment`. For instance, for Hadoop, the following string for the depiction of the relation is used:

```
'Has degree of fulfilment' only ({Weight_Hadoop_Availability,
    Weight_Hadoop_Computational_Complexity,
   Weight_Hadoop_Cost,
   Weight_Hadoop_Documentation_and_Support,
   Weight_Hadoop_Fault_Tolerance,
   Weight_Hadoop_Flexibility_and_Scaleability,
   Weight_Hadoop_Installation_and_Maintenance,
   Weight_Hadoop_Regulation, Weight_Hadoop_Reliability,
   Weight_Hadoop_Security,...})
```

Currently, the ontology covers the information of 51 tools. Although not all annotations are always used to describe these, the information needed for the FRs and NFRs is included. This was done by performing an extensive material collection, where tool documentations, blog entries, and even research articles were examined. While for the FRs, no greater deliberations were required in terms of the degree of fulfillment and, thus, every piece of information was taken as evidence, the collection of the NFRs required an extra step. In particular, this refers to the assigned severity. Similar to the SUCs a Likert scale from 1-5 was used. To facilitate a comprehensible and reproducible assignment, different criteria need to be satisfied for each rating of every NFR. In particular, the criteria were formulated for the values one, three, and five. The remaining values two and four were used to make possible gradations when classifications for the surrounding categories were not possible. These can be found in Table A.5. The made assessments for all technologies, the respective FRs, and NFRs can be found in Table A.6 and Table A.7.

Apart from the connection to existing FRs and NFRs, compatibilities between certain technologies and their specific version were partially realized using `is_compatible_to`, as data object properties in Protégé. Drawing whole-part relationships, the eponymous object property `part_of`, and its inverse `has_part`, oriented towards the IAO, were applied. In the case of multiple occurrences or similar definitions, as in the case of the class `Software` and the previously defined tool, the `has_synonym` annotation was used. However, it needs to be mentioned that for both of the highlighted relations, further specifications were only performed for some of the included tools, showcasing the overall capability. Especially with view on the number of integrated tools and their potential interplay, numerous information regarding their compatibility is required to sufficiently cover the current state of those. This, in turn, would require not only the investigation of existing documentations, marketing pages, and other material.

Furthermore, functional tests would be necessary to ensure that specific versions are really compatible with each other. In case that this feature is expanded, a community-driven approach could be harnessed in which interested users integrate those compatibilities based on their research, experience, and own tests. However, this specific feature is not further considered in this work due to the effort for continuous maintenance and the complexity of an extended integration. Notwithstanding that, an example shall be given, which provides an idea and visual representation of this characteristic.

This is realized in detail by using Apache Hadoop, which includes various modules allowing, for instance, distributed processing of large data sets using clusters and simple programming models [Apa22a]. For every related tool of a specific, a separate individual was created, containing the name and the specific version (e.g., `Hadoop_V3.2`). While general information is aligned to the tool class, version-specific information is covered by the individuals and various data properties. This includes, among other things, the release date (`versionDate`), specified by the data type `xsd:dateTime`, and `downloadLocation` with the type `xsd:anyURI`.

Hadoop and Hive were used to showcase the relation between different tools and their compatibilities. The Apache Hive project provides an additional data warehouse infrastructure as part of the complete Hadoop ecosystem. Hence, Hive and Hadoop's general classes were linked using the `has_part` relation. Depending on the specific version of the tools, these can be related to each other using the symmetric and transitive property `compatible_to` on the individuals' level. The resulting interconnections are depicted Figure 4.15. Eventually, it should be mentioned that not every single detail was shared. Due to the comprehensiveness of the respective ontology, the particular file should be checked to obtain a better overview and description of the elements. In doing so, the tool used for the creation and management, called Protégé is recommended.

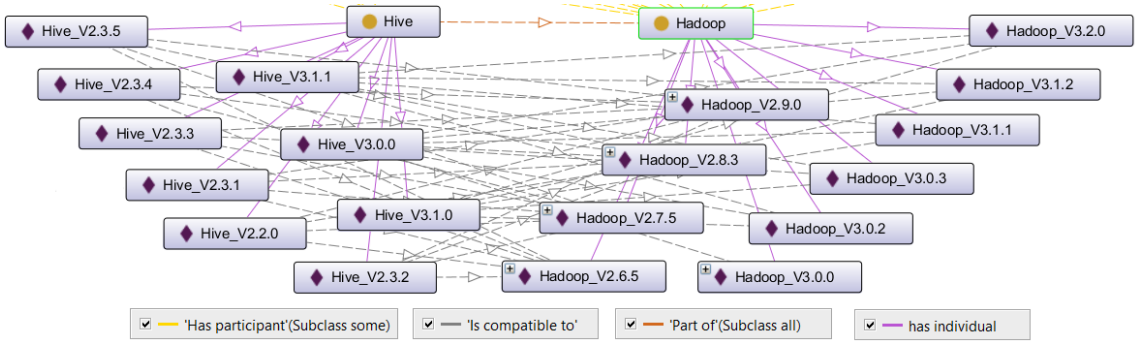


Figure 4.15: Tool compatibility relation on the individuals' level [VSJ+20]

4.3.3 Evaluation of the Developed BDTOnto

The chosen DSR methodology, as well as the OE, requires an evaluation of the developed artifact to check whether the designed solution fulfills its purpose. For this task, in the domain of ontologies, the concept of continuous evaluation has been established in the literature [Góm01; NVB+13; FGJ97b]. The DSR also intends a similar idea. In particular, this refers to the DSR evaluation approach proposed by Sonnenberg and vom Brocke [SV12b; SV12a], which was previously discussed in section 1.3. The Eval activities were also applied for this particular component.

- **Eval 1:** The first phase, Eval 1, legitimizes the research endeavor by outlining its necessity. This includes the identification of a problem statement or gap and the jus-

tification of the design objectives [SV12a]. The importance of a thorough knowledge base was motivated by the tremendous amount of existing research and information in this domain while still being confronted with a shortage of skilled talents and experts. In many cases, essential activities, mainly related to identifying and selecting suitable technologies, require comprehensive expertise. At the same time, the number of technologies is continuously increasing, reinforcing the problem. Therefore, extensive expedients are almost mandatory for optimal results [SPB16; SS13]. Otherwise, it is likely, that new and possibly superior solutions are being ignored in favor of well-known approaches due to the ignorance regarding their existence. To do so, a modifiable knowledge base is required that allows the integration and extension of new elements and relations.

The proposed ontology constitutes such a solution that can, if appropriately maintained, provide an overview of relevant information, available tools, as well as all their connections and dependencies. Since emerging approaches can be integrated incrementally, allowing for ongoing growth, and vanished ones can be simply deleted, the maintenance is comparatively uncomplicated. Furthermore, due to their machine readability, ontologies cannot solely be used in an isolated way but are also capable of further integrations, such as in the intended DSS as described before and revealed by several examples [KML+16; MWH+16; SK16]. Therefore, the questions regarding the importance, applicability, and plausibility can all be answered positively, suggesting the opportunity of creating a genuine added value by further pursuing the proposed concept.

- **Eval 2:** After establishing the general justification of the idea, Eval 2 addresses the evaluation of the design itself. This includes the specification along with the chosen tools and methodology. For this artifact, the course of action for creating the initial taxonomy is based on commonly accepted procedures. The same applies to the used concepts and structures that emerge from further research artifacts, such as the SUCs or used requirements. These findings, as well as the followed middle-out approach [Mik95] and used ontologies originate from the present body of knowledge. Thus, all aspects can be considered as already positively evaluated.

  Furthermore, the ontology development process by Neuhaus et al. [NVB+13] was followed, ensuring a continuous review of the developed structures. By using Protégé for the artifact's construction, a proven tool was used that can be regarded as a standard. Since the general concept of ontologies is suitable for the desired task, as shown in Eval 1, the used fundamentals are approved by the scientific community, and the entirety of the available information is preserved during the continuously reviewed creation of the ontology, Eval 2 is completed with a positive verdict.
- **Eval 3:** The next evaluation step, Eval 3, deals with providing a proof of concept. For this reason, a prototypical implementation of the evaluated artifact is realized and validated in an artificial setting. Besides the overall creation of the ontology as a

working proof of concept, a first practical application of the prototype can be found in [RVN+19], which proves its general functionality. Here, techniques for annotating huge volumes of financial text documents were analyzed. To handle those volumes, big data technologies were chosen and applied using an early prototype of a DSS that utilizes the artifact developed in this section.

- **Eval 4:** The final stage of the evaluation, Eval 4, is executed by applying the regarded artifact in a naturalistic setting. That way it can be determined to which extent it is not only working correctly but also if its usage provides a real benefit when fully integrated, for instance, into an organization and its structures [SV12a]. Based on the description above, completion of this step is not finished yet. However, since the ontology serves only as one part of the later framework, the fulfillment of this Eval step will be fulfilled by successfully evaluating the overarching artifact in which the ontology plays an essential role.

4.4 Multi-Criteria Decision Making for Big Data Projects

Numerous works were already identified in advance that allows the selection of big data technologies. In addition to structured processes, numerical approaches were also found that supported the identification of individual technologies and environments with the help of MCDM algorithms. While these approaches give an idea about the suitability of an MCDM application for these purposes or other related environment decisions, it was noted that they are either conceptual, limited in their level of detail and applicability, or barely adjustable and extendable. Furthermore, in many cases, only a few criteria were observed, primarily focusing only on qualitative attributes of the planned systems.

As they have been discussed before, functional aspects were only partially found. To overcome the scarcity of a method utilized to select big data technologies, a multi-staged MCDM application is proposed that considers both, FRs and NFRs. The sub-chapter itself is based on the previous research described in [VST21]. Accordingly, the idea of a multi-staged MCDM method is introduced in the first section, followed by a detailed description of the used algorithm and its use.

4.4.1 A Multi-Staged MCDM Method for Big Data Technology Selection

The overarching idea to incorporate both, FRs and NFRs, was to do this in a stepwise manner. As highlighted, particular functionalities, as requested via FRs, can be either fulfilled or not by a potential solution. In turn, NFRs are instead used to specify further quality attributes that influence the overall appearance of the possible system. Based on this observation, a five-stepped procedure was developed. While the first and second steps must be performed manually to deliver the required input information, the remaining three steps can be done via computer assistance. As recommended for a multitude of different

problems, one of the initial steps in a multi-staged approach is constituted by the general identification of the problem as well as the requirements used for the criteria set [TZ13].

Therefore, it is mandatory for a big data project to define the application environment and, thus, the most important information (1). In doing so, requirements for the potential system need to be ascertained and specified in more detail through established methods, as proposed in section 2.1.2 and 4.1.3. Eventually, the number of identified requirements, mainly focusing on FRs, might indicate the requested functionalities and thus the technologies required for the fulfillment (2). Afterward, the FRs, respectively functionalities, of the targeted system are compared with those that are provided by individual technologies. Due to the number of available technologies and required comparisons, a computer-supported solution could be needed, as it is investigated in the context of this work. Generally speaking, for a potential DSS, to achieve a pre-filtering, the user receives a logically ordered list containing the possible functionalities, at which he can either select or deselect all entries that are (not) relevant for the planned endeavor. In terms of a prechecked list, the inexperienced user could only deselect those which are certainly not necessary.

This procedure has the advantage that inexperienced decision makers are provided with a generally valid technology combination recommendation for the future use case. In contrast, the targeted selection can lead to specialized solutions. While the latter is particularly suitable for experienced users looking for specific technologies to possibly expand existing infrastructures, the latter is ideal for projects starting from the greenfield. However, due to the all-encompassing functionalities that are to be covered here, such solutions can be very complex and resource-intensive in their creation, maintenance, and application (3). The remaining technologies are then investigated in terms of their overarching suitability. In doing so, an assessment of NFRs, as found in Table 4.8 and described in Table A.7, is performed by the user, based on the remaining set of technologies. Depending on the chosen MCDM methods, a pairwise comparison can be required.

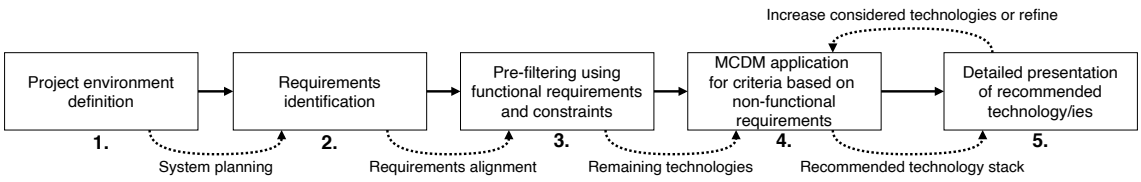


Figure 4.16: Concept of a multi-staged MCDM method [VST21]

In the end, the final selection and ranking of the potential technology combination are presented (5). In case refinements of the made preferences are required or one technology is insufficient and a stack of multiple technologies is needed to fulfill the degree of the requested functionalities, a staged MCDM is performed within the last two steps (4 & 5). While only a revision of the given preferences is performed in the first scenario, in the

second, the number of used technologies can be increased as long as no suitable stack of technologies was identified that fulfills the demand of the decision maker. Depending on the implemented MCDM approach, specific adjustments might be required, with which this can be realized. The described approach is depicted in Figure 4.16.

4.4.2 Algorithms for the Identification of Big Data Technologies

By considering the information given up to this point, the project definition (1) and the identification of relevant requirements should be manageable (2). In the third step, the aforementioned comparison of all required functionalities with those provided individually by the big data technology is carried out. If there are already individual candidates here that cover all functionalities entirely, these are returned as such. Otherwise, these are investigated in terms of potential combinations. Correspondingly, two different algorithms can be defined here, one for identifying single best solutions (3a), and the other for identifying technology combinations (3b). The procedure for 3a is shown in Algorithm 1.

Algorithm 1 Identification of the single best big data technology

Require: rf = Requested functionalities
Require: lat = List of available technologies
Ensure: rtl = Recommended technology List

procedure SINGLEBEST(rf, lat)
    ctl = Create empty *complete technology list* ▷ Technologies fulfill all RF
    itl = Create empty *incomplete technology list* ▷ Technologies partially fulfill RF
    for all t=Technology $\in lat$ **do**
        $isBestTech$ = check whether t fulfills all rf
        if $isBestTech == true$ **then**
            $ctl \leftarrow t$
        else
            $itl \leftarrow t$
        end if
    end for
    if $SIZE(ctl) == 1$ **then**
        $rtl \leftarrow ctl$
    else if $SIZE(ctl) \geq 2$ **then**
        $ctl \leftarrow$ MCDM(ctl)
        Sort ctl by MCDM value
        $rtl \leftarrow ctl$
    else
        Calculate number of fulfilled functionalities itl
        Sort itl by number of fulfilled functionalities
        $rtl \leftarrow itl$
    end if
    return rtl
end procedure

Specifically, this means that each available technology and its offered functionalities are compared with the required ones for the first part of the pre-filtering. If exactly one solution is found, it is given as a direct result, and no consideration of the MCDM value occurs. However, if more than one perfect solution is found, one such value is calculated for each of them, where the NFRs play a role. The sorted list is then the output as the result. Further comparisons are necessary, when no perfect single solution is found. All related technologies partially covering the functionalities are then used as input for 3b. Equivalent to 3a, a similar procedure is performed in the second step, focusing on the technology combinations, which is depicted in Algorithm 2.

The list of incomplete technologies is taken as input, and corresponding candidates are sought for completion. As generically described before, the combination does not represent a trivial undertaking in the general approach. First, all basic technologies from the input list are considered as starting points for possible combinations since they can fulfill at least one of the required functionalities. For each candidate found, a list of the still needed functionalities is created, which must be fulfilled with additional technologies until all of them can be covered. A candidate is selected according to the fulfillment of the remaining functionalities. If several suitable technologies satisfy the exact same remaining functionalities equally, these are chosen based on the MCDM value. All technologies to be examined also originate from the same list since these, as mentioned, meet at least one of the functionalities. The process itself is repeated until a suitable combination has been identified for each individual technology from the input list that covers all functionalities. Subsequently, all realizable functionalities are identified for each combination, and the MCDM value is calculated. The former refers not only to the required functionalities but also to those enabled by the complete technology stack.

In the second case, the MCDM value is calculated on the basis of the average value of each NFR within the combination. After that, all those information is provided. The resulting list can be sorted by preference either according to the MCDM or the number of functionalities to be fulfilled. In this way, the user has the greatest possible scope for decision-making and is supported extensively. As the most used MCDM method, both in general and for the articles found in the context of this work (cf. section 3.1.2), the AHP algorithm could be identified. The complete procedure for calculating the AHP is shown in Algorithm 3. As described in section 2.3.2, pairwise comparisons concerning single conflictive criteria are necessary, represented here by the previously described NFRs (cf. Table 4.9). As usual in AHP, a value between 1 and 9 is assigned for each comparison, and a reciprocal is used for the permuted combination. If both criteria have equal importance and there is no dominance in any direction, a one is assigned.

Algorithm 2 Identification of big data technology combinations

Require: rf = Requested functionalities
Require: rtl = Recommended technology list
Require: $sortT$ = Type of sorting
Ensure: ctl = List of all recommended technology combinations

```
1: procedure TechnologyComb(rf, rtl)
2:     for all t=Technology ∈ rtl do
3:         stc = Create empty single technology combination
4:         stc ← t
5:         copyRtl = rtl
6:         while stc does not fulfill all rf do
7:             cbt = Initiate current best technology combination candidate
8:             for all ct=Technology ∈ copyRtl do
9:                 if t ≠ ct and ct fulfills missing rf then
10:                    if cbt == NULL then
11:                        cbt = ct
12:                    else if ct fulfills more rf than cbt then
13:                        cbt = ct
14:                    else if ct fulfills same rf as cbt and has a higher MCDM value then
15:                        cbt = ct
16:                    end if
17:                end if
18:            end for
19:            if cbt ≠ NULL then
20:                stc ← cbt
21:                Remove cbt from copyRtl
22:            end if
23:        end while
24:        Calculate the average for each NFR ∀technologies∈ stc
25:        ctl ← stc
26:    end for
27:    Calculate MCDM based on the average for each stc ∈ ctl
28:    if sortT == byMCDM then
29:        Sort ctl by MCDM
30:    else
31:        Sort by minimum number of t and maximal fulfilled FRs of each stc ∈ ctl
32:    end if
33:    return ctl
34: end procedure
```

In particular, if certain comparisons cannot be performed because the necessary information about the project is unknown or undefined, the default value `one` for indifferent comparisons is assigned. When all comparisons have been completed, the calculation takes place. This includes the calculation of the weighting vector W for the given preferences of the decision maker and the individual technology for each criterion. Especially for the latter, it is important to mention that only those technologies that withstand the previously

undertaken filtering in terms of fulfilled functionalities are taken into account. Concerning technology combinations, the average of all expressions is used, to not favor the outliers. After all vectors of the alternatives have been calculated, these are combined. The resulting matrix, which shows for each technology the weights to the individual NFRs, is finally multiplied by the weighting vector from the own preferences. The resulting recommendation vector contains the percentage recommendation for the checked technologies or their combination.

Algorithm 3 AHP to identify the MCDM value of a technology(-stack) using NFRs

Require: rtl = Recommended technology list
Require: upl = List of preferences provided by the user $nfrl$ = NFR List
Ensure: atl = List of all recommended technology combinations with the AHP value

procedure MCDMAHPCalcuation(rtl, upl)
    cm = Create comparison matrix from given preferences upl
    cm = Normalize matrix cm
    w = Create weighting vector of m
    ws = Create weighted sum vector of m
    rm = Create blank rating matrix
    for all nfr=NFR $\in nfrl$ **do**
        $altMat$ = Create blank matrix with length $size(rtl)$
        for $i = 0$ **to** $size(rtl)$ **do**
            ti=Technology at position i
            for $j = 0$ **to** $size(rtl)$ **do**
                tj=Technology at position j
                for all t=Technology $\in rtl$ **do**
                    $iNFRV$ = Get rating value of nfr of technology j
                    $jNFRV$ = Get rating value of nfr of technology i
                    $value$ = get value from Table 4.11 using $iNFRV$ and $jNFRV$
                    Set $value$ in $altMat$ at position i, j
                    Set reciprocal value in $altMat$ at position j, i
                end for
            end for
            wc = Create weighting vector of $altMat$
            $rm \leftarrow wc$ ▷ Add vector to rating matrix
        end for
        $rVector = rm \times w$
        Assign each technology in rtl the corresponding MCDM value from $rVector$
        return rtl

Typically, when performing the pairwise comparison of the individual technologies, a rating between one and five is used, similar to the SUCs described above (cf. Table 4.9). This is because the values should be easy to reuse without being too dependent on the selected MCDM algorithm. Therefore, additional assistance is taken up here. Specifically, an interpretation table for AHP ratings is used, as it appears in the presented algorithm.

The table below gives the initial AHP rating for comparing a specific technology with another one (cf. Table 4.11). For example, if the technology to be compared has a rating value (RV) of four and another has a RV of two, where RV describes the severity of a particular NFR, the value six is assigned. This is done in the respective cell of the AHP matrix, indicated by the position of NFRs to be compared (position i,k). The same applies to the inverse location (position j,i), in which the reciprocal value is put. In case both ratings are the same, an indifferent AHP comparison rating of one is assigned. By means of this, all gradations similar to table Table 6 can also be mapped here. The table itself was manually created by observing each of the recommended values and their gradations.

RV(j)/RV(i)	**1**	**2**	**3**	**4**	**5**
1	1	3	5	7	9
2	1/3	1	3	6	8
3	1/5	1/3	1	3	6
4	1/7	1/6	1/3	1	3
5	1/9	1/8	1/6	1/3	1

Table 4.11: Transformation of the given NFR ratings, of each technology, into AHP conform ratings

As the detailed explanation and the three algorithms have demonstrated, the steps in the multi-stage process are not entirely detached from each other. Thus, a pre-filtering with respect to the functionalities is not completely isolated at first. Instead, in particular, the third and fourth steps are rather interwoven with each other. The fine-grained representation of the process described above, as also described by means of 3a and 3b and taking into account the AHP, is, therefore, shown once again in Figure 4.17.

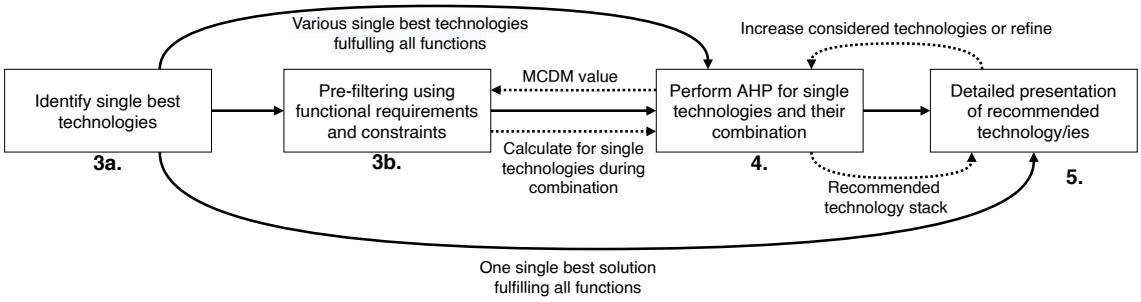


Figure 4.17: Extended multi-staged MCDM approach using the AHP

4.4.3 Evaluation of the Multi-Staged MCDM Approach

To conform to the used methodology as well as the steps performed for the other research artifacts, an evaluation for this procedure was performed to examine the validity of the developed solution. In doing so, an initial experiment was realized, using a preliminary version of the system to identify suitable technologies. Further details regarding the setup

and the inference engine in which the algorithms were implemented can be found in Chapter 5. Notwithstanding that, it should be highlighted that the BDTOnto was already used here to provide the stored ratings of each technology. For the evaluation of the approach, an existing use case was selected that originated from the results of the structured literature review that was conducted in the context of the related research [VST21].

In particular, the contribution provided by [CLC+15] was used, in which the realization of a smart city big data platform is described. Besides the description of essential FRs and NFRs, the solution architecture is also presented that depicts a combination of Hadoop, Spark, and CouchDB. This case served to be a suitable choice to find out if the developed prototype and, thus, the created multi-staged method delivers similar results. Although only partial information was identified and some vague judgments concerning the ratings were required, the implementation of the multi-staged procedure delivered sufficient results. A similar technology stack was determined that provided instead of CouchDB, the NoSQL database HBase. The main reason for the small difference can presumably be attributed to the specific tool that was not implemented within the BDTOnto at the time of the evaluation.

4.5 Deployment Diagrams for Modeling Big Data System Architectures

Models are a useful addition to support the planning of system architectures in advance and to derive corresponding requirements (cf. section 2.1.2). At the same time, they illustrate complex correlations by providing common specifications and notations for all relevant elements. In general, this discipline *"has become a common practice in modern software engineering"* [CHN12]. Especially for the communication with relevant stakeholders, a common understanding can be created, which is essential for the later interaction and possible transformation. Hence, the application of standardized modeling techniques seems to be a promising way to prevent misunderstandings, misinterpretations, and information gaps independent from the actual profession and expertise [DGK+11; CHN12].

As indicated in previous chapters and sections, although approaches for the provisioning of concrete implementation details in the form of use case descriptions, documentation, or reference architectures exist, these either miss the level of detail or the way of presentation. Many contributions are rather using self-defined depictions in favor of standardized approaches, as it can be seen in [Gee13; NHR+17]. In the course of this investigation, one possible approach will be presented, with the help of which a BDA can be modeled. This should not only be usable in the context of the envisaged solution but also, like the other artifacts, independently. All the information presented in the following is primarily based on the content described in [VSP+20]. Deduced from this, within the first sub-section, the current state regarding the modeling of BDA is described. Apart from existing at-

tempts that offer basic ideas, relevant information for modeling BDAs are extracted. In the adhering sub-sections, a potential solution is proposed and further evaluated.

4.5.1 State of the Art in Big Data System Architecture Modelling

As already highlighted during the presentation of related research articles to this work, no suitable language or particular tool was found that supports the modeling of BDA. There are also no sufficient preliminary works to build upon, which could be facilitated. Consequently, a keyword-based search in combination with the forward-backward search (F&B Search), based on the methods by [WW02; LJ06] was utilized to discover existing approaches to model BDA. At first, for the collection of the initial material, six literature databases were queried, namely AIS Electronic Library (AISEL), Science Direct (SD), ACM Digital Library (ACM), Institute of Electrical and Electronics Engineers Xplore Digital Library (IEEE), Scopus and Wiley Online Library (Wiley).

In all of them, search terms consisting of *big data*, *system*, *architecture*, *modeling*, and *model* were applied to title, abstract, and keywords. A two-stepped procedure was chosen for the refinement of all found papers, in which various inclusion and exclusion criteria were used. Their creation and application were made similarly to the previous structured literature reviews, as, for instance, described in Table 4.5. This includes observing peer-reviewed documents published in conferences, journals, or books, using English or German, and focusing on contributions that are not exclusively on single algorithms or frameworks. Instead, original articles were searched that dealt either implicitly or explicitly with modeling system architectures in the big data domain.

As described before, only the title, keywords, and abstract of the found articles were examined in the first phase of the two-stepped refinement procedure. All duplicates originating from meta-databases were removed. After that, the remaining papers were read thoroughly and qualitatively analyzed in the second phase. Eventually, 60 articles were deemed relevant. As mentioned before, as a third step, the forward and backward search [WW02] was subsequently conducted. This resulted in another nine contributions, focusing primarily on the realization of reference architectures. An overview of all results, including the origin and reference, is depicted in Table 4.12.

The assumed absence of a sufficient modeling approach was confirmed during the qualitative analysis of the found papers. This may imply that a specific modeling approach is not required because these do not differ very much from *traditional* IT architectures. Furthermore, authors could be insecure about the modeling due to the absence of a common approach or don't even realize that the created models should follow these. In fact, in only eleven contributions, architectures were depicted through the (partial) use of well-known diagrams from the UML, such as components [GTT+16; Gee13; NHR+17; VS14], deployment [Gee13; CFC+18] or package diagrams [GTT+16; Gee13].

Database	Article
AISEL	[TSL15; GCA15; BDS18; Goe15; CKG+17; PLB; SSB14; LPB16; ANW15]
SD	[PPP+18; YCJ+17; SGW+15; HCH17; NHR+17; CSG+17; ABD+19; AKP+18; BRA+18; BD18; Mis17; SWF17]
ACM	[CKG+16; ES16; GTT+16; KAS+18; KBB+16; Nie11; PV17; SNC+17; SB16; ZAR+17]
IEEE	[SXV16; CKH16a; DA18; BT15; GHK+18; HMD17; Bar14; KMM+15; PB16; KND+16; DGP+15; WWL+17; KL14; AIS+14; Liu18; MLR17; MCS14; CY16; SPK+15; SOI15; VS14]
IEEE	[CFC+18; KBP18; LX19; SDM+18; NKK18; SVP18; WSS+18]
Wiley	[BBL18]
F&B Search	[Gee13; ACB+15b; BSH16; CKH16b; CKH16c; DNM13; PP15; PZ14b; UPA+15]

Table 4.12: Results of the literature review focusing on system architecture models in big data

In turn, in 43 contributions, self-developed models were used. However, concerning the previous cases, modifications to notation elements and their interconnection were also partially observed here. One particular example is the case of [Gee13], in which a reference architecture for the realization of predictive analytics is introduced. This was also noticed for several other contributions. In general, only a few of them dealt either implicitly or explicitly with the formulation of requirements on BDA [NHR+17; HCH17; WSS+18; BDS18]. At this point, the ability to store, manage, and process *big data* was always of mandatory interest. An overview of all found architectures as well as the identified modeling types is depicted in Table 4.13.

Regarding the self-created models, it was noticed that these differ greatly from each other in terms of the information density of the BDA and the used notation elements for the description. In particular, some of the models focus only on the general overview, describing various components [KBB+16; DA18; LX19], while others deal with complex sub-systems and the relationship between each of the elements, such as in [SXV16; LPB16]. In multiple publications, only self-defined elements were identified, such as in [GHK+18; CSG+17; KND+16]. Those were used and modified from existing types of modeling languages or entirely newly created.

For instance, in [UPA+15], a system architecture for remote sensing of satellites was developed and presented. The very complex depiction groups the relevant components and their interaction into different levels, which are logically structured from the bottom-up, starting with acquiring the remote sensing data. Although a flow chart is later used for

the general workflow, the idea remains unclear at the beginning. Hence, interpretations concerning each of the components can be made. Similar to this, in [BD18], an architecture is constructed that intends to forecast social and economic changes.

Type	**Article**
Component diagram	[GTT+16; Gee13; NHR+17; VS14]
Deployment diagram	[Gee13; CFC+18]
Package diagram	[GTT+16; Gee13]
Flow diagram	[DGP+15; YCJ+17; UPA+15]
Self-created model	[SSB14; PP15; UPA+15; KMM+15; DNM13; BD18; KL14; Bar14; CKH16b; SGW+15; KBB+16; DA18; AIS+14; CY16; BT15; AKP+18; ANW15; TSL15; GHK+18; MCS14; SPK+15; CSG+17; PPP+18; BSH16; BRA+18; LPB16; SB16; SOI15; ABD+19; SWF17; HCH17; Liu18; LX19; WSS+18; PLB; PB16; WWL+17; HMD17; BDS18; KND+16; SXV16; NKK18; SVP18]

Table 4.13: Categorization of found modeling approaches [VSP+20]

Although the model gives a good and structured overview, few details are provided. Within the depiction of the system, the authors use the data life cycle to reveal the different stages, starting from the *data receiving* to the *publishing module*. In doing so, arrows between the different components show the flow of data. However, no detailed transition information is presented, such as the connection on a system level or the implemented functions. Thus, as one may note, the modeling of BDAs exhibits large differences, not only in terms of the level of detail. However, in most of the related papers, important information and elements were identified that are required to model a BDA. The overview of those is depicted in Table 4.14. In the following, each of those shall be discussed in more detail.

A *component*, as it was introduced before (cf. section 2.1 and 3.4), encapsulates several functionalities for a later reuse [MT00, p. 73]. Often, these are further distinguished into hardware and software components. Typically, they are either part of sub-systems or assembled to these. This corresponds to the description of the [OMG17] that refers to sub-systems as a collection of elements, components, and connections to fulfill a specific purpose within a system architecture.

In this regard, various types of sub-systems were identified, such as data source, data storage, data collection, data preparation, data integration, data preprocessing, analysis, visualization, and management. Data sources, in turn, consisting mostly of sensors, machines, databases, or streams, were almost always part of the found BDAs, as it can be seen in [ACB+15b; AKP+18; GHK+18; BSH16]. The same applies to the data storage that

was often described by the HDFS, data warehouses or SQL, NoSQL- and cloud storages, such as in [PP15; BD18; SOI15; PLB; PB16; WWL+17; Nie11].

Sometimes those components revealed concrete insights regarding the interfaces, functionalities, or the inner structure. At this point, executable (e.g. scripts, programs) or non-executable elements (e.g. web data, documents, videos) were often used as an input and output, and aligned to the term of an *artifact*. This also corresponds to the definition according to the UML that uses those elements in their deployment diagrams. In their documentation, an artifact *"represents some (usually refinable) item of information that is used or produced by a software development process or by operation of a system"* [OMG17].

Elements	**#**	**Article**
Subsystem	45	[PP15; UPA+15; KMM+15; DNM13; BD18; KL14; Bar14; CKH16b; KBB+16; DA18; Gee13; AIS+14; CY16; AKP+18; ANW15; HHR+17; GHK+18; SPK+15; CSG+17; PPP+18; BSH16; DGP+15; VS14; BRA+18; Goe15; LPB16; SB16; SOI15; ABD+19; SWF17; HCH17; Liu18; LX19; WSS+18; PLB; PB16; WWL+17; HMD17; Mis17; Nie11; BDS18; SXV16; CFC+18; NKK18; SVP18]
Component	48	[SSB14; PP15; UPA+15; KMM+15; DNM13; BD18; KL14; Bar14; CKH16b; SGW+15; GTT+16; KBB+16; DA18; Gee13; AIS+14; CY16; AKP+18; ANW15; HHR+17; TSL15; GHK+18; MCS14; SPK+15; CSG+17; PPP+18; BSH16; DGP+15; VS14; BRA+18; Goe15; YCJ+17; LPB16; ABD+19; SWF17; HCH17; Liu18; LX19; WSS+18; PLB; PB16; WWL+17; Mis17; BDS18; KND+16; SXV16; CFC+18; NKK18; SVP18]
Artifact	43	[SSB14; PP15; UPA+15; KMM+15; DNM13; BD18; KL14; Bar14; SGW+15; GTT+16; KBB+16; DA18; Gee13; CY16; BT15; AKP+18; HHR+17; TSL15; GHK+18; MCS14; SPK+15; CSG+17; PPP+18; BSH16; VS14; BRA+18; Goe15; YCJ+17; LPB16; ABD+19; SWF17; Liu18; LX19; WSS+18; PV17; PLB; PB16; WWL+17; HMD17; KND+16; SXV16; NKK18; SVP18]
Relation	43	[SSB14; PP15; UPA+15; KMM+15; DNM13; BD18; KL14; Bar14; CKH16b; SGW+15; GTT+16; KBB+16; DA18; Gee13; AIS+14; CY16; BT15; AKP+18; ANW15; HHR+17; GHK+18; MCS14; SPK+15; CSG+17; PPP+18; BSH16; VS14; BRA+18; YCJ+17; LPB16; ABD+19; SWF17; HCH17; Liu18; LX19; WSS+18; WWL+17; HMD17; KND+16; SXV16; CFC+18; NKK18; SVP18]
Actor	3	[VS14; SOI15; SSB14]
Software	30	[PZ14b; SSB14; PP15; KMM+15; GCA15; BD18; KL14; SGW+15; GTT+16; CY16; BT15; AKP+18; HHR+17; SPK+15; CSG+17; LPB16; SB16; ABD+19; PV17; SNC+17; PB16; HMD17; Nie11; SXV16; SDM+18; CFC+18; NKK18; SVP18; KBP18; BBL18]

Table 4.14: Identified elements of a BDA in the literature [VSP+20]

Relations are used in almost all of the found out papers, either using undirected lines or arrows. Sometimes legends or needed information were provided, but frequently not. At this point, the corresponding text sections had to be read or interpretations made. A closer look at this notation element revealed that the connection was often used for different purposes. For instance, in [SSB14; CKH16b; CY16; ANW15; PPP+18], it was used for the exchange of messages, such as inquiries for reports, uploads, and others. Whereas in [SGW+15; Gee13], deployments, as they are known from deployment diagrams, were implicitly depicted by this symbol.

An *actor* in this context describes a person that interacts with the intended BDA. Although these were addressed only in a few papers directly [VS14; SOI15; SSB14], they represent an essential piece of information that especially reveals how the system as a whole, single components, or the software is going to be used. The *software* element directly refers to specific implementation details regarding the used tools and technologies within the architecture. While some of the papers contained detailed information, in either the depicted architecture or the related paragraphs, others described general functionalities without naming concrete solutions.

Although numerous modeling approaches for a BDA were identified, no uniform applicable method was found within the contributions. Only a few made use of established and well-accepted diagrams, such as from the UML, to display a BDA. However, missing parts, incorrect use of notation elements, and other obstacles were identified even in those contributions. It can be assumed that this originates partially out of the targeted audience, which is here rather a researcher than a system architect. Again, it highlights the absence of a unified approach and the necessity to find a suitable way to depict BDA as a whole.

4.5.2 A Systematic Modeling Approach for BDAs

By definition, diagram types such as the deployment diagram already offer a good way to present system architectures [OMG17], even if they do not necessarily deal with big data-related elements. According to the OMG, a deployment diagram *"specifies constructs that can be used to define the execution architecture of systems and the assignment of software artifacts to system elements"* [OMG17]. Compared to the component diagrams, they can also give general insights to provide a conceptual overview of the architecture rather than a concrete functional implementation. Hence, in the context of the artifact required for this work, the adoption and modification of deployment diagrams appear to be suitable for modeling BDA. In order to perform manipulations for the realization of *"more elaborate deployment models"* [OMG17], profiles and meta-models can be used. Considering the results mentioned above, several changes were made to the conventional deployment diagrams from the literature review.

The element of a person, known from use-case diagrams, is integrated. This reflects

the fact that often a number of stakeholders interact with the system in different ways. The same applies to the other frequently used elements, such as the cloud or specific sub-systems. Furthermore, already existing diagram elements were changed, mainly focusing on the referred profiles. Artifacts that depict input and output received the stereotypes `processed_data`, `audio`, `video`, `picture`, and `webdata`. Sensors, predominately used in the IoT domain and smart cities, are defined through the node stereotype `sensor`. If complex networks that handle several components and artifacts need to be displayed, the stereotype of the `network` can be used for a node. Since the data processing speed and its creation plays a significant role, the connection of the elements needs to be adjusted. This can be done through additional connection information `data_stream`.

In order to simplify the modeling process, facilitate a possible way to share and change created diagrams easily, and enable updates of all created diagrams automatically without manually modifying them, an automated procedure was chosen that creates a BDA model out of an existing JSON configuration file. The reason for this is manifold. At first, sometimes related architectures are getting very complex, especially when further parts of the infrastructure are considered in which the BDA is to be integrated. By using a manual *drag-and-drop* option, the setup of those might be error prone due to the size and complexity of the model as well as a potential lack of knowledge regarding their creation. In case elements are missing and in the future to be added, thorough revisions might be required.

To ensure that the model can be successfully created, a potential user is prompted to follow the predefined structure of an input file. By using a self-developed validator, the structure is checked before the actual modeling takes place. This includes, for instance, verifying missing elements, parameters, or connections, identifying circulations, multiple occurrences of an element, and syntax errors. In general, the structure of the JSON file is comprised of the elements `general`, `settings`, `artifacts`, `nodes`, `components`, `actors`, `sub-systems`, and `connections`.

Within the first area, descriptive information for the overall document are created. Apart from a name, this includes, inter alia, a short description, and a version of the model. The settings element describes basic characteristics for the desired diagram, such as the alignment of the elements. For a more appealing visualization, the tag `leftToRight` was defined to either build the model in a horizontal or vertical direction. This shall ensure that even though numerous elements are included, still, good readability is achieved. Additionally, a magnifying glass is implemented with which the view can be zoomed in and zoomed out. An excerpt from this JSON file is depicted in Figure 4.18.

All other elements are used as described before. `Actors` represent stakeholders that are directly interacting with the system. Those are symbolized by a person. Relevant system elements are described through nodes as either separate entities or children of a sub-system. If concrete information about the functionalities and defined interfaces needs to be displayed, components can be used.

```
{
 general: {
   id: "Diagram",
   title: "Architecture Towards an Architecture for Big Data-
      Driven Knowledge Management Systems",
   description: "This is an exemplarily deployment diagram of
       the architecture described in the paper.",
  version: "1",
  creation: "30.02.2022"
    },
  settings: {
    leftToRight: true
  },
  artifacts: [

  ],
  nodes: [
    {
      id: "ndrtdd",
      name: "Real Time Data Device",
      type: "DEVICE"
    },
    {
      id: "ndbs",
      name: "Batch Server",
      type: "DEVICE",
      children: ["nebde"]
    },
    ....
subsystems: [
    {
      id: "sds",
      name: "Data Sources",
      children: ["ndrtdd","ndbs"]
    },
    {
      id: "sis",
      name: "Data Information as a Service",
      children: ["ndrtps","ndks","ndbps"]
    }
  ],
connections: [
  {
    senderid: "nebde",
    targetid: "nedl",
    type: "ACCESS_BOTH"
  },
  ....
  ]
}
```

Figure 4.18: Exemplarily JSON configuration file

The definition of each of those, except the `settings` and `connections`, needs to be performed by a unique ID and an expressive name. The technical implementation of the prototype is realized by a client-server architecture using Java as a runtime environment, Vaadin as a web interface, and PlantUML as the modeling tool. After a thorough comparison of various modeling tools, such as Modelio, Papyrus, BOUML, Eclipse UML2, and PlantUML, only the latter turned out to be a promising solution for the pursued task. Criteria used for the comparison were, for instance, the support of existing UML diagrams, supported programming languages, diagram layouting, frequent updates, exemplarily code fragments, and comprehensive documentation.

The visual representation of this prototype is depicted in Figure 4.19, which includes the aforementioned changes to the existing diagrams, the JSON input field, and the field for the compiled model. Noteworthy, auxiliary functions were created that allow to open, save, and create diagrams. The export, in this regard, can be either done as a dedicated JSON or PDF file. While the first contains the raw document, the second depicts the compiled diagram.

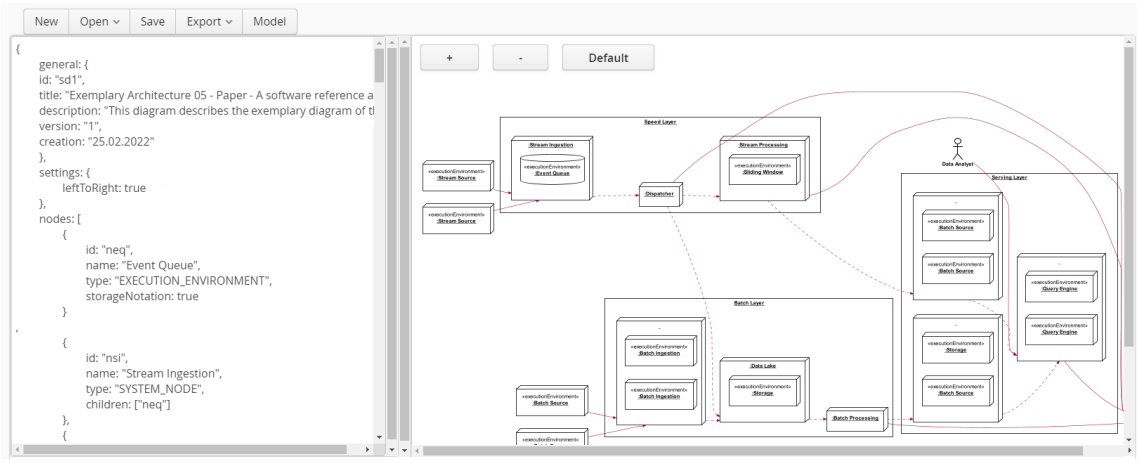


Figure 4.19: The developed prototype of the BDA modeling tool

4.5.3 Evaluation of the Big Data System Architecture Modeling Approach

Similar to the artifacts created before, another time, in accordance to the applied research methodologies [HMP+04; PRT+12; SV12a; SV12b], the proposed artifact was continuously evaluated. The starting point is formed by architectures previously identified during the literature review and mostly introduced self-defined notation elements. In particular, the following contributions were exemplarily selected for the evaluation [LPB16; KL14; GHK+18; NHR+17; BD18; Gee13; UPA+15]. Each of them was reconstructed by following the proposed modeling approach. First, the needed input information was extracted from the existing model and the corresponding text section for every single architecture. Secondly, the configuration was manually created and afterward imported into the developed prototype. Due to the huge number of elements and size of the final model, only

three of those architectures [LPB16; BD18; KL14] are depicted in the Appendix (cf. Figure A.1, Figure A.2 and Figure A.3), using the `leftToRight` command. Although it was once more noticed that most of the architectures did not explicit any concrete insights about the connected parts and specific tools to be used, it was possible to (re-) model each of them without discarding system architecture-relevant parts. Hence, the functionality of the developed modeling approach and the constructed prototype was successfully evaluated. Furthermore, in addition to the existing documentation of the developed solution, it was noticed that a future reuse might be sensible. As a further extension, those were integrated into a later stage for the prototypical implementation of the main artifact.

4.6 Automatic Deployment of Big Data Technologies

With the advent of new trends, such as social media, IoT, and other data-intensive application scenarios, the necessity to handle those became ubiquitous [DL20a]. As a result, apart from new technologies, sophisticated architectural considerations are required that provide a scalable and robust framework for the current and future development. Through the mutual influences of both aspects, the complexity of the engineering of the related systems is steadily increasing [Lee17]. This applies not only for the selection of those, as described before, but also for the actual deployment. Eventually, this complicates the situation for inexperienced researchers and practitioners, such as to plan, conceptualize, and test related systems.

Despite the fact that all-encompassing platform solutions exist, delivering the convenience of having everything in *one place*, not only the use of the platform-specific technologies themselves but also the cost estimation of their usage can be sometimes cumbersome. The combined use of multiple technologies and services in parallel may be challenging. Sometimes, it can even be prevented due to a lack of existing connectors or interfaces. Ultimately, this could result in high monetary expenditures, extensive knowledge required to handle big data projects, and a potential vendor lock-in effect. The user is forced to stick to the services offered by the provider solely. Especially the latter was noted for many different providers in several research studies, such as [AAA+20; HPK20; CPC+20].

In contrast, free and open-source solutions like the previously discussed DICE framework or Apache BigTop (cf. section 3.1.2) and further options, like pre-packaged images by Cloudera Hortonworks project, partially attempt to overcome the referred problems. However, in many cases, these are limited in terms of their applicability. Particularly, Cloudera offers their distribution as a sandbox that comes with an extensive collection of big data technologies, but in many cases, they exceed what the user needs, which, in turn, results in the necessity for a potentially complicated and cumbersome customization and configuration. The limitations in terms of a potential modification were likewise ascertained regarding DICE and BigTop. While the first is currently deprecated and the project not further conducted, the second can hardly be modified in terms of new technologies and

their combination due to the tool's background processes and inner mechanics. Instead, the focus here is put on specific technologies provided by Apache. Generally speaking, there is no opportunity to extend or reduce the setup to the required technologies for a targeted solution.

By keeping the flux of big data and the constantly emerging and changing technologies in mind, potential users should instead be confronted with a lightweight and adjustable approach that relies on open-source technologies, allowing an automated system architecture deployment even in non-cloud, limited resource environments. A potential concept is proposed, described, and evaluated in the following section of this work. The artifact created in the context of this section was published in [VSI+22]. Apart from container and configuration management technologies used for the creation application, also the previously introduced BDTOnto [VSJ+20] is utilized.

4.6.1 Preliminary Considerations

For the identification of relevant research articles, which incorporate deployment technologies in the field of big data, a thorough literature review was conducted. Again, this procedure was performed using an initial keyword-based search, as well as a subsequent forward and backward search [LJ06; WR02]. At first, an initial material collection was performed using the databases IEEE, ScienceDirect, Scopus, and CiteSeerx, and the search term: *"big data" AND (architecture OR application) AND (DevOps or deployment) AND (strategy OR framework OR practice OR method OR survey)"*. Another time, the title, abstract, and keywords were searched, and several inclusion and exclusion criteria were recognized throughout the process. Here the focus was put on deployments, open-source technologies, precise concept details, and the publication years. In particular, no papers before the early hype were recognized, including only literature published after 2014 (cf. section 2.2. Eventually, eight articles were deemed relevant, which are shortly described here, to highlight the most important aspects of this sub-chapter's targeted solution.

An approach to how Hadoop clusters can be deployed in the cloud is discussed in the article of [FRM15]. Their deployment discussion notes that Hadoop was not designed to be deployed in virtual machines (VMs) as it expects data and compute nodes to coexist. There is also no concept of elasticity. A potential solution for the deployment of Hadoop workers in the cloud was given in [TWW+16]. Here, the authors highlight that the computing VMs are typically running entire operating systems (OSs) in cloud environments. However, the hypervisor of VMs, in charge of the resource allocation, often degrades the performance of the virtual OS. To overcome this issue, an approach is proposed that harnesses the capabilities of Docker. Wu et al. [WCC+14] introduce in their work the *YZStack* architecture, where big data tools are implemented in separated layers. The deployment of those is performed using an *adaptive image*. The infrastructure layer pre-generates a virtual server image that includes the OS and minimum required modules

commonly used. The intended big data tools are then built onto these images, with all configurations happening in an ad-hoc manner.

In the work of Higgins et al. [HAH18], an automated deployment model for high-performance clusters (HPC) is described. The presented solution focuses on the complexity of deployment automation and configuration management. Especially container technologies are highlighted here as the key technology for HPC cases. Specifically, Ansible was mentioned as one of the most important deployment automation engines. This is because, inter alia, common standards such as secure shell (SSH) connections are used and no dedicated daemons on each node are required, effectively reducing the overall overhead. Apart from that, all configurations can be done using *Yet Another Markup Language* (YAML) files. The authors highlight that configurations for deploying a component can be abstracted into roles, which consist of several tasks [HAH18].

Generally speaking, containers are virtualized, lightweight OS processes that provide portable runtime environments independent of the underlying hardware [SGM18]; [Hoc15, p. 237]. Hence, they help to deal with issues like conflicting and missing dependencies as well as platform specifics [Mer14]. In this regard, one frequently used tool is Docker, which offers a user-friendly *application programming interface* (API) that is unified across different platforms. It uses namespaces to completely isolate the application's view of the underlying OS and environment, including process trees, network, user IDs, and file systems [Hoc15, 237–239]. In turn, Ansible is an automation engine that can be used for configuration management, application deployment, intra-service orchestration, and other needs. When utilized, it connects different nodes with each other and runs small programs called *Ansible Modules*. These programs perform various actions on the host and bring them into the system's desired state. Ansible then executes these modules and removes them once the task is completed. Compared to similar tools, such as Puppet or Chef, Ansible is efficient, lean, and does not require an active server, daemon, or database to run specific modules or keep states. The flexibility of running a task or role on specific nodes using *inventories* in Ansible offers complete convenience and freedom for system administrators to define and maintain re-usable scripts, called *playbooks*. The latter can be used to take a node into the desired state for a specific package or technology. Updates and changes are distributed via a push-based approach. Once a playbook changes, the related modules are immediately informed, which differs from the pull-based system where a central service is periodically requested for updates. The latter often applies to tools, such as Puppet or Chef [Hoc15, 1–34].

Docker images were also of major interest in other research articles, such as in [TPK+16]. Within this article, the authors propose a deployment method based on a general docker workflow, where individual components are packaged into Docker images and deployed in container engines as necessary. Beyond that, they highlight the benefits of using this approach compared to classic VMs. Morabito et al. [MKK15] performed a comparison between the hypervisor and container-based virtualization technologies. In doing so, vari-

ous strengths and weaknesses of each type were highlighted. The work presented by Felter et al. [FFR+15] denotes another comparison of virtual machines and *Linux containers*. In particular, a kernel-based virtual machine (KVM) as the hypervisor and Docker as the container engine were used. They came to similar conclusions as Morabito et al. [MKK15] and concluded that both solutions generally achieved a mature status. However, container deployments using Docker still outperform the KVM deployments in terms of all tested metrics. Nevertheless, both solutions have their advantages and disadvantages. Lastly, in [VCC+15], an approach to deploy large-scale datasets in cloud environments is presented. Their deployment automation is achieved using configuration management tools and a modified version of BitTorrent.

According to the given summaries, it becomes apparent that different approaches exist that attempt to provide suitable solutions. For instance, while in [WCC+14] a pre-packaged VM was used to offer a complete solution, another approach used configuration and management tools, such as Ansible, to allow automation for the configuration and deployments of various components [HAH18]. Despite the great acceptance of classical VM-based approaches, including not only the OS but also multiple preinstalled functionalities, it was noted that in many cases, container technologies outperformed those in a direct comparison [MKK15]. Thus, Docker and Ansible appeared to be desirable solutions for further deployment and configuration management.

4.6.2 A Framework for the Automated Deployment

After investigating the existing approaches as well as the recommended container and automation tools from the literature review in more detail, general steps were identified that are required for the setup of a potential solution. Related deployment and management nodes need to be created initially, resulting from the interplay between Docker and Ansible. While the former is used to deploy the targeted technologies and architectures, the latter manages and handles all required implementations. After that, if available registries like DockerHub [Doc22] do not already provide them, each technology component needs to be newly formed. In that case, a base image needs to be created and integrated into a registry, independent of its public accessibility, for each big data technology. Another option, especially for companies, can be the set-up of a personal registry at which new images are stored.

The same applies to Ansible. As the deployment management framework it converts manual process steps into automated, small scripted ones, called playbooks, for the automation to either deploy the referred images individually or in combination. However, when those are built automatically, an intermediate mapping element is required, capable of connecting relevant Docker images and required information to create the playbooks, including name, version, and the Docker registry reference for each big data technology. This is not related to the inventory file, which is used for the definition of the hosts [Hoc15,

45–46]. Preliminary research to sufficiently store, manage, and handle big data-related information was presented in section 4.3, in which the BDTOnto was introduced. Eventually, this ontology was used for the planned framework. Although not directly related to big data, in [MHK+15] the suitability of such a combination of an ontology and DevOps for composable architecture is also stated, reinforcing the sensibility for this approach.

By considering all of the information mentioned above, a hybrid framework was derived, where big data components, their combination, configuration, management, and deployment are prepared via machine-readable format files to achieve increased automation port- and reusability. Compared to other existing solutions, the focus is put on non-commercial, low complexity, resource conservative, and easily extendable elements. For an improved (re-) usability, a sophisticated concept is utilized that allows potential users to discover, identify, and keep track of interdependencies between single big data technologies as well as complex architectures. An overview of this concept is shown in Figure 4.20.

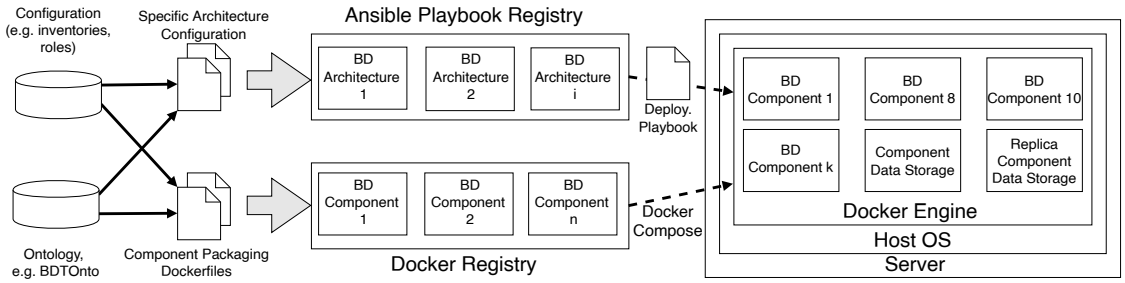


Figure 4.20: Depiction of the big data technology deployment automation framework [VSI+22]

All required information relevant for the general understanding of the technologies and their relation to each other, is stored in the ontology. This includes the respective reference to the related Docker files and playbooks that deal with single technologies in this context. If a container for a specific technology is neither formed nor available in openly accessible registries, an initial creation must be performed. As mentioned before, once a container image is available, it can be persisted and distributed for later reuse through a private or public Docker registry, such as DockerHub. There, a multitude of publicly available big data implementations is already provided. Generally, an adhering deployment of single components and generic architectures can already be performed through docker-compose files, created using YAML.

However, different tasks are required for multiple components in combination, as it is typical for related projects and systems. This includes, for instance, the structuring, copying, managing, or changing of configurations, regarding both the aimed destination and the component interaction. The interplay is automated through the use of Ansible and logically structured by utilizing the *role* concept. These roles, in turn, can be used within playbooks. All used images of relevant big data technology components need to

be either built prior or directly pulled from an existing registry. The persistency and linkage of the created playbooks can be achieved for complex architectures in the same way as the Docker components, through a registry and the used ontology. To reduce the effort of manual settings and frequent interactions during the deployment process, various configuration information is required within the playbooks, such as the specific endpoints. This information needs to be declared in *inventory* files. After successfully creating a playbook, the deployment of the BDA can be executed. Ansible autonomously performs all steps required for the deployment to the desired host in sequential order. The inventory files are stored in separate data storage for multi-user management and user-specific endpoint declarations.

Once the setup is created, even inexperienced users can easily set up their intended architectures using this approach. By developing an additional computer-supported solution that incorporates this concept, the effort of realizing everything could be reduced to only *a few clicks*. The interconnection with the ontology requires no all-encompassing knowledge in all domains. As a benefit, prospective big data technology users can easily include and use their desired technologies in the ontology, facilitating their automated deployment at a low cost. The fast creation of system architectures in a (semi-) automatized way is also a principle of modern paradigms like DevOps, which is composed of the two terms development and operations [GG17] and follows the building, testing, and deploying principle, known from continuous software engineering [SAZ17]. Potential users of the concept may take advantage of this and create sandbox environments for their potential architectures, open to migrating them to further environments, enabling additional integrations.

4.6.3 Evaluation of the Developed Automatic Deployment Approach

In correspondence to the followed methodology of this work, an evaluation of the proposed concept was realized by an experimental implementation and application of a potential BDA. The ascertained applicability and complexity were afterward compared to the previously described DICE framework [CL18] and Apache BigTop [Apa] using various criteria. As one of the most prominent approaches, the Kappa architecture was used and tested [Kre14]. In doing so, an openly accessible dataset of taxi trips in New York City [NYC18] was used.

Compared to the two systems required in the Lambda architecture, described in section 2.2.5, the Kappa architecture requires only a stream processing system through which the data is incoming and transformed. Afterward, everything is stored within an analytical database [Kre14]. As the heart of the system, the streaming layer is constituted by a messaging system. For this purpose, the recommended tool Apache Kafka was chosen. The same applies to the further technologies frequently used in the context of this architecture. In particular, for the data storage Apache Cassandra, as the serving layer, as well as Apache ZooKeeper, for cluster state management, were used [Kre14]. An overview of

the architecture can be seen in Figure 4.21.

For the implementation of the architecture, a multi-node setup was followed, for which the preparation of the management and deployment nodes were required. On both machines, Docker is needed as well as for the management node an Ansible instance. Hence, two VMs are used with Ubuntu Linux 18.04.1 LTS as the distribution. The first VM contains the processing layer and cluster management, represented by Kafka and Zookeeper, along with the evaluation data as the test workload. A Cassandra cluster and the stream processors were deployed in the second VM. Both VMs were connected using an overlay Docker network that can be accessed from both nodes, facilitating communication between them.

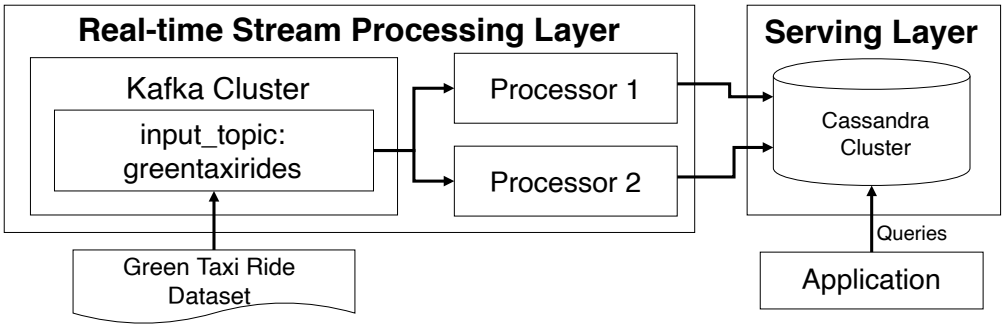


Figure 4.21: Exemplarily deployment diagram using the Kappa architecture [VSI+22]

After finishing the overall preparatory steps, each component was prepared and defined in an independent cluster configuration for the architecture deployment, using a docker-compose file. The initially required information about general relations and dependencies of the architecture as well as the specific technologies were already included within the used BDTOnto. However, further extensions were performed after the successful deployment by using additional classes, data, and object properties. This includes general implementation information, such as the specific runtime environment and also the linkage to the related files for the deployment.

After the Docker Compose files were created for each component, those were used in Ansible. In particular, different tasks were defined and combined into roles that dealt with the setup of the specific component. Then, each role was put together into one single playbook that is in charge of the automated deployment of the architecture. The needed inventory information for the used machines was defined in the user-specific inventory file.

Thereupon, the deployed architecture was successfully tested, using the exemplarily dataset as well as some simple data analysis methods on the dataset mentioned above. Eventually, all created files were linked to the related big data technology classes within the ontology. The same applies to the user-specific information in the user data storage regarding the connection endpoints of the used machines. Through the use of a computer-supported solution that gathers all required files and information, the deployment is afterward automated and made executable via *one-click*.

By considering all of the noted aspects, multiple benefits become visible by comparing the proposed concept with related approaches from section 3.1.2. Namely, the DICE framework [CL18] and Apache BigTop [Apa]. This was done by comparing quality attributes relevant to the targeted goal of developing a *convenient and low complexity solution, where individual components can be offered as re-usable and re-configurable packages.* To do so, the usability, portability, reproducibility, resource requirements, reusability, and flexibility were thoroughly investigated.

Further insights and detailed comparisons, especially in terms of the performed implementation, can be found in the related article [VSI+22]. Notwithstanding that, an overview of the comparison results is depicted in Table 4.15, in which the severity ranges from low to high. If a specific attribute is not comparable, the term *limited* is used to highlight the limitations and restrictions of a tool.

Attribute	**Developed Solution**	**Apache BigTop**	**DICE Framework**
Usability	High	Moderate	Moderate
Portability	High	Moderate	Moderate
Reproduceability	High	Moderate	Moderate
Resource Consumption	Low	Low	Low
Reuse	High	Limited	Low
Flexibility	High	Limited	Limited

Table 4.15: Comparison of the deployment concept to known tools, based on [VSI+22]

Concerning *flexibility*, for instance, each solution was investigated and compared by examining the modifiability and extendibility. Regarding the developed concept, the user can easily add or remove new components and seamlessly integrate completely new technologies without any changes to the core. Primarily through a self-creation or use of open access repositories, thorough extensions are imaginable in a short time (e.g., Docker repository). BigTop, on the other hand, only allows for building and installing specific container images. As a result, the user can only use components the team behind the tool provides.

DICE, in turn, does not intend to offer further big data technologies since the tools, configurations, and functionalities are very complex and tailored for each of them. Combined with discontinued development, this circumstance acts as the greatest counterargument for a potential application.

While both of the investigated approaches deliver numerous additional functionalities, configurations, and supplementary material, it was observed that the created solution outperforms both of them in terms of the used criteria. Depending on the skillset and

expertise of the user that uses the big data technologies, only basic knowledge is required. Nevertheless, additional effort needs to be put into it for further configuration management and other specifications. For now, stress testing or the setup of a *turnkey-ready* solution may only be feasible in a limited way. However, with the combination and use of the well-known open-source technologies Docker as well as Ansible, an integration in cloud environments, such as the Google Cloud Platform (GCP), is imaginable without fearing a potential vendor lock-in effect.

4.7 The Decision Support for Big Data Projects (DECIDE) Framework

Conceptual models serve in systems engineering as a helpful way to outline the relationships between components and their use in a technology-independent way. In particular, they appear to be useful for newly developed systems in order to have a framework for their interaction concerning the functions, underlying processes, and the data to be processed (cf. section 2.1. Based on these insights, the decision support for big data projects (DECIDE) framework is introduced below in Figure 4.22, consisting of the components described above and their specificities. Although the given model does not use a well-known modeling language and thus contradicts the implicitly conveyed idea from section 4.5, using established approaches for modeling, a comprehensive description of its structure and usability shall be given.

First and foremost, the model is composed of different levels covering all aspects as previously described and presented in Table 3.6. At the top level, all essential components are identified. Namely, these are *requirements engineering*, *reasonability check*, *system design*, *architecture modeling*, and *system deployment*. Although all components are aligned to correspond to the overarching process shown in Figure 3.2, they can also be used in isolation for other considerations and purposes. The main intention is to address the specific problem regarding a successful project execution and to support the area of big data in general. This, in turn, is in line with the idea of component discovery as discussed before (cf. section 3.4). In the event of comprehensive decision support from end-to-end, it is not only necessary to have at least a general idea about the planned endeavor or, more precisely, the project. Depending on which steps are taken, the decision support can be manifold, focusing on single components and their combination. In the case of process execution, decisions made in this way can be validated and verified, as it is commonly required for SE [IEE12] and decision making (cf. Figure 2.14) activities. The expenditure for implementing possible changes is comparatively low since it concerns only decision support and does not affect any productive systems. Nevertheless, further tests are necessary before and after an actual integration (cf. section 2.1).

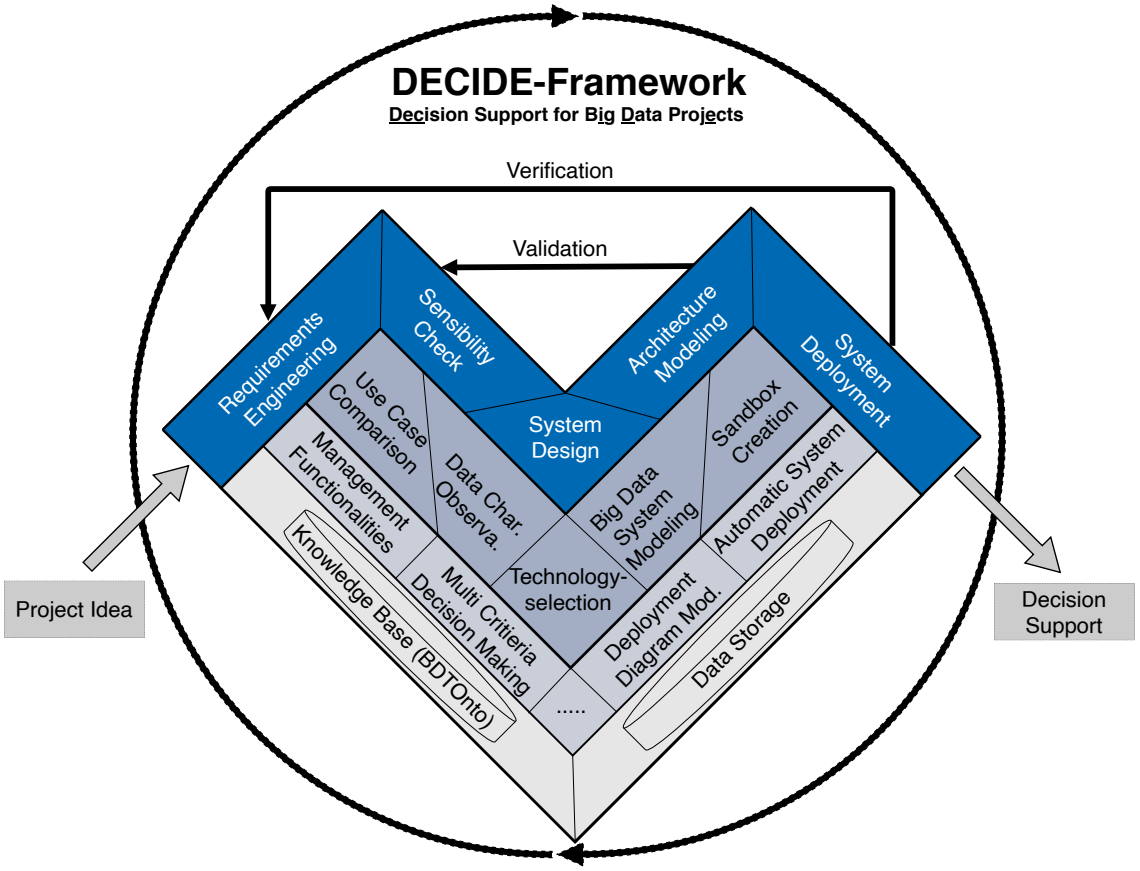


Figure 4.22: The DECIDE framework

All considerations and operations necessary, thereby, must be accomplished around these components that require extensive knowledge and various supporting functions. These additional measures are depicted in the second level of the DECIDE framework. Besides the *use case comparison*, *data characteristics analysis*, *technology selection*, *big data system modeling*, this includes the *sandbox creation*. Once more, for each layer element, an encapsulation of the functionalities is aspired to use these detached, depending upon the intended use.

The same applies to the detailed algorithms and used concepts listed on the third level, as they were also addressed in the work. These fine-grained aspects are essential for achieving the higher-level goal and do not need to be broken down further. Here, another time, these contribute either solely or in combination to the activities on the level above. For instance, for the technology selection, the identification of basic functionalities and an MCDM algorithm are required. In this context, at least the depicted functions should be included for a prototypical implementation (cf. Table 3.2). Auxiliary ones can be added, as the dotted box displayed in the middle of the figure. This may include functionalities originating from Table 3.3, such as multi-tenant support, role concept, or technology representations.

Finally, the last level represents the data basis. In addition to general data used for storing user or other system-relevant information, the knowledge basis, in particular, plays an essential role for the successful operation. Despite the fact that the exact specification here is already named in the framework, with BDTOnto, this leaves a considerable degree of flexibility for changes and extensions. In the course of this work, the BDTOnto was discussed several times implicitly and explicitly. Almost all elements of the layers above harness the knowledge to some extent and interact with it. Nevertheless, it is to be mentioned here that this can also be replaced in case an equal or better solution may emerge, covering all of the required information comprehensively.

As can be easily seen from the sub-chapters described above, all the levels and their components described here have been successively derived and inserted. The superordinate components were subdivided and decomposed piece by piece from the general to the special. Although the illustrated framework and the associated description give an extensive overview, the sub-chapters described before are essential for an implementation into a DSS. However, further detailed information can be found again in the respective contributions that this chapter is mainly composed of [VHB+16; VJT17; VPT18; VSP+20; VST+20; VSJ+20; VSI+22; VST21].

4.8 Summary

In this chapter, the components identified in the previous chapter were successively examined scientifically. As could be found beforehand, numerous ideas and possible approaches exist. However, these provided little to no information, were unsuitable, or did not permit direct employment in the sense of component-based use. Due to this, extensive investigations were performed for each of these, and comprehensive descriptions were given, usually containing a short excerpt from the existing literature, preliminary consideration, and concrete development steps. In the sense of the DSR, which was the method of choice for the development of each of the components, extensive evaluations were conducted. Finally, the obtained findings were mapped in the DECIDE framework, which depicts, in addition to a general structured process for the general acquisition of a comprehensive decision support for big data projects, a component-based structure of a computer-aided solution. On different levels, *knowledge*, as identified in the course of the work in general and the current chapter in particular, was compiled. While individual results were partly prototyped and successfully tested, it still requires a comprehensive prototype, covering both the components and the supporting functionalities of the end-to-end process. Thus, the fourth chapter did not only supply an answer to the second and third SRQ but also a first framework for an implementation concept, as it is required for SRQ4. Accordingly, an overview will now be given in the following fifth chapter.

5 Prototypical Implementation

After all scientific preparatory work for the production of the suitable components for a DSS could be attained and a comprehensive framework created, the presentation of the prototype takes place in the current chapter. Similar to SE, according to [SV12b], the development of a prototype builds an essential activity in the evaluation of the DSR. Hence, through that activity, in the spirit of the already conducted research, another foundation for the performed evaluation in chapter 6 is completed. Besides the already mentioned components here, all other requirements identified in advance are to be implemented (see section 3.2). According to the used research methods, the created system represents not only an instantiation of the developed framework [SV12b] but, at the same time, another component of the answer to the fourth SRQ. Consequently, this chapter describes the structure and defines the utilization of the prototypical DSS. To begin with, the first subsection provides essential information on the design and the selected basic technologies for implementation. Many of the statements and findings made there are based in particular on the preliminary work [VSB+20b]. Following this, an overview of the developed GUI will be given in section 5.2.Based on the structure of a DSS, the data basis and integration will be described in more detail.

As shown on the lowest level of the DECIDE framework, this includes the data store and the BDTOnto. The various components and their implementation are presented in the eponymous section 5.4. Sometimes these are also referred to as tools, in the context of this prototype. All of them are described successively in the same order as their scientific investigation. Additional functionalities, simplifying the interaction, use, and modification of the system and related information about the domain of big data, are presented in the second last subsection, 5.5. Finally, the realization of the proposed *end-to-end* procedure from Figure 3.2 is showcased. During the description of each implemented component, code fragments and sometimes automatically created class diagrams are shown. For the creation of the documentation, Doxygen was used, which coins itself as *"the de facto standard tool for generating documentation from annotated C++ sources"* [Dox22] that can also be used for several other programming languages, such as Java, Phyton and C#. Due to this size of more than 300 pages, the documentation is provided digitally. The same applies for the source code.

5.1 Structure of the System

To realize an initial prototype, the actual system specifications, and the architecture are first required. Commonly, DSSs follow a similar structure, dividing the system components into a GUI, an inference engine, and a knowledge base (cf. section 2.3). Based on the principles and ideas shared by [Tur03], the components and their loosely coupled connection constitute the foundation. All further details were identified by considering the defining characteristics of a DSS as they were found and discussed prior in Table 2.4. Consequently, all of these were incorporated to develop the overarching system architecture. Based on those, preliminary considerations were made regarding each of the characteristics a DSS may fulfill:

- **Characteristic 1 – Semi-structured or unstructured problems:** The system either provides the standalone use of the different components or the compound application in the form of the end-to-end procedure. At each point in time, the user is free to perform changes in each component, even though some default values are set, such as when the results of a SUC comparison are not all relevant.
- **Characteristic 2 – Support managers at all levels:** No restrictions in terms of a profession or level of knowledge are made. Independent from the position, access is granted to everyone, as long as this is performed by the administrator (login). Consequentially, the information density at each point is kept high.
- **Characteristic 3 – Support individuals and groups:** Decisions can be made collaboratively. No restrictions are made that force the isolated use by only one user. By its very nature, in the form of a web-based solution, the screen could be shared with colleagues independent of their geographical location.
- **Characteristic 4 – Interdependent or sequential decisions:** The component-based setup and the freedom to use them in an isolated or compound way facilitate that decisions can be made independently or in sequential order. This is additionally supported by saving and loading functionalities in each of those.
- **Characteristic 5 – Support intelligence, design, choice, and implementation:** The entire process is supported, which is already constituted by the components themselves.
- **Characteristic 6 – Support a variety of decision processes and styles:** The user is not forced to stick to the end-to-end procedure. The sequential order can be altered based on preferences and certain activities can be added or removed.
- **Characteristic 7 – Adaptable and flexible:** The prototype provides a lean and intuitive structure. Furthermore, a documentation, commented code, and readme files provide further information for adoption and extension and, thus, guide the developer to alter, extend, add, or delete any kind of functions or elements. The system uses supportive libraries, a sophisticated web-development framework, and an easy-to-use programming language.

- **Characteristic 8 – Interactive, ease of use:** A responsive, web-based interface is provided that provides a rich GUI with which the user can interact. All information is supplied in different forms, computer and human-readable.
- **Characteristic 9 – Effectiveness and efficiency:** Depending on the available information and time by the user, the decision support procedure can be differently structured. Different values can be determined individually, or default values are used.
- **Characteristic 10 – Humans control the process:** The user can manipulate all required inputs based on their own preferences. Before the decision support recommendation is calculated and given, the user receives an overview to check all made inputs once again.
- **Characteristic 11 – Ease of development by end users:** Basic programming knowledge, as well as the shared information from the thesis, documentation, and readme files, support the overall future ability to continue with the development. The setup of the prototype, the chosen programming language, and used framework(s) are kept lean and not overcomplicated. Additionally, only frameworks and libraries with a thorough documentation and a strong community are chosen.
- **Characteristic 12 – Modeling and analysis:** The modeling and deployment component can create various models and experimental setups.
- **Characteristic 13 – Data access:** The user receives complete access to all relevant information to use, maintain, and extend the DSS. This includes, for instance, generic textboxes, related research contributions, the created documentation, readme files, and the BDTOnto.
- **Characteristics 14 – Standalone integration and web-based:** The system itself can be either used locally or accessed through a server-client connection when distributed by the organization by harnessing web technologies.

Based on preparatory considerations, the programming language was selected first. As one of the best known and most popular languages, Java was chosen in the course of this [Hac20]. All other decisions were then influenced by this decision. This includes, first and foremost, the web framework. After some generic comparisons, Vaadin was selected as a suitable free and open source web application development platform with a strong community that offers numerous benefits [Vaa22]. For instance, although a visual designer exists, pure Java code can be used for programming and design in contrast to other web development frameworks. Hence, it neglects the necessity to create time-consuming layout files. However, through the broad and active community, a multitude of various plugins, templates, and other extensions exist that allow the development of sophisticated web applications. The same applies to the company itself, which continuously delivers updates, new releases, guidelines, and materials to modernize the developed solutions. To handle the used ontology, especially with regard to the active use during the decision support, another thorough comparison of existing frameworks was performed. In addition to Protégé's

own solution, Apache Jena and Stanbol were considered along with this. The focus of this comparison was placed on the support of the ontology, functional capabilities, and the offered long-term support. Due to the dominance in terms of the last two comparison criteria, Apache Jena [Apa22b] asserted itself as suitable for the planned implementation. As a free and open source Java framework for semantic web applications, it facilitates the utilization of the BDTOnto in the prototype by providing an ontology API. For storing differently structured data that are not already finalized at the beginning, a NoSQL database was chosen. For this prototype, a document-oriented database was selected that does not have a fixed structure in terms of the collections and single documents. Particularly, the free version of MongoDB [Mon22] was chosen. Considering all of the previous deliberations, tools selections, and investigated components, a potential system architecture was created, as shown in the deployment diagram in Figure 5.1. The overarching system, described as the `DSSServer` contains a Java application server required to set up the Vaadin application. All of the components are listed within this solution, except for the defined knowledge base. The BDTOnto is separately used and integrated using Apache Jena. Attached to this is the sandbox environment, used for the deployment of the big data systems and the user client that interacts via web browser with the application. Due to the limitations of the utilized diagram, additional functionalities, such as the user technology viewer or user management that also includes the authentication to the system, are not depicted here.

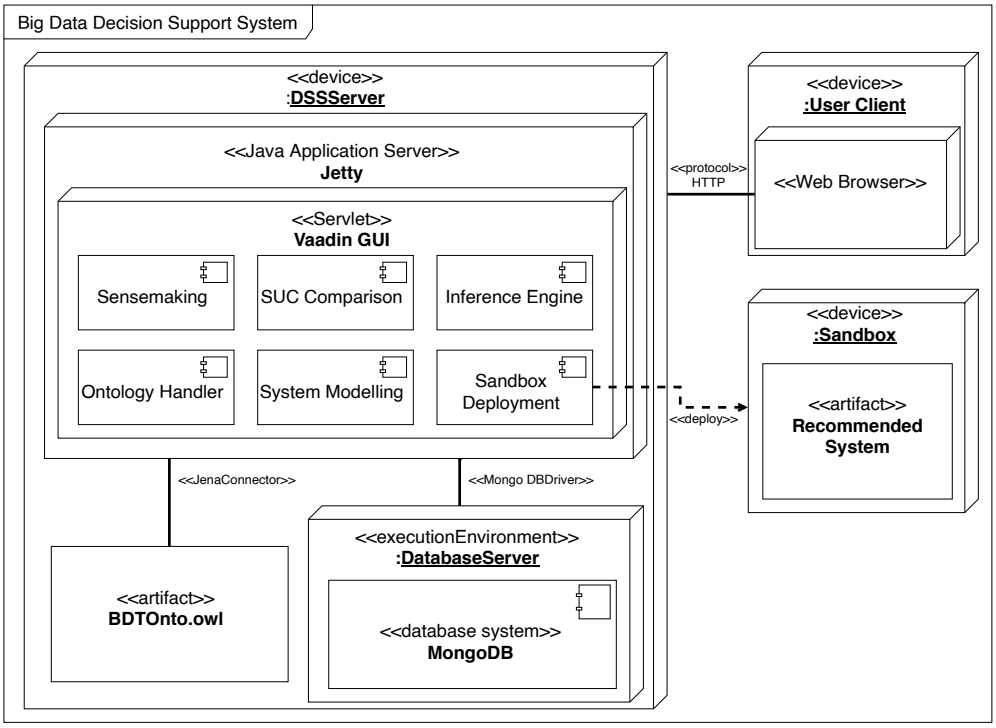


Figure 5.1: Deployment diagram of the component-based system architecture

5.2 Graphical User Interface

The creation of the GUI is relatively easy using Vaadin, at least in terms of the overall design procedure. The strong focus on programming and thus the automatized creation of layouts during runtime does not require sophisticated web application and design experience. However, these can be beneficial when it comes to modifying standardized components. At this point, several tutorials, examples, and other materials, such as books and blog entries, exist that help with these tasks. Based on these observations, the development of the GUI was mostly done via code and used only minimally customized *elements*.

The navigation is kept as intuitive as possible and should resemble other familiar applications. Based on a menu and various entries, the user should have the option to use existing *tools* in the prototype. Within each of these, Vaadin will then work with the respective elements provided by the framework itself. The main layout essentially consists of four different components. Specifically, this includes `Header`, `MenuBar`, `ContentLayout`, and `Footer`, as shown in Figure 5.2. The latter and the header contain general information and contribute to the visual enhancement. The `MenuBar` includes all available tools and the status of the logged-in user (marked here at the outer edge). Depending on which tool is selected within the menu bar, the `ContentLayout` is adapted. All tools are described in the enum `DSSInternalTools`. Each tool gets an `ID`, `short title`, `full title`, `short description`, `description`, and `listed features`. Within the class, the respective text fields are marked only by unique IDs, which provide a direct reference to the fields within the language configuration files. These provide multi-language support. The form of each file is always similar and follows the schema `Messages_XX.properties`, where the `XX` defines international country codes as described by the ISO 639 [SIL22]. By default, the language of the prototype is set to English.

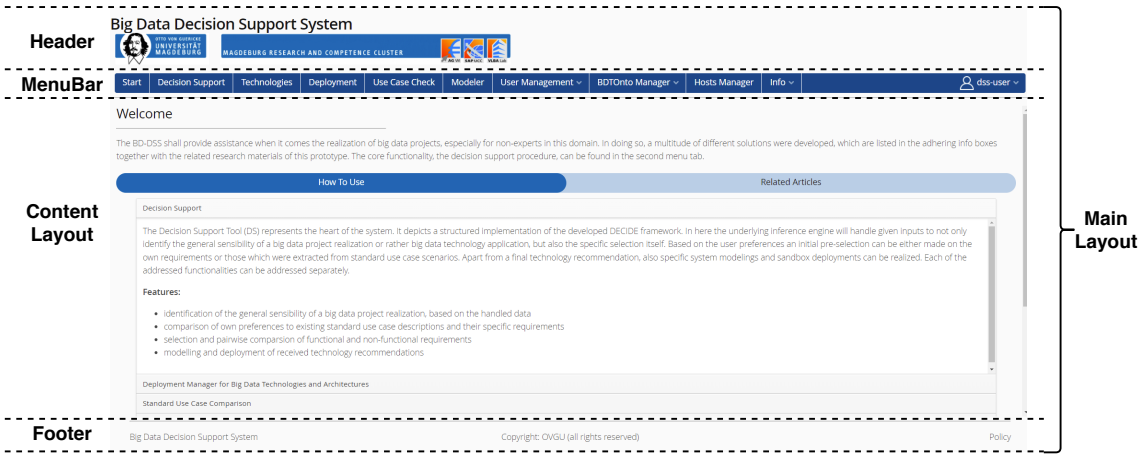


Figure 5.2: Welcome screen of the decision support system

In order to apply this language selection within the entire application and to change it at runtime, if necessary, the `TranslationsView` was created, which is used as the basis for almost every tool. The `getString(String id)` method contained in this class accesses the corresponding `Message_XX.properties` file and returns the description in the selected language, depending on the ID passed. Apart from these and a few exceptions, almost all elements originate from Vaadin's standard catalog. `Accordions` were often utilized for the representation of individual information and the structured implementation of individual steps, as will be described later, inter alia, in the component sub-sections 5.4.1, 5.4.2, and 5.4.3. All used colors or settings, not directly defined by the code, can be found in the Cascading Style Sheets (CSS) and Sassy CSS (SCSS) files in the `WebApp` folder.

As depicted in the figure shown above, the starting screen delivers initial insights and provides the user with the required information to interact with it. Besides a description used for almost all tools, an overview of the provided functionalities is given. In here, general information, as well as the features, are described. Furthermore, relevant papers published in the context of this prototype can be checked here. The documents themselves are located within the resource folder `related_papers`. The related enum for this is located in the model's folder, named `PublishedPrototypePaper`.

5.3 Data Storage and Knowledge Base

As described in advance, two essential elements are necessary for the prototype that deal with all data management processes. While the document-oriented MongoDB exclusively holds and manages data dealing with the users of the system, the BDTOnto represents the actual knowledge base. In the following, the integration of both will be described in more detail.

5.3.1 Implementation of the Data Storage using MongoDB

The document-oriented database MongoDB uses a flexible scheme for the individual documents that are held in collections. Especially during the long-term and prospective development of the prototype, this is a useful feature that allows extensions without any major complications. In this way, some functionalities were already tested in advance, such as the introduction of a role concept, which was not finally adopted for the prototype. However, the first attempts can already be found in one of the four collections, which are namely `users`, `uml-model`, `server-hosts`, and `git-ansible-files`.

The `users` collection currently manages all information of the users. A unique ID, user name, hashed password, and some security information is always specified. Further information can also be given depending on the specification, such as the email or telephone number. The security information defines the accessibility of individual tools by a user, because not everyone should have full access to all functionalities, not only not to confuse

them unnecessarily but also to prevent possible unwanted changes to the system, which would have an impact on others. Within the `uml-model` collection, all relevant information are stored. Apart from a the given ID defined in the settings (cf. Figure 4.18) this includes also the complete JSON file itself. The remaining collections are required for the handling of the deployment manager. While in the `server-hosts` collection the endpoint data are stored in separate documents, that can be created using the *host manager*, the `git-ansible-files` collection manages the required deployment files (cf. section 5.4.5. Apart from the name, origin of the deployment files, also the current version is implicitly stored via the ssh key. Notwithstanding that, an excerpt of the data model is depicted in Figure 5.3. As highlighted there, whenever new actions are performed, such as the modeling or new server-hosts created, the relevant data is stored using the current user.

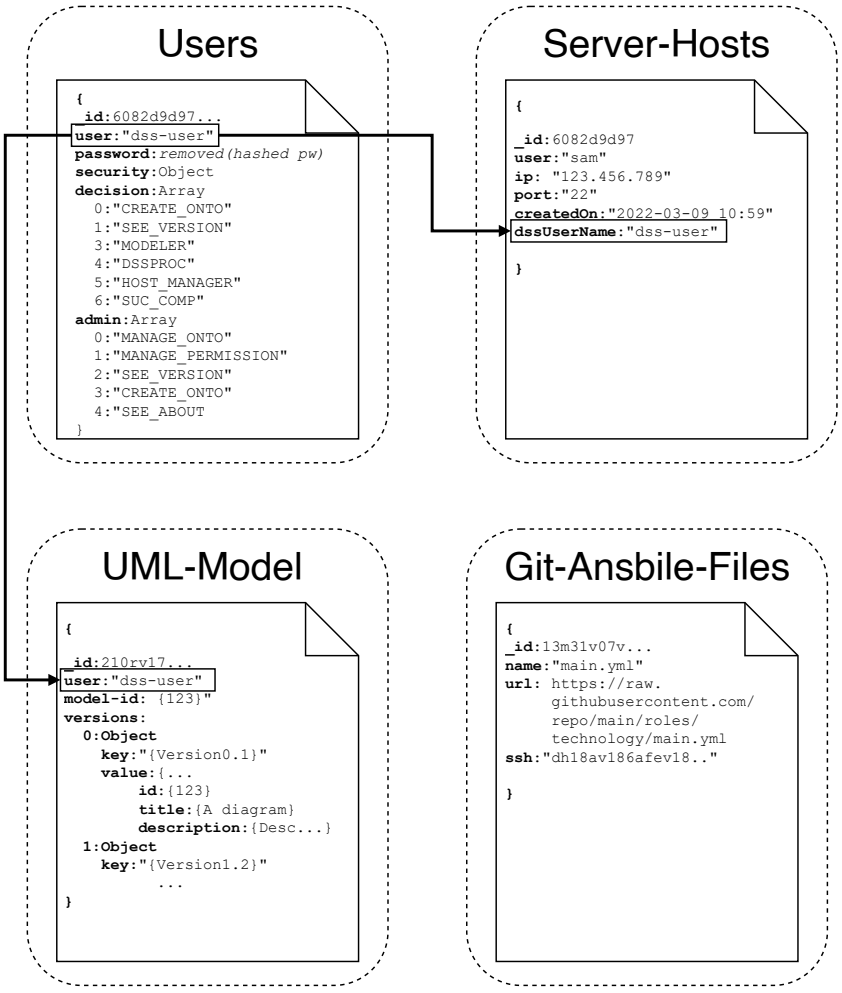


Figure 5.3: Overview of the data model using a document-oriented structure

To achieve an integration of theses within the system, the installation of the corresponding drivers of the database is a precondition for the creation of the system. Appropriate information, in addition, can also be found in the `README` file, which is digitally

provided. To create the basic structure and receive some initial data, the dump files located in the `mongo-dss/backup` directory should be used. If the respective fields are not available from the beginning, the `MongoDBManager` creates them. This is the main class that deals with all interactions with the database within the prototype. An overview of all functionalities can be found in the following class diagram, which is shown in Figure 5.4. Important at this point, relevant information for the connection is obtained from the `config.properties` file, located in the resources directory. This comprises the URL, port, and name of the database that are loaded with the `getConnection()` method. Similar to the multi-language support, configurations do not have to be hardcoded into the prototype.

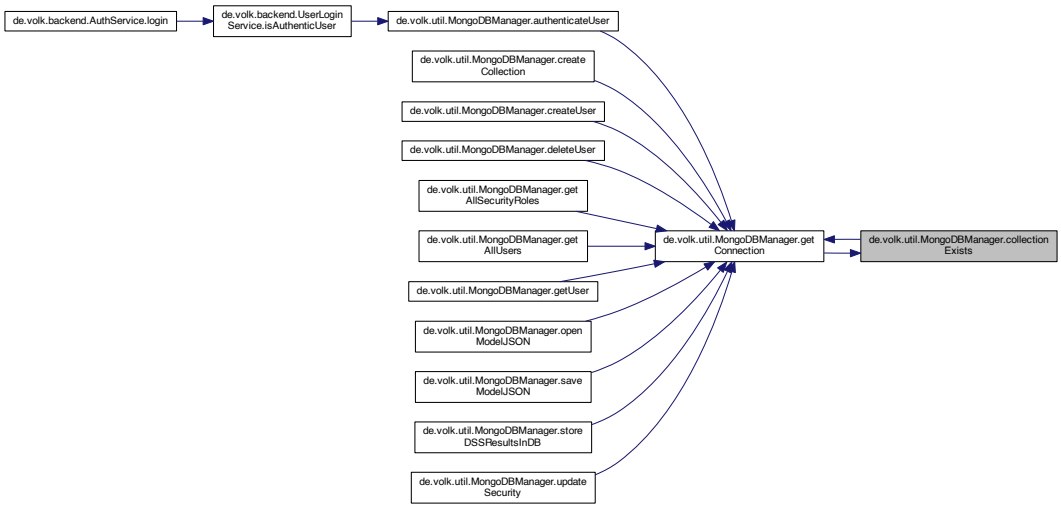


Figure 5.4: Overview of the data storage integration

5.3.2 Implementation of the BDTOnto

Other than for the date storage, no further standalone tools are mandatory for the usage of the ontology. Only for the editing and maintenance of the ontology Protégé is recommended, as described before (cf. section 4.3). The ontology itself is available in an OWL file and located in the same path as the other resources. Similar to the identification of the targeted MongoDB, the ontology has to be defined using the `config.properties` file. In addition to the name and the created namespace, other key points are also defined here, which are specifically described for handling the prototype. Similar to the defined collections of MongoDB. The usage itself is enabled via the libraries of Apache Jena [Apa22b] as well as ONT-API [Git22b]. While Jena inherently enables a variety of different ways to deal with ontologies, the latter is primarily used to overcome the lack of support for OWL 2 ontologies.

The extensive `OntologyExtractor` class is responsible for handling the ontology within

the prototype. In this, a multiplicity of methods is listed, which load and add various information from it. Extensive methods of the used libraries as well as SPARQL statements are used for this purpose. In the context of the superordinate framework (cf. Figure 4.22), this class represents the necessary bridge between the lower levels to load and administer technologies, requirements, SUCs, and other information. To ensure that these are kept consistent at runtime and do not have to be reloaded repeatedly, the class, like many others, is treated as a singleton and instantiated only once, after the successful login as described in the code fragment located in Figure 5.5.

```
1  private OntologyExtractor() {}
2      ...
3      public static OntologyExtractor getmInstance() {
4      if (null == mInstance) {
5          mInstance = new OntologyExtractor();
6      }
7      return mInstance;
8      }
9      public void loadOntology(String path) {
10     manager = OntManagers.createONT();
11     try {
12         ClassLoader classLoader = getClass().getClassLoader()
               ;
13         File file = new
14         File(classLoader.getResource(path).getFile());
15         _ontology = manager.loadOntologyFromOntologyDocument(
               file);
16       properties = PropertiesConfiguration.getProperties();
17         _base = _ontology.asGraphModel();
18     } catch (OWLOntologyCreationException e) {
19         e.printStackTrace();
20     }
21     }
22 ...
23 }
24 public class WelcomeUI extends MainLayout implements View {
25     public void makeContent() {
26         OntologyExtractor.getmInstance().loadOntology(
               properties.
27         getProperty("ont.owl"));
28 ...
29   }
30 }
```

Figure 5.5: Initiate the ontology

5.4 Implemented Components

After each identified component (cf. Table 2.10) has been scientifically investigated in the previous section, the insights gained from this will be applied to the current prototype, fulfilling the second layer of the DECIDE framework (cf. Figure 4.22). For most of the

components, the decision support view was created as an entry point for better navigation, as shown in Figure 5.6. Here, the user has the possibility to check the sensibility of a big data technology application, to compare the planned project with existing SUCs and to start a structured decision support procedure. Since all activities described here provide decision support at least to some degree, they have been bundled together. Menu items have been created for all others, namely those that deal with modeling and deployment. The integration of each component is described in the following subsections.

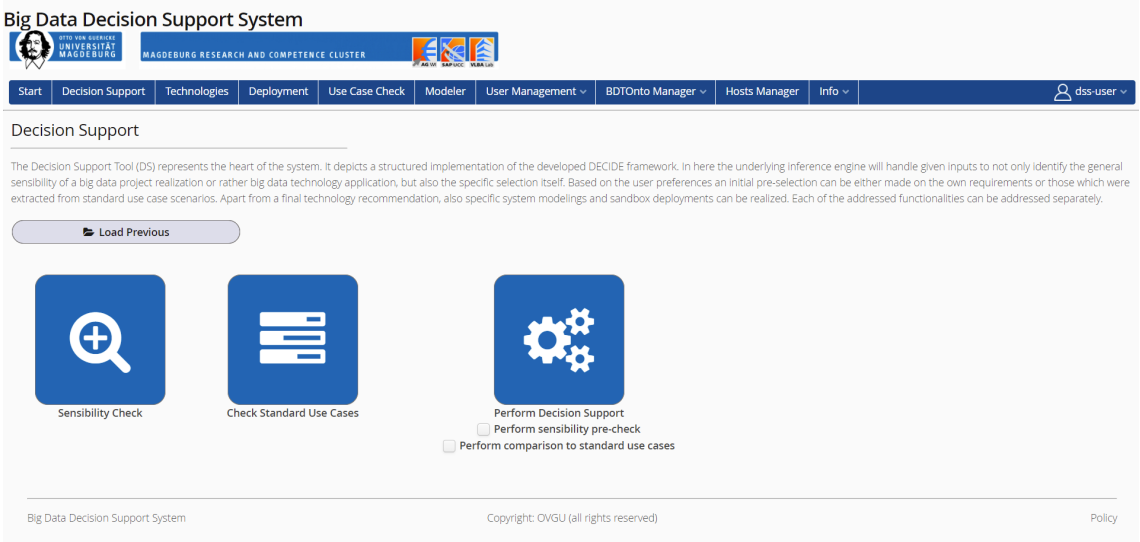


Figure 5.6: The starting screen of the decision support procedure

5.4.1 Big Data Technology Application Check Component

For the overall assessment of the general sensibility, as investigated in section 4.1, a framework and a semi-structured procedure were created that rely on existing data characteristics commonly originating from the requirements. Although extensive knowledge has been gained regarding RE, this process is still very specific to the planned undertaking and dependent on various factors, such as the organization. A computer-aided implementation is therefore difficult to achieve. Instead, the application of this component assumes that a RE with a focus on the data as well as the system functionalities and properties has already been performed. The information collected in this regard can then be transferred then into the system. For this purpose, a single view was created, containing a short description of the usage and an accordion. In the first tab, the defined framework is to be found and some explanations for it. The second tab contains a slider for each characteristic, which contains the expression of the data. Here, the user can use or neglect optional characteristics, depending on whether the check box is activated.

Each characteristic on the slider can also be found in the framework in the same way or in a slightly modified form. The slider remains untouched if no statements can be made and the value is NULL. However, by default, it is not possible for the three

core characteristics. After all inputs have been made, the third tab can be selected. The calculation, as shown in Figure 4.3, is executed when clicking the button. According to the previously identified range, a recommendation regarding the implementation of big data is then suggested or not.

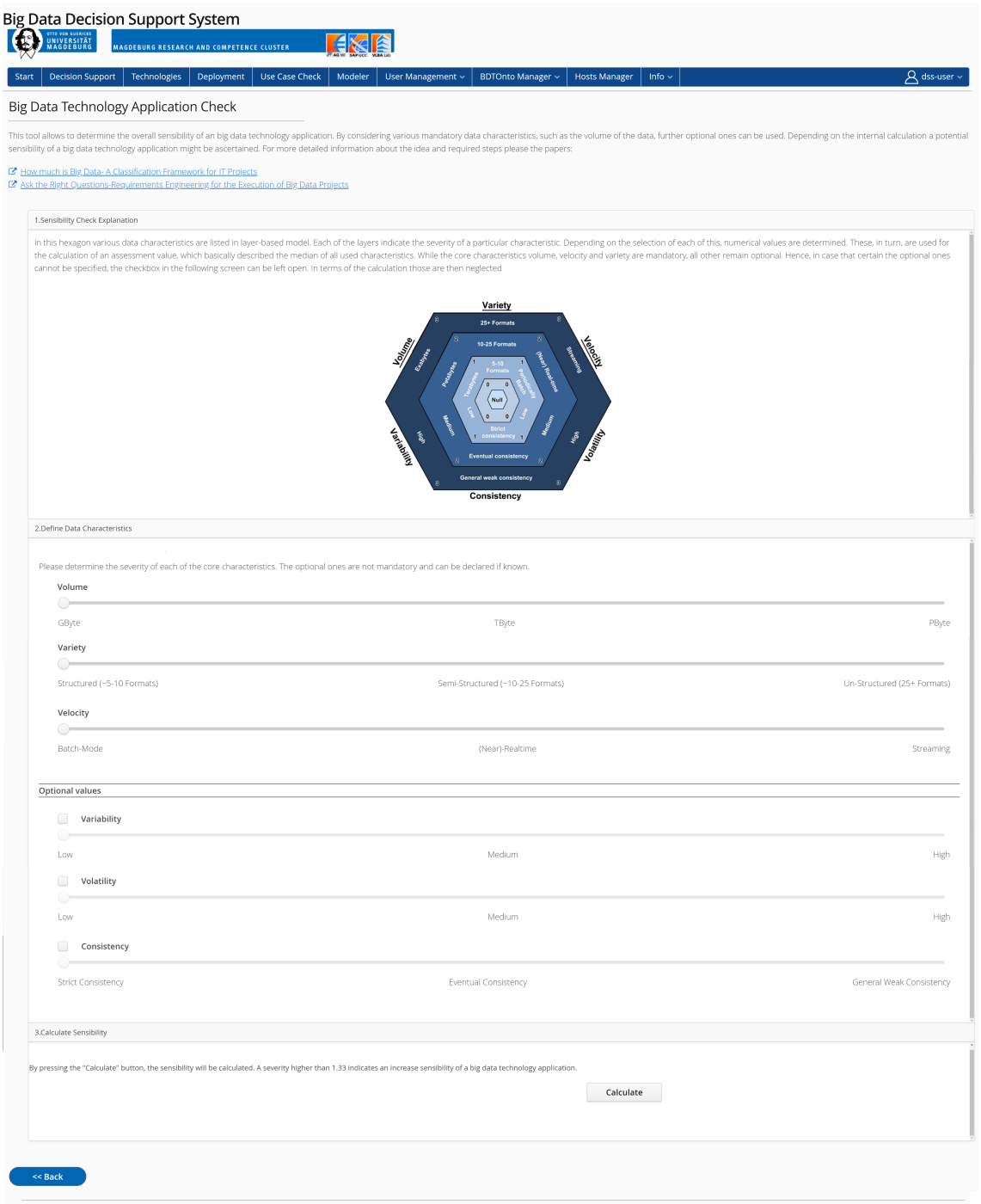


Figure 5.7: View of the technology application check component

Suppose the sensemaking was performed as part of the structured procedure. In that case, the inputs and results are stored in the overarching `DSSBean`, which holds all information while conducting the end-to-end process. Further details about this class are shared in section 5.4.3. An overview, which shows all expanded tabs, can be found in Figure 5.7. Due to the fact that neither the ontology, MongoDB, nor other complex structures have to be considered here, the implementation is the lightest one.

5.4.2 Standard Use Case Comparison Component

The SUCs identified in section 4.2 offer interested users the opportunity to get an impression of big data projects. As already described, each of these is based on a number of different use cases that have been documented in great detail and contain information regarding FRs and NFRs. This information is presented in a rudimentary form on the comparison screen. Once again, this is an independent view containing an Accordeon. Each of the related tabs describes a SUC in detail, provides the corresponding contributions and the values determined for FRs and NFRs. The first tab provides a short summary for better clarity, on the basis of which a possible comparison can be carried out quickly and independently by the user. If the component is included in the structured process, a selection screen is also activated where the user can select a SUC as a basis for further considerations. A depiction of this view can be found in Figure 5.8.

Unlike the previously mentioned component, the integration requires significantly more effort. This is mainly due to the loading of the data from the ontology and the provision of the contributions for the individual SUCs. Each SUC is defined by the class `StandardUseCase` that is located the `models` package. Each of them contains an ID, name description, requested FRs, requested NFRs, and a list of related UCs. Since these are depicted by particular research articles, similar information as for the `PublishedPrototypePaper` class are used, including, for instance, the title, abstract, publication year, and authors. At start, all SUCs are loaded via the `getAllStandardUseCases()` method located in the `OntologyExtractor` class, which is similar to loading the big data technologies. As indicated in section 4.3.2, the severity of each NFR for technologies and SUCs is determined by an individual within the ontology. In the context of the SUC, the same applies to the UCs. Hence, a `getIndividualsList` method is used for both.

Noteworthy here, the median and the highest value of each NFR are stored, using the `hasWeight` and `hasWeightMax` data property, providing the potential user as much information as possible. The aligned UC individuals are named `Paper_Reference_SUC_ID_YY`, where the `YY` denotes the number used in Table 4.6, Table 4.7, and Figure 4.8.

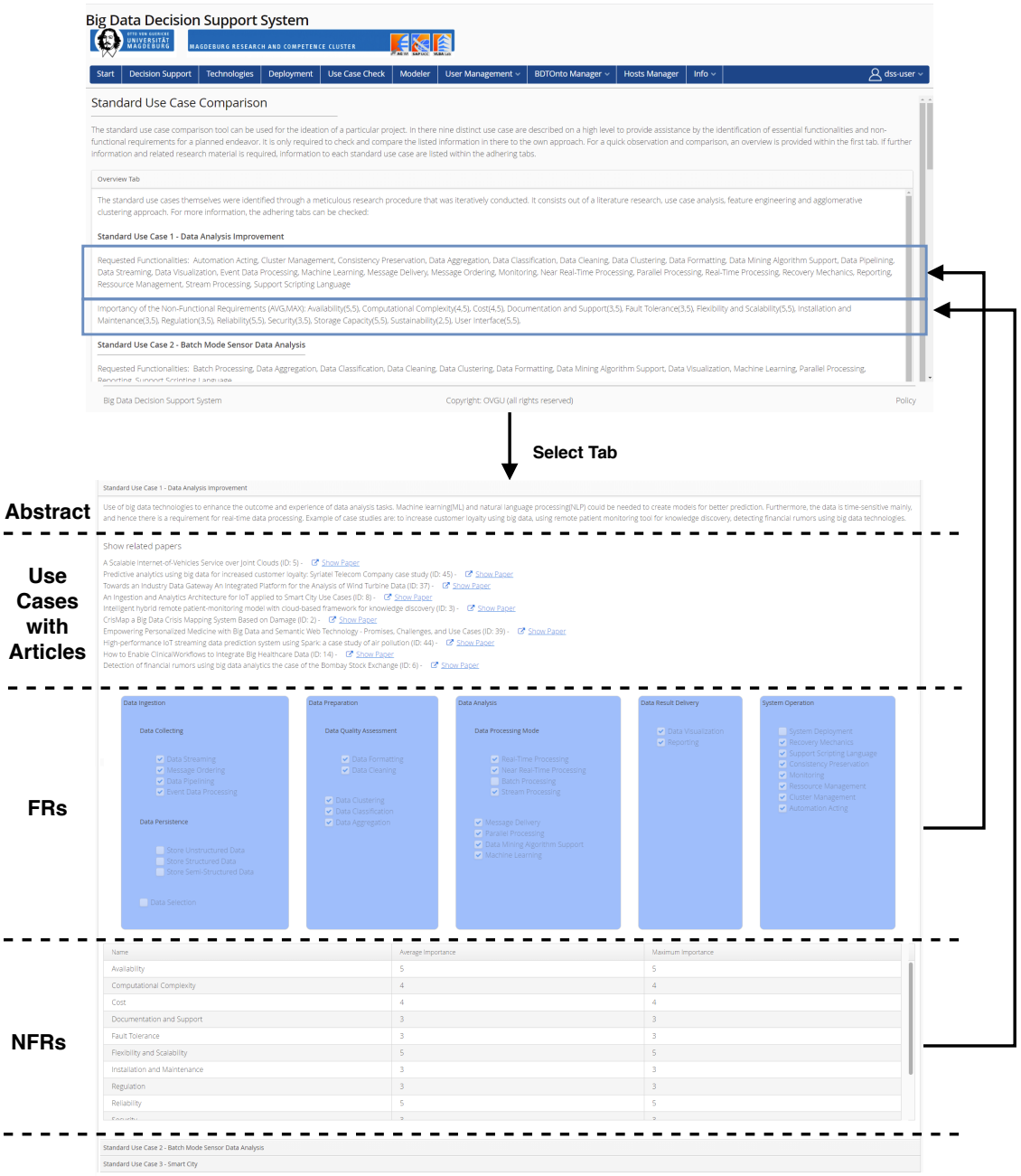


Figure 5.8: Overview of the standard use case comparison screen

These are then searched within the `standard_use_case_papers` folder, located in the `WebApp` resources. With respect to the FRs, similar methods are triggered for single big data technologies. All of these are dynamically loaded from the ontology, including their categorization. In each of the related tabs, these are depicted by preselected and disabled checkboxes. If the user hovers a particular FR, the description is shown (these can be found in Table A.4). The same applies to the NFRs and their severity (see Figure 5.8). An overview of every interaction with the `OntologyExtractor` class is depicted in Figure 5.9.

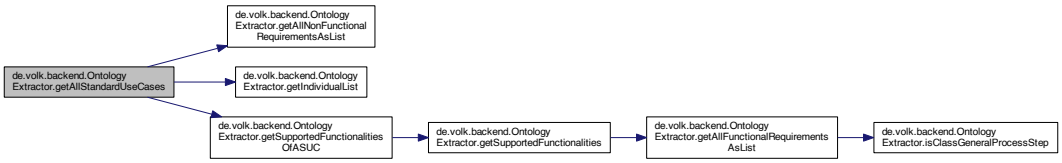


Figure 5.9: Class diagram of the SUC comparison view interacting with the OntologyExtractor

5.4.3 MCDM Inference Engine Component

The MCDM component constitutes the inference engine and represents the *heart* of the system, as it was described in section 4.4. As an essential part of the system, in charge of the overall identification of a suitable recommendation, various steps are carried out here. As noted in the previous course of this work, the sole recognition of NFRs as performed in other contributions (cf. section 3.1.2) is insufficient, mainly because of the number of existing technologies and their steady increase. Hence, the insights achieved through the previously conducted research were implemented. Due to the comprehensiveness and complexity of the required steps, for the application of the inference engine, independent from the number of involved steps, the mentioned `DSSBean` is used. In this, relevant data is stored during run-time to interconnect the steps and, thus, the components with each other. Apart from plain fields as well as relevant getter and setter methods, auxiliary functions are integrated here. In later stages, these are in charge of the calculation of the MCDM value calculation, the identification of the single best technologies, and their combination. Comprehensive information is presented to support the user with the relevant inputs, as in the whole system itself. This does not only comprise some introductory words during the presentation of each view. Instead, also notifications for missing and incorrect inputs are given. Like other text fragments, these are stored in the related `Messages_XX.properties` file.

For the procedure itself, first, it is required that the user identifies relevant FRs, as they are delivering information about the provided functionalities of the system in the end. In doing so, the views and methods for depicting fulfilled FRs in the SUC comparison screen were reused, as shown in Figure 5.10. The loading procedure for those is similarly realized as for the SUCs. Each entry is depicted as a separate checkbox the user may select. The system cannot fully handle the precondition of completed requirements engineering with which the user determines the details regarding the FRs and NFRs (cf. section 4.1.3). However, in case of uncertainty, the user could utilize the provided view to identify relevant functionalities, similar to the use case comparison component. The same applies to the NFRs that shall be determined afterward. At this point, it is sensible for a user to observe those in beforehand and discuss them with other stakeholders.

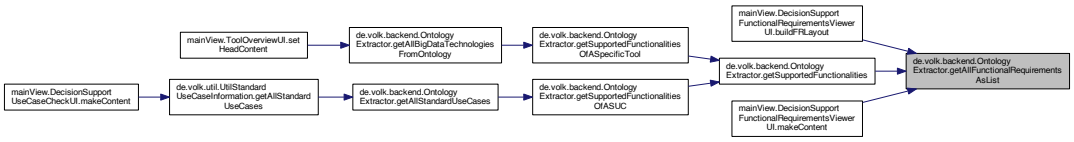


Figure 5.10: Extraction of FRs used for the SUCs, technologies, and the selection screen.

Furthermore, the number of comparisons that need to be conducted might be a daunting task, for a single person, especially if some of the NFRs would be not specified or irrelevant for a particular case. Due to this circumstance, counter-measures were implemented, which tolerate that the user only recognizes relevant NFRs. The respective viewer described by the class `DecisionSupportNonFunctionalRequirementsViewerUI` consists out of three tabs, each hosted in an `Accordeon`. Within the first tab, as shown in Figure 5.11, similar to the FRs, the user can perform the selection for the preferred NFRs that can be compared with each other. The indifferent comparison value one is given for the remaining, unselected ones.

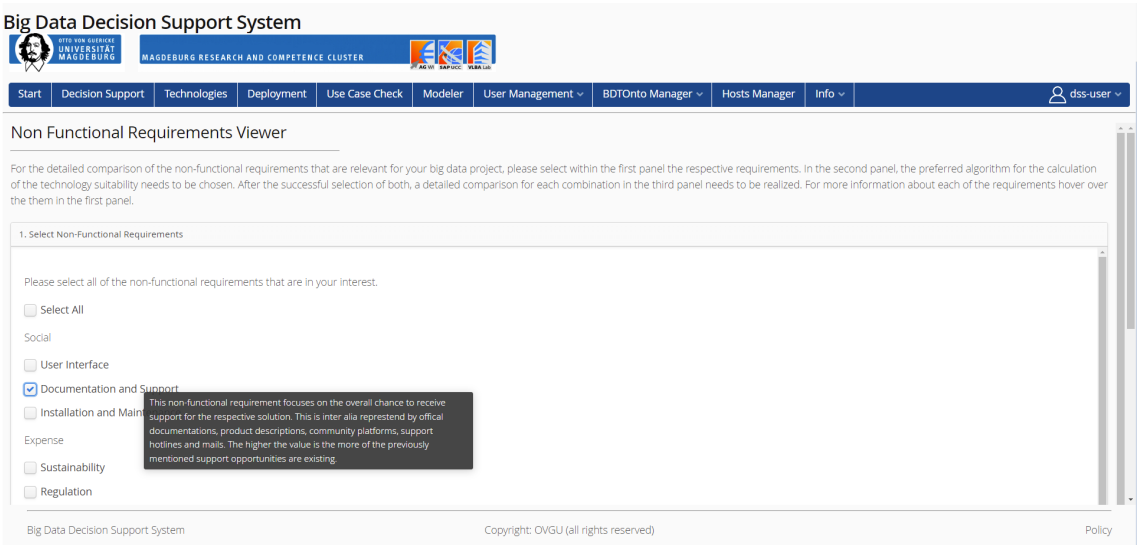


Figure 5.11: Selection screen of the non-functional requirements

Like in the previous views, all information is dynamically loaded from the ontology, including also their overarching categories. Once the choices were made, in the second tab, the desired MCDM algorithm can be selected. As discussed before, the AHP was identified as the most suitable approach for this and therefore chosen for the prototypical implementation. However, further placeholders are defined here that can be extended in future research. Remarkable, the TOPSIS algorithm is implemented as well but has not been fully tested so far. Furthermore, additional options, such as determining default comparison values, could be integrated. At the current moment, only extra weightings of the defined NFR categories and the possibility for category-only comparisons are imple-

mented in the prototype but not added to the view because of the extra effort required by the user. At this stage, these available options could be too complicated for inexperienced people. Nevertheless, once the role concept is fully integrated, this could be enabled for expert users, such as big data engineers. The tab itself is depicted in Figure 5.12.

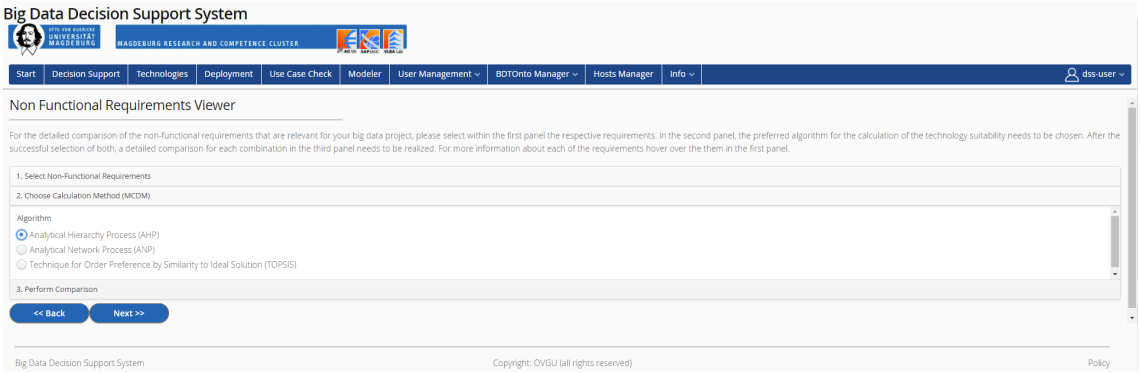


Figure 5.12: Selection screen of the MCDM algorithm

After the selection is made, the third tab can be opened. When this is attempted before, the user will be notified regarding missing inputs. Depending on the choices, primarily the number of selected NFRs, a slider is integrated for each pairwise comparison. The user can then decide in favor of a specific NFR or keep the default value if no clear preference exists. This would, in turn, result in an indifferent state, in which *one* is assigned later on. Generally, each row entry describes one pairwise comparison between two NFRs in the AHP. Basically, this denotes one particular cell as well as the reciprocal value within the created preference matrix (the values above and below the diagonal in the matrix). The given range does not match the values ranging from one to nine, as they are recommended by Saaty [Saa08] (cf. Figure 2.16). Vaadin does not support custom slider descriptions. Instead, as showcased before, the value set by the user is programmatically manipulated by adding a *one* to the NFR. For example, when the user sustainability with regulation and rates the former with a three, internally, a four is used to confirm the recommended one to nine rating by Saaty. In turn, the reciprocal value for *regulation* and *sustainability* is one-quarter instead of one-third. An overview of the automatically created tab tailored for the AHP is shown in Figure 5.14

After all inputs have been made, the final inference engine screen for the recommendation is provided. The view itself follows a similar setup as the one before. Again, an `Accordion` is used that comprises four tabs. Within the first, an overview of the made inputs is given. Depending on the selected checkboxes and performed preparatory steps, the information of the sensemaking, use case comparison, identified FRs, and NFRs are listed. In case any inputs need to be revised, the user can go back and perform the required changes. Otherwise, the second tab can be clicked, in which the single best technology is identified.

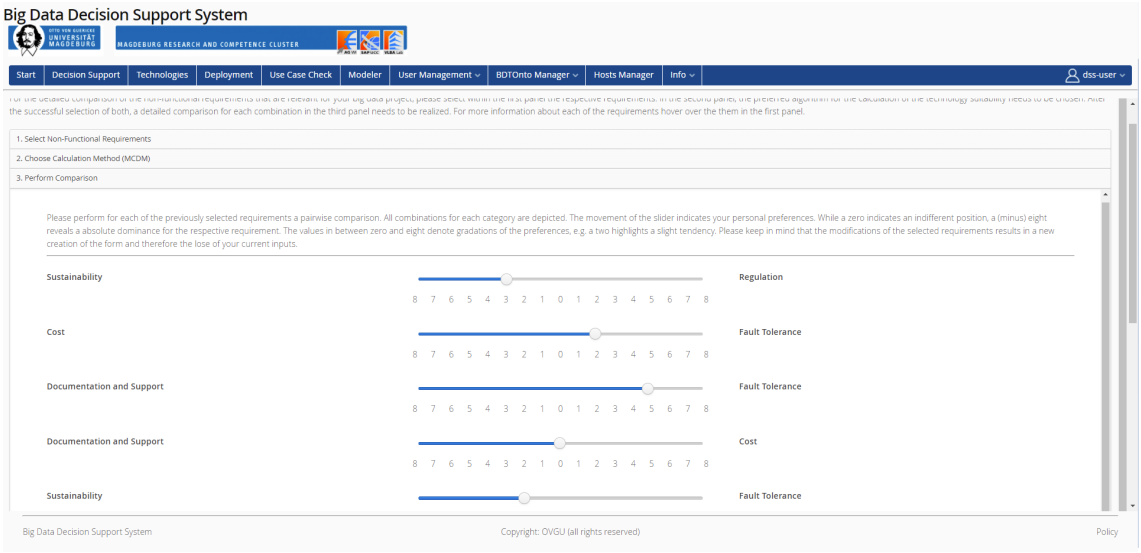


Figure 5.13: Build a pairwise comparison layout for the AHP

As described within Algorithm 1 functionalities are first checked to examine whether technologies exist that are capable of fulfilling all of them. If this is the case, these are stored in the `completeTechnologies` list, which is later ranked based on the calculated MCDM value (e.g. AHP) and the number of totally fulfilled functionalities. The same applies to those that are at least partially fulfilling some of the requested functionalities. As a foundation for the further combination, these are stored in the `incompleteTechnologies` list. When these are presented, the already met functions are underlined. An overview of the single best technology tab is shown in Figure 5.14.

Compared to identifying fulfilled functionalities, the calculation of the AHP values requires more effort for the single best solution. The entire analysis takes place in the class `AHPCalculation`. Here, two constructors define whether the AHP is conducted based on the given preferences or the technologies. Both are required to calculate all relevant weighting vectors for the own preferences and the fulfillment by the *alternatives* (cf. section 2.3.2). While the first solely takes the inputs from the NFR screen, the latter uses all information of a single technology defined by the weighting individuals in the ontology (cf. section 4.3.2). As described before, a rating between one to five was used for the technologies (cf. Table A.7), similar to the SUCs, which do not conform the recommended rating by Saaty [Saa08]. Hence, when building the related matrices for each NFR of all technology comparisons, the previously proposed mapping in Table 4.11 was implemented using a nested switch-case setup. For the identification of the single best solution, the AHP calculation is only used for all completed or partially fulfilling technologies. Therefore, the number of comparisons does not comprise all of the technologies stored in the ontology. Independent from that, by using the second addressed constructor in the `AHPCalculation` class, for every NFR, a separate matrix needs to be created.

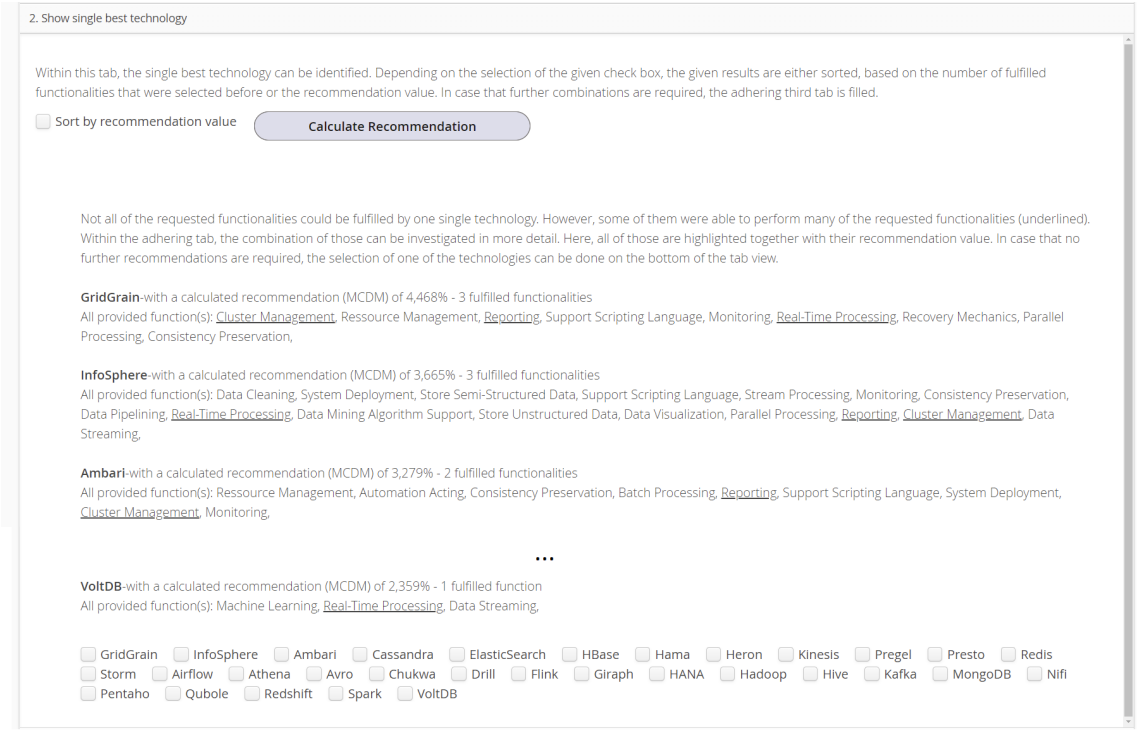


Figure 5.14: Recommendation of the single best solutions

At the current moment, this results in the creation of 13 matrices (cf. number of NFRs in Table A.4) and, thus, 13 weighting vectors which later on form the rating matrix. As soon as the results are created, the user can either finish the decision support by checking the intended technologies and moving to the fourth tab or continue with the recommendation of the technology in the third tab. If no single technology was found that fulfills all FRs, the third tab could be chosen; otherwise, the user is notified. An excerpt of this view is depicted in Figure 5.15.

Like in the second tab, a `Calculate` button within the technology combination recommendation triggers the algorithm, as proposed in Algorithm 2. In particular, all entries of the `incompleteTechnologies` list are used as a foundation for a potential combination. Then, for each of them, the list is iterated another time to check for technologies that fulfill most of the currently missing functionalities. When found, the technology is combined with the presently existing. This is repeatedly done until all requested functionalities of the user are covered by this combination. In case single technologies are here equally filling the gap, the individual AHP is also calculated to consider the NFRs. Eventually, each found combination is stored in a `Map<BigDataTechnology,List<BigDataTechnology>>` containing the foundational technology as `key` and the recommended technology combination as a `value`. After this is done for all technologies within the `incompleteTechnologies` list, technology recommendation duplicates are searched and purged. Due to the circumstance that each combination covers multiple technologies and thus multiple NFR expressions, the AHP cannot be performed in a multi-leveled setup here.

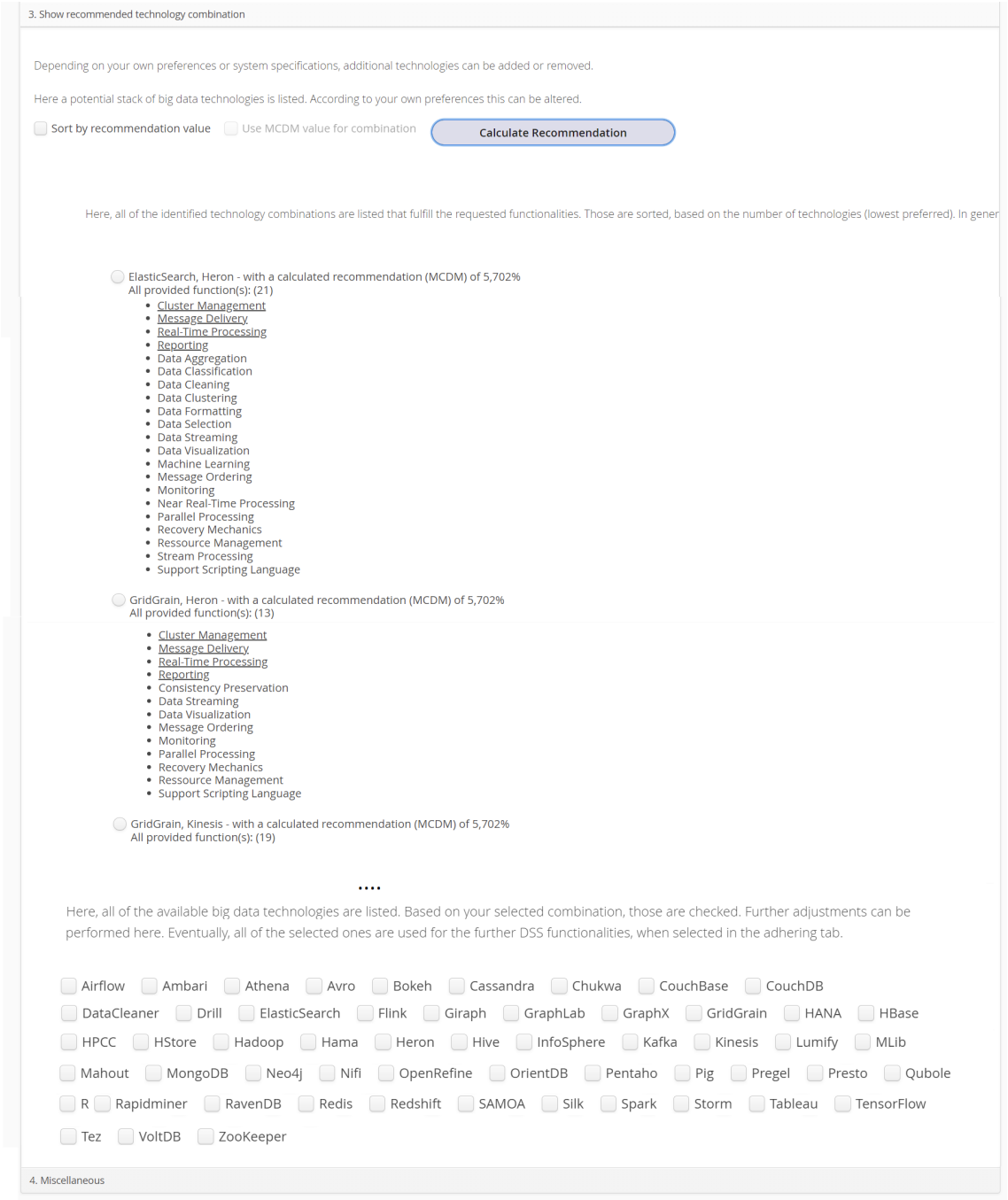


Figure 5.15: Technology recommendation tab

Instead, the average for a particular NFR of each technology is calculated and used for the AHP calculation. Based on the received value, the list of combinations is sorted based on the estimated value. The user can then choose the desired combination. After selecting a suitable combination, a preselection of all available tools is made at the bottom of this tab. If specific tools should be added to or removed from the overall recommendation, the user can select and deselect the check boxes for each technology. When finished, the

fourth tab can be pressed. Here the final results and further options are presented, from which the user can choose. In particular, this comprises the architecture modeling (cf. section 5.4.4), the deployment of the technologies (cf. section 5.4.5), or both. An overview is shown in Figure 5.16.

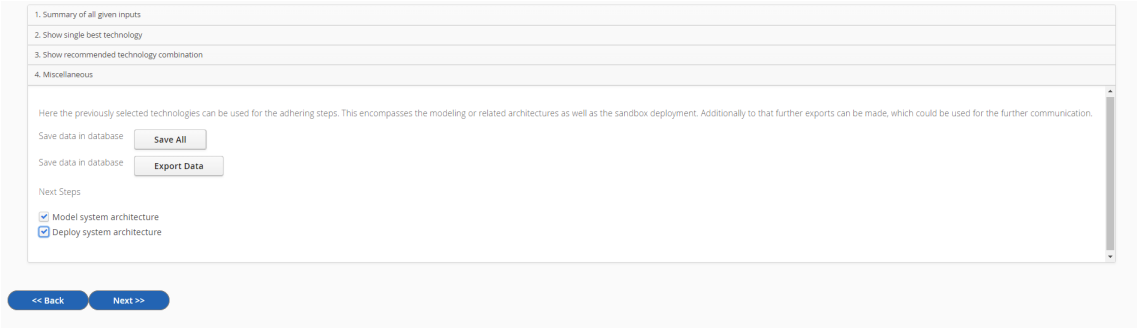


Figure 5.16: Final recommendation tab that depicts further options

5.4.4 Architecture Modeling Component

The modeling of BDAs was studied and described in detail in section 4.5. The result was a special profile for the deployment diagrams used in UML to represent system architectures. The creation of these diagrams does not take place using a typical drag-and-drop approach, with which the elements can be moved manually. Instead, this procedure is automated by utilizing a configuration file. In particular, those follow the standard structure of JSON files, which, like other semi-structured files, such as XML or YAML files, are easy to read, understand and distribute. Therefore, they form the basis for the creation of the models. A separate view was created and divided into two parts for the prototypical implementation. While on the left-hand side, the configuration is built, on the right-hand side of the view, the created model is placed.

In addition, another menu bar provides a variety of options, which allow the loading of already existing models and examples and their export. Independent of whether the modeling component is used as part of a complete decision support process or independently by the user, all provided functionalities can be equally harnessed. Regarding the former, in the case of existing technology recommendations, these are also displayed as elements to be modeled in the upper part of the screen. The actual application follows precisely the same procedure as described in section 4.5. The potential user creates their model in the left box by defining the JSON file independently. As a basis for this, an already validated and compatible file is provided, which can be modified as desired. Once all adjustments are made, the diagram only needs to be created using the menu item `Model`. The used segmentation of the overall view can be resized by shifting the delimiter to one side. As a foundation, the `HorizontalSplitPanel` provided by the Vaadin framework is used so that not both areas can be differently sized. Three additional buttons above the displayed model allow for zooming in and out. An overview of the implemented component interface

can be found in Figure 5.17. If the diagram is successfully compiled, the export function can be used to load it as a PNG or JSON file. Furthermore, both can also be loaded together in the form of a zip file. By selecting the menu item `New`, the base diagram is re-initiated that is dynamically loaded from the source folder where all component-related files are stored. In particular, all of these are located in the `/static/modeling-examples/` folder.

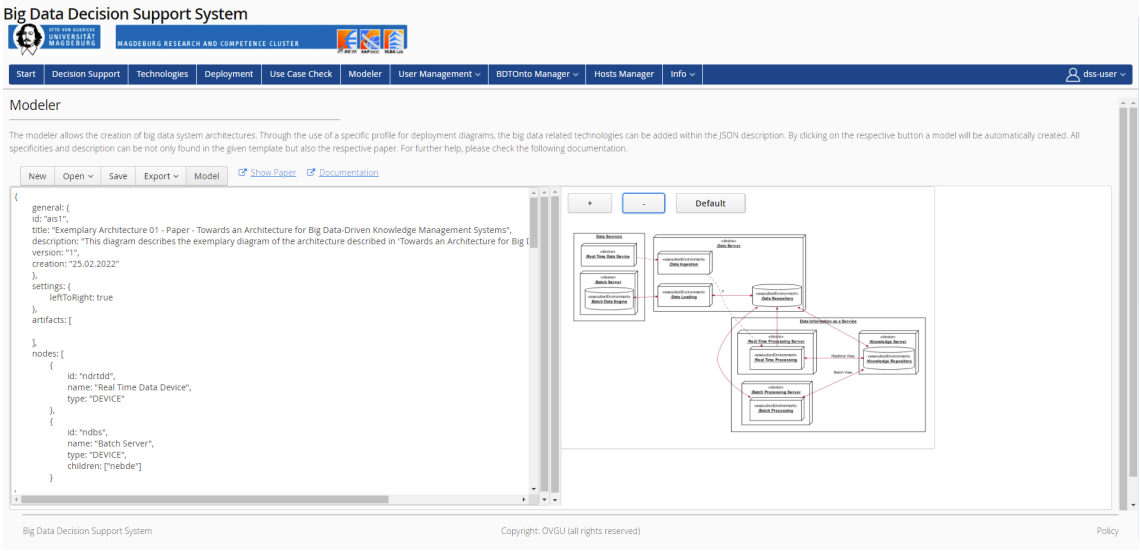


Figure 5.17: Implementation of the big data system architecture modeler

If incorrect entries, connections, or other problems occur during processing, such as missing certain characters, the developed validator recognizes these independently and provides the user with corresponding information. The error log is displayed in the same window as the model so that the user can quickly and clearly understand where the problems exists. In the future, the menubar could be expanded to include predefined elements where, for example, connections between certain technologies can be directly and automatically integrated. Particularly in the case of comprehensive models in which organisationally relevant components of the IT infrastructure are also to be mapped, the implementation can be very time-consuming.

Additional help for the creation can be found in the form of a multi-lingual documentation, which can be accessed by clicking on the respective link `Documentation`, next to `Show Paper`. Similar to the procedure for setting up the language files (cf. 5.1), the specific version is called up here. In another entry hidden under the menu item `Open`, examples of already existing deployment diagrams and their configurations can also be accessed. The dialogue that is opened is shown in Figure 5.18.

The displayed table currently contains eleven deployment diagrams. As described before in section 4.5.3, all of the models were created in the course of the evaluation of the scientific foundation. Thus, the corresponding scientific contribution is stored in

addition to the actual configuration file, which can be used, once again, to check relevant information about the background here. All the prefabricated examples can be found in the resource directory. In addition to the actual JSON file, this includes the corresponding paper. If further samples are planned for the future, they can be implemented in the corresponding enum `ModelingExamples`.

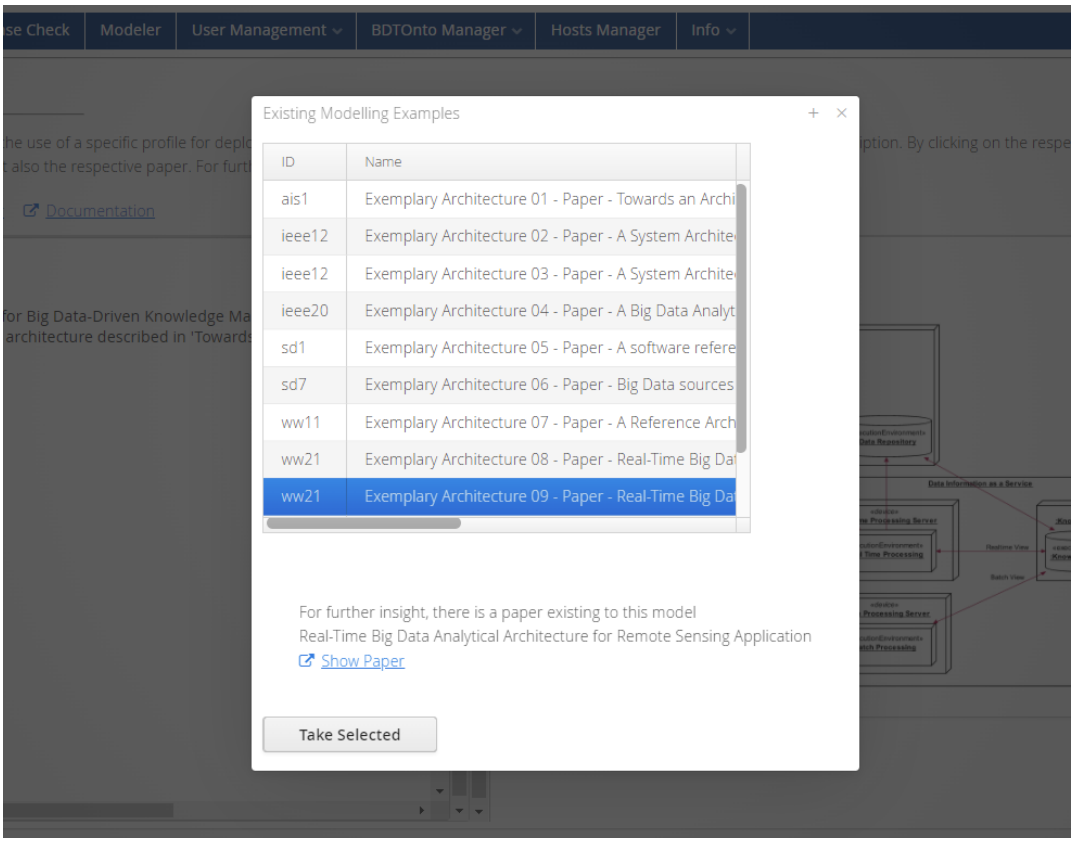


Figure 5.18: Selection screen of related examples, shipped with the system

The actual model creation takes place through the interaction of the main classes `ModelingUI`, `PlantUML`, and `ModelProcessing`. Within the first class, which is responsible for the complete construction of the view, the creation is triggered with the `buildUMLModel()` method by clicking the *model* entry in the menu bar. First, a `PlantUML` element is created, containing the raw JSON code, its validation status, and the translated PlantUML code. With the instantiation of such an element, the input defined in the constructor is first evaluated, with a number of validators being involved. If this is correct and none of the errors mentioned above have occurred, the JSON file is translated into the necessary PlantUML code. All the methods required for this can be found in the corresponding `ModelProcessing` class. After the PlantUML code is created, it is passed to the PlantUML element and translated into an image using the PlantUML library. Finally, this image is integrated into the right part of the `HorizontalSplitPanel`.

5.4.5 Architecture Deployment Component

The automatic deployment component described in section 4.6 was implemented similar to the previous elements. The previously formulated general process remained unchanged here. However, in some places, special concretizations were handled, such as the definition and use of the registry, which was done here with the help of a GitHub repository. The component itself spans over two views and several classes, responsible for the automated creation of the necessary files. In the following, an insight into the views created as well as the use and implementation will be given again.

First of all, the `Host Manager` should be mentioned in the context of this. It is essential for defining and managing the endpoints and thus crucial for creating the inventory file. The user can specify the IP, port, and the defined user for Ansible here. As described in section 4.6, the configuration management tool Ansible and Docker are mandatory in addition to the BDTOnto. Ansible and Docker must be available on the host system and installed in advance. This is the only requirement to enable the deployments. The user created for this purpose is entered accordingly at this point. The persistence of the endpoints takes place via MongoDB, using the collection `server-hosts`. An overview of the view can be found in Figure 5.19.

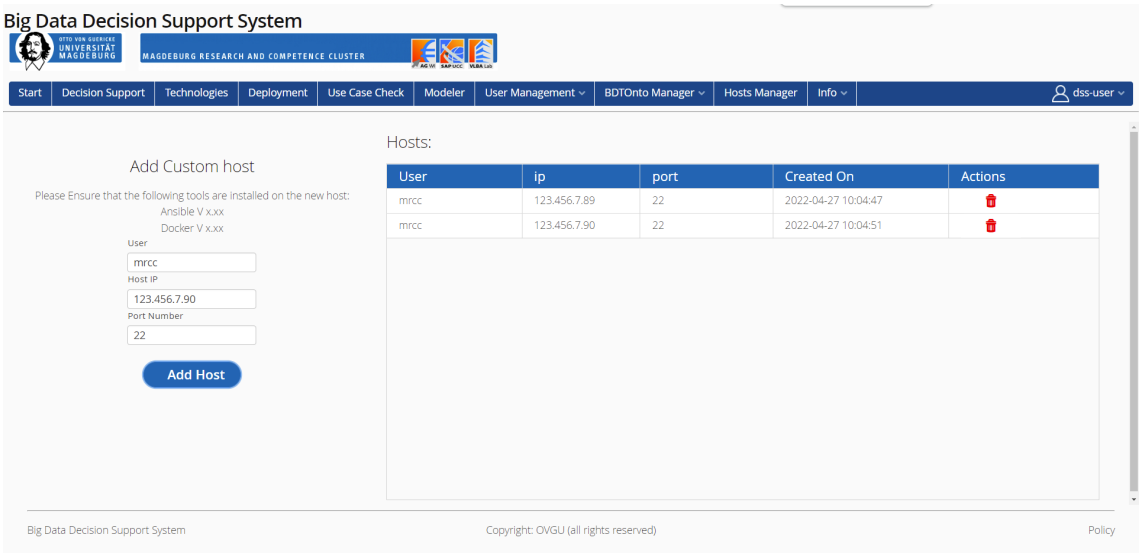


Figure 5.19: View of the Host Manager for the configuration of the hosts.

Once the endpoints have been created, the `Deployment Manager` component can be used, as it is shown in Figure 5.20. It can be used independently or as part of an overall process, similar to the others. The main difference lies in the provision of the available technologies. When the deployment tool is called independently, all available big data technologies are shown in the table on the left. However, when technologies have been recommended and selected in advance by employing the inference engine (cf. section 5.4.3), only these technologies are displayed. Important here is that only technologies can

be used or selected for which corresponding pre-configurations exist. For this purpose, the initial processing of the ontology first checks whether an entry exists for each big data technology. For the prototypical implementation, an external GitHub repository was used as well as the annotation `git_repository_name` within the ontology, for each technology. These provide the required name of the *role* folder that is relevant for the construction of the Ansible playbooks. By serving as an external *registry*, numerous benefits are enabled.

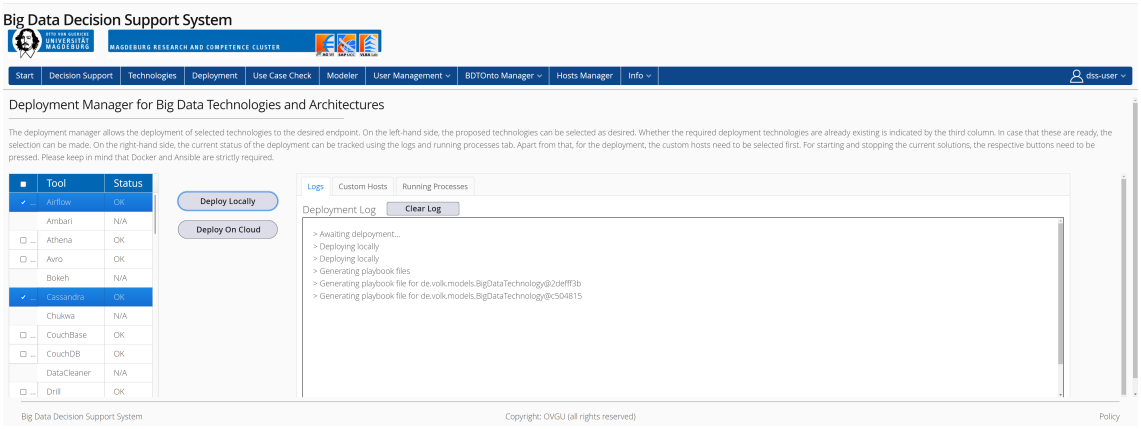


Figure 5.20: Deployment manager view

The external management of the data, facilitates increased security, which is given by the division of responsibilities between users and maintainers. Consequently, the data can be held and managed decentralized. New technologies and versions can be added without updating and redeploying the system. The MongoDB collection `git-ansible-files` stores all relevant metadata of each file. In particular, this includes the file names where the tasks are described, their URLs, and the Secure Hash Algorithm (SHA) values. When a change of the files takes place, in the remote repository, these values are updated, and the new data is retrieved. Within the repository, two roles for each technology are located. In particular, these are two separate folders. Within the eponymously named folder of each technology, the tasks and related files for the deployment are defined. In turn, the respective tasks for the decommissioning are included within the folders with a leading *remove* identifier.

When the external data is loaded, a host chosen, the targeted technologies selected, and the process initiated, a playbook for deploying and decommissioning the system is dynamically generated, consisting of the selected technologies. In particular, the respective roles are loaded, the inventory and playbooks created, and the required command sent to the custom host system. The user can monitor all performed steps in the log tab throughout this process. Once the deployment is finished, the process is registered using the `RunningAnsibleProcessesBean`. If the user wants to decommission the system, all running processes can be checked in the third tab. Here, all running technologies can be stopped by triggering the removal task.

5.5 Additonal Functionalities

Besides the basic components, which are essential for the instantiation of the DECIDE framework, additional functionalities were added to meet the characteristics of a DSS and to prepare the prototype for future developments. In the following, some of these are briefly described.

5.5.1 Multi-Tenancy

The system as such has a multi-tenancy function, which allows several `users` to share a system. It is therefore not necessary that this must be operated independently. Basically, the initial users collection within the MongoDB is used, containing essential information about each user. Here, potential role permissions, personal details, and login data are persisted in a hashed form. During the initial login, the user can specify these and set a cookie to stay logged in. For the development, an `autologin()` method was created within the `MainView`. When deployed in a production environment, this needs to be turned off. Furthermore, a cookie can be set once the user can successfully log in. An overview of the corresponding view is given in Figure 5.21.

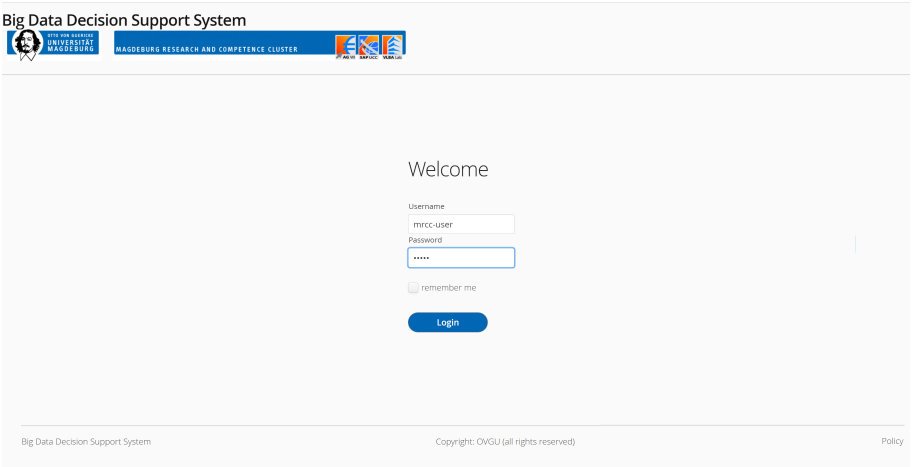


Figure 5.21: Login view used for a multi-tenancy support

5.5.2 Ontology Viewer

In order to get a better understanding of the underlying knowledge base and to avoid using external tools for its representation, a separate ontology viewer was created. This is shown in Figure 5.22 and offers interested users the possibility to view the general hierarchy of the parent taxonomy as well as the dependencies and details of each class. By clicking on a single entry on the left side of the view, the details can be viewed using a dedicated field on the right side.

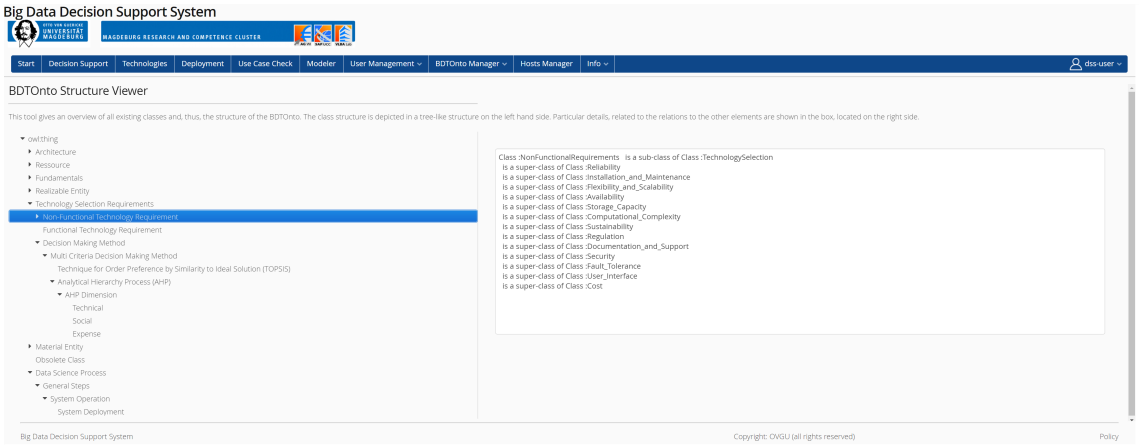


Figure 5.22: Ontology viewer depicting the overall class hierarchy

5.5.3 Technology Presentation

For an additional overview of all big data technologies and their specificities, a technology presentation tool was created, as it can be seen in Figure 5.23. All information recorded in the annotations and relations within the ontology is depicted here. Besides the name, a description, example usage commands, as well as potential advantages and disadvantages are mostly listed.

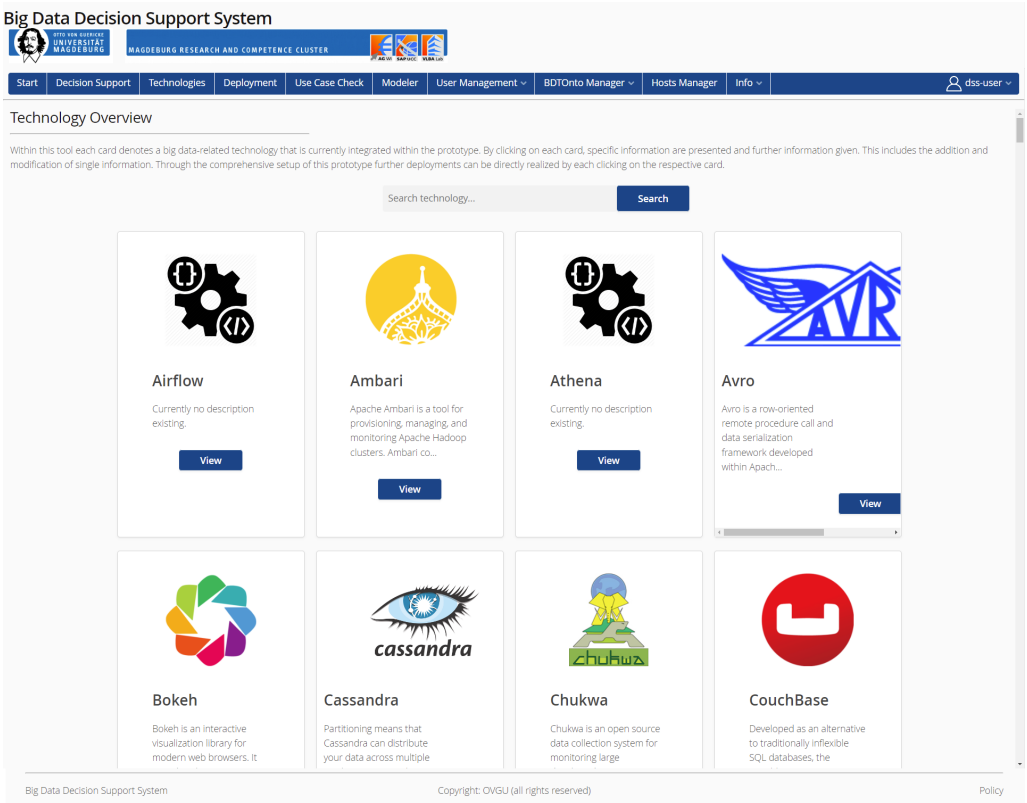


Figure 5.23: Technology overview screen

Additionally to that, the fulfilled FRs are always given. The respective image, when available, is loaded from the image resources located in `images/tools/default`. The named details are displayed once the user clicks the corresponding card. Thus, users can inform themselves about specific solutions apart from external sources by utilizing the prototype.

5.6 Using the System for an End-to-End Decision Support

The use of the system and the interaction of the individual components were already partially described. Successively, each tool, which represents the instantiation of a component, was presented. In order to use the system for the fulfillment of the holistic decision support, from end-to-end, various activities must be completed by the user, independently and in interplay with the system. In contrast to the sole use of individual tools in the system, the overarching project idea plays the central role here. Once planned and defined, the first interaction takes place by means of entering the login data. These are checked against the available data in the MongoDB. If this was done successfully, all necessary data is loaded from the ontology and provided to the user. Then, the user finds himself in the `Welcome Screen` (cf. Figure 5.2).

Depending on how concrete the first idea was formulated and whether requirements were already identified, the user can start directly with the decision support or compare the project with existing SUCs in advance, to specify further details. By means of the menu item `Use Case Check`, the view described in section 5.4.2 is called up, in which SUC descriptions, associated UCs, FRs, and NFRs can be compared independently. When no requirements were identified yet, the FRs and NFRs listed there can be primarily used as an orientation to create these, as it has been outlined in section 4.1.3, where requirements consisting of FRs, NFRs, and data characteristics were introduced. Regarding the data characteristics it was found that these do not provide detailed decision support. Still, they do provide a first impression of whether big data technologies are necessary for a specific project or not.

Accordingly, after specifying the problem, the `Decision Support` tool can be accessed by selecting the appropriate entry in the menu. There, the user has the choice of performing the aforementioned preliminary check as well as a comparison with the SUCs before the actual determination of the technology recommendation. If both are chosen, these will be shown before accessing the inference engine. The same applies if only one was selected. The preliminary application check for sensemaking requires information about the possible characteristics. Once this is provided, a general recommendation regarding a big data technology application can be made. The user can accept the result as a rough orientation or as a decisive criterion for a discontinuation of the system use. The started decision support process is not automatically aborted, even if the value is less than 1.33. However, all information that are concurrently entered will be saved in the `DSSBean`.

Apart from the sensemaking, this also applies to subsequent decisions, including the possible selection of a SUC. If the option is selected in the initial DSS, the user can choose an existing SUC as a base. The step introduced here can be used as a new orientation or to define a foundation for further specifications if not already done before. If the latter applies, all FRs of the SUC are transferred to the inference engine, which starts here with the view of the functional requirements and their preselection. The user can perform a (de-) selection depending on the made choice during the SUC comparison.

After this is done, the NFRs are chosen in the next step. As a rule of thumb, the more of these are selected and compared, the more detailed recommendations can be provided. This is mainly due to the indifferent actions that are assigned otherwise. The user must also select the MCDM algorithm itself in the second tab. Depending on the choice of the MCDM approach, the comparison screen is created at the end.

Once all comparisons were made, the results of these, FRs, NFRs, selected SUC, and data characteristics are stored in the `DSSBean`. In the last view, all of the data from the `DSSBean` is loaded again and given as an overview to the user. The user can then start the technology recommendation according to the behavior described in advance. In the case that no all-fulfilling technologies were found, the user can also create different combinations, which represent the final recommendations of the system here.

The user can then use the results independently, referring solely to the obtained information, or use them further. Specifically, it concerns additional modeling of the planned system architecture and the deployment of the identified technologies in a sandbox environment. Depending on how the decisions are made, the user is automatically forwarded to the respective tools and their views. The complete sequence of interaction with the system is shown again in a sequence diagram in Figure 5.24.

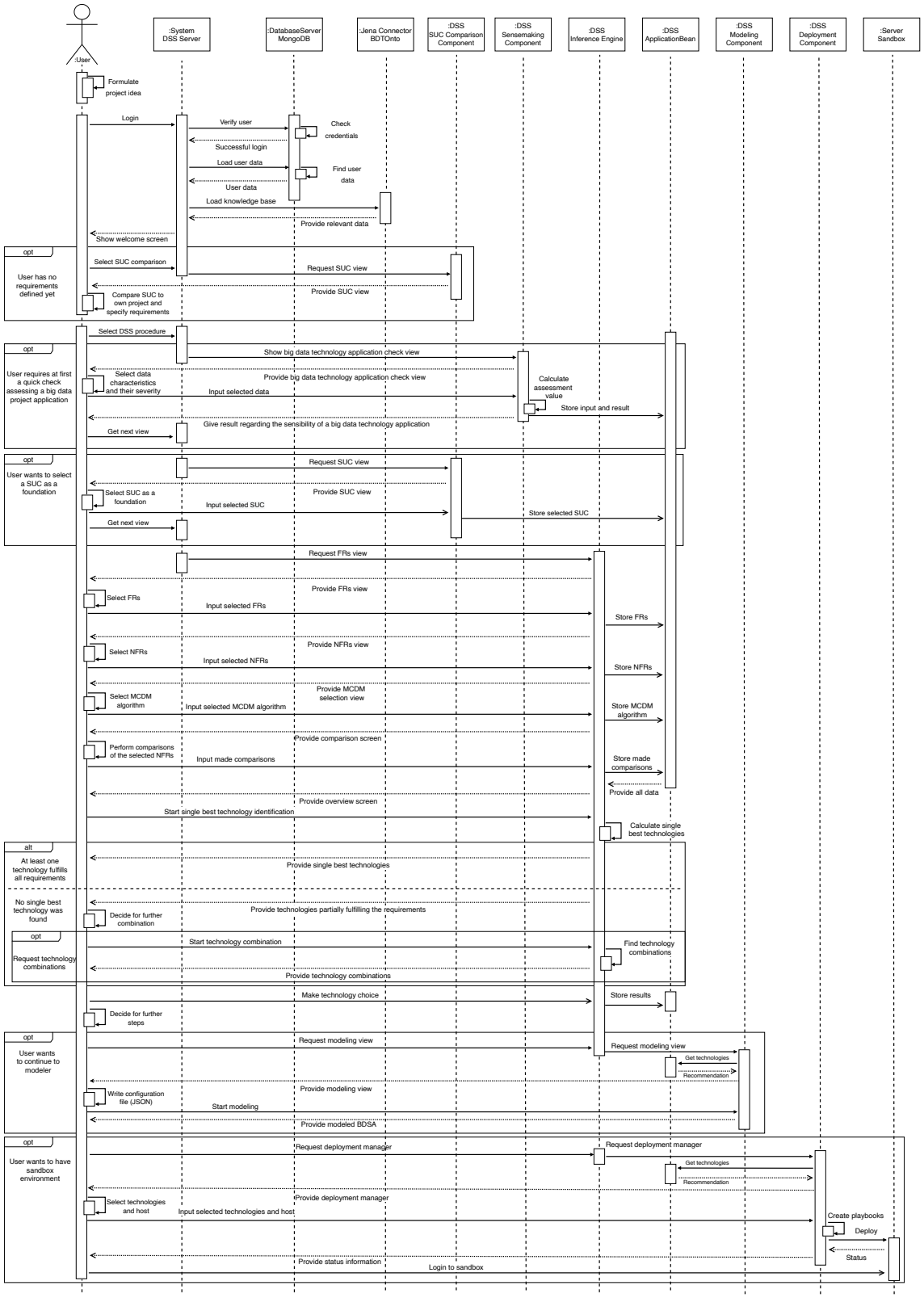


Figure 5.24: Sequence diagram of an end-to-end application of the DSS

5.7 Summary

Within this chapter, an extensive description of the developed prototype was given, representing an instantiation of the DECIDE framework. The general structure and the choice of the technologies used for the conceptualization were comprehensively described in the beginning. Subsequently, all components, which were called *tools* in the context of the prototype, were successively outlined. Apart from the essential ones, some further were presented, which extend the overall function space of the prototype and, thus, supply additional information and support for potential users. In doing so, further requirements, as they were defined in Table 3.3, were fulfilled. This also applies to the multi-tenant support presented in section 5.5.1, the knowledge access shown by ontology viewer in 5.5.2, and the inner tool encapsulation.

All of the remaining requirements, in particular the role concept (RC), the knowledge base editor (KE) as well as the information distribution among user groups, were realized only in parts. While the MongoDB collection and some classes partially integrate the RC in the prototype (e.g., via the admin view `UserManagementAdminPermissionControlUI`), basic manipulations of the knowledge base are facilitated concerning only implemented tools. However, both of them are only basically implemented and not usable at the current moment.

An application of each component was presented in isolation but also in the context of the targeted *end-to-end* process. However, as touched upon several times, the content shown here represents only a small fraction of the whole. Therefore, it is necessary to refer to the extensively commented source code and the documentation created using Doxygen for further information. With the help of these, the far-reaching interrelationships of individual classes and methods become visible. Due to lack of space, further explanations of these were omitted here.

Notwithstanding that, with some minor exceptions, as mentioned above, all requirements and characteristics are fulfilled, as they were placed on a DSS in general as well as the targeted solution in particular. Another time, this is also specified within the adhering evaluation. In summary, even though the current implementation only serves as a prototypical implementation of the developed framework, it offers a good starting point for future use and extensions. Details in which way the prototype can be changed and extended are described in the discussion part of the concluding remarks, in chapter 7.

6 Evaluation

In the course of DSR, the evaluation represents a decisive step that examines the construction of the artifact as such, and its applicability, as best as possible [SV12a; SV12b]. However, the implementation of the evaluation is not identical for every artifact. A generalized, universally applicable procedure does not exist and is not feasible, which complicates the overall approach and choice of methods [PCA14]. Instead, many constructivists in this field provide hints and recommendations on how this can be done in the most rigorous and structured way. As described in the first chapter, many see the need for a continuous evaluation to check individual design decisions, partial implementations, and their usability over the entire period of artifact creation. Typically, this incorporates a basic distinction between ex-post and ex-ante evaluations [SV12b; VPB16].

According to Hevner et al. [HMP+04], the generated artifact may look very different and could be, among other things, an algorithm, a special program, a framework, or even a complete information system. In the IS domain, the system idea and thus the component-based and the structured view is often used as a starting point to conceptualize and create artifacts [NCP90]. Yet, according to Prat [PCA14], precisely this approach constitutes a solid starting point for creating possible evaluation criteria along the system dimensions.

After detailed investigations, based on existing contributions in the DSR field, he identified numerous criteria and generally applicable evaluation methods that can be used to support this process. While most of them are similar to the ones provided by [SV12b], additional ones are stated, mainly focusing on the developed system. An overview of these criteria can be found in Figure 6.1. Given the stated terminologies, it is not only the pure application that is necessary for a complete evaluation. Instead, implementation and external assessments are also required.

While the scientific artifacts behind all of the identified components were extensively evaluated using different methods, this has been done for the entire DECIDE framework by means of a prototype implementation. In this context, however, some of the aspects described in Figure 6.1 have not yet been fully checked. Among other things, this includes the overarching *structure* and the *activities* carried out.

All necessary steps for the successful evaluation of the individual dimensions will be presented in this chapter. For the required structured approach, which also refers back to the individual components and summarises the continuous evaluation carried out *ex-ante* and *ex-post*, the *Eval* patterns described by Sonnenberg and vom Brocke [SV12a] are used,

as introduced in the first chapter.

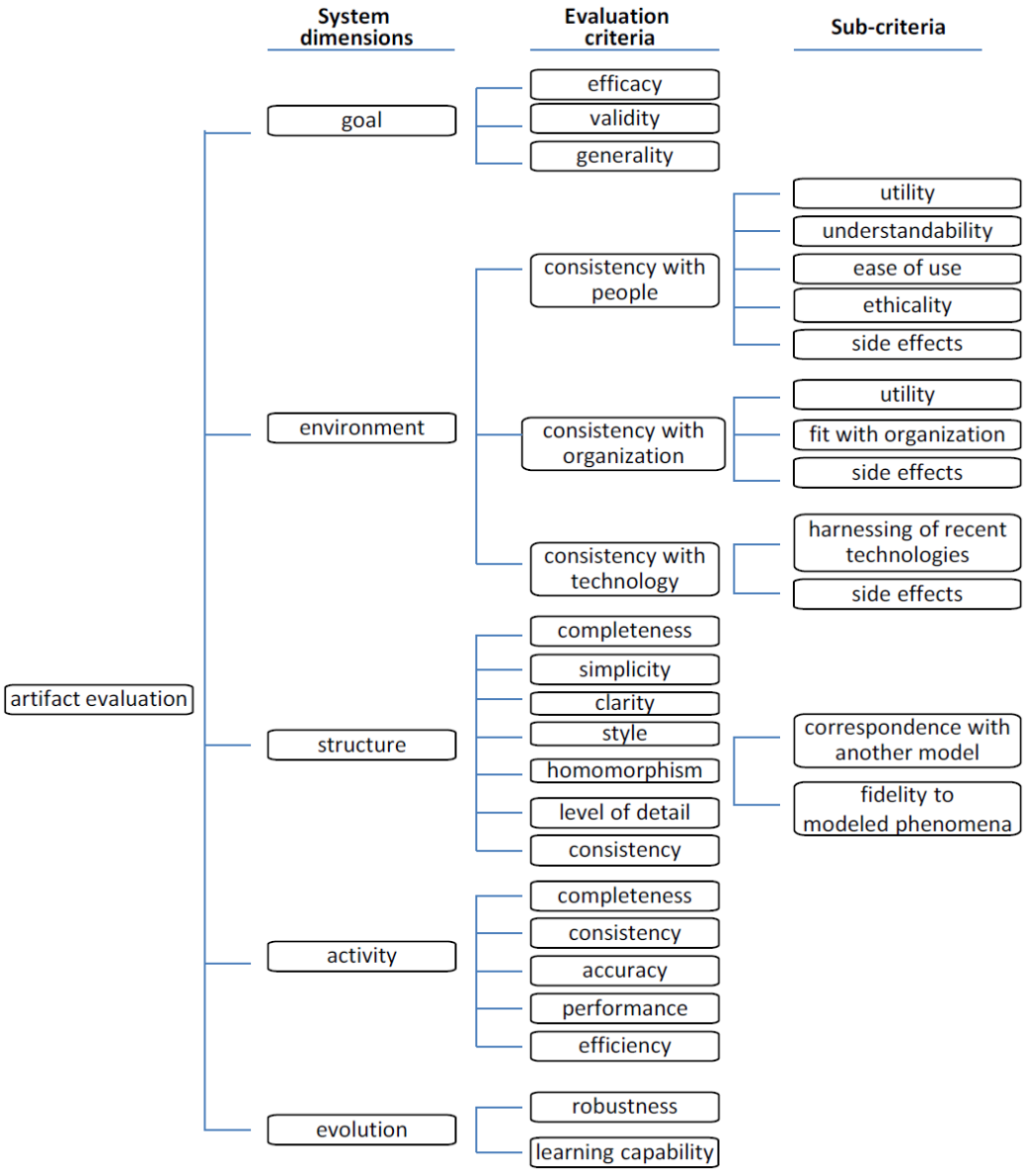


Figure 6.1: Evaluation Criteria based on [PCA14]

In addition to the structured evaluation, the aim of this rigorous methodology, which was already implicitly applied in the course of the work, is to check the relevance of the approach as well as the design in the early steps of the DSR. Although this has already been done during the complete research project, the content necessary for Eval 1 and Eval 2 is described in Sections 6.1 and 6.2. For the overall evaluation of the addressed solution, referring to Eval 3, extensive tests as well as expert interviews were conducted, which are presented in the main section 6.3.

The Eval 4 step proposed by Sonnenberg and vom Brocke provides for implementation and instantiation within a real-world context, where the PoU can also be evaluated concerning long-term monitoring. Due to the problem of a (still) prototypical implementation as well as the general willingness of larger and suitable companies, this step could not yet be implemented at the current time. Nevertheless, with individual questions during the interviews, it was already possible to partially investigate what general user acceptance might look like.

6.1 Evaluation of the Research Endeavor (Eval 1)

Eval 1 is intended to first justify the general problem, the resulting research gap, and the design objectives of a potential solution, raised by an existing problem [SV12b]. For the conducted DSR, in the contribution at hand, the problem was discussed in detail in the first chapter using existing literature and statistics. Most aspects described there were also confirmed by the expert interviews in section 6.3.2. In particular, as the general problem, the complexity of a big data project realization as well as their implementation, denoting related architectures, was highlighted.

As a consequence, comprehensive knowledge about the general domain and related technologies forming the cornerstone of each project was emphasized as a crucial requirement. Due to the constant changes and the numerous innovations emerging year by year, primarily referring to existing technologies, their selection, and implementation, there are often large gaps in the required knowledge. Eventually, this leads on the one hand to an increased demand for experts, while on the other hand, the available number of those is decreasing. Hence, based on that, most of the currently conducted big data projects are still less successful than anticipated. SMEs are mainly confronted with this situation. They can establish countermeasures only with great effort that often require large monetary expenditures, such as for training. In the long term, this can negatively impact the company and thus have a long-lasting effect on their evolution.

As a solution to this, a comprehensive end-to-end approach was suggested, which should deal extensively with implementing such projects and cover the necessary knowledge. Although there are isolated approaches that can support this, no satisfactory solution was found that solves the addressed problem. Due to the complexity of individual activities when it comes to the realization of big data projects and the related construction of the required system, the manual collection of relevant information is difficult and time-consuming. Hence, the idea was proposed to provide assistance with a computer-based solution that is easy and intuitive to use. In the course of this, design objectives were therefore successively created and implicitly described in the second chapter, which mainly refers to BDE, the creation and use of DSS as well as information management by means of ontologies. Throughout the description of the relevant information of these domains, the importance and novelty were repetitively highlighted, e.g., by the NTCP

model (cf. Figure 2.8). The information itself originated from previous discussions and literature reviews of already published articles, which can be found in Table 1.1.

6.2 Evaluation of the Design of the Artifact (Eval 2)

Eval 2 takes place after the first creation of the design specifications, in which those, the corresponding design objectives, and design methodologies are evaluated. Basically, this step is based on the proposition that the generated artifact was neither created nor instantiated yet, and only the mentioned inputs are examined. Nevertheless, at each point in time, not only was an attempt made to present each design decision transparently and fact-based, but also to evaluate it consecutively. Thus, the required criteria [SV12b], such as comprehensibility, clarity of the made decisions, completeness as well as a high level of detail of information, should be fulfilled in this step. In this sense, justifications for Eval 2 can be found extensively in chapters 2, 3, and 4.

Specifically, this includes, first of all, the end-to-end process (cf. section 3.3) that is based on the findings of BDE (cf. section 2.2.6). Apart from that, it forms the overall foundation for the main artifact. For the development of the process, in a comprehensive and comprehensible manner, established SE methods were applied, including the creation of a use case diagram that emerges from the description of a potential scenario. It was also in later stages evaluated by means of expert interviews (cf. section 6.3.2). In addition to this subsequent evaluation, within Eval 2, the individual specifications should be evaluated, for instance, by demonstrations, simulations, or formal proofs. In fact, for each of the components derived from this process, an independent evaluation took place (cf. sections 4.1.4, 4.2.5, 4.3.3, 4.4.3 and 4.5.3). This results from the fact, as already mentioned at the beginning of the fourth chapter that each component was developed through an independent DSR. According to this, all artifacts were constructed and evaluated by means of dedicated methods and patterns.

Regarding the former, extensive literature reviews took place for all components [WR02; LJ06], whose results were not always completely repeated in the present work. Additionally, for the technology application (section 4.1.) as well as standard use case comparison (section 4.2), case study research was conducted [Yin09]. For the development of the ontology, however, recognized methods of ontology engineering were followed [IMM+13; BS05; FGJ97b]. The structured approach in the creation and evaluation of each component leads not least to the fact that the superior design specification of the end-to-end process as well as the choice of the supporting system could be proven. Formally the necessity of a DSS, which is able to cope with the extensive knowledge in this area and supports potential decision-makers, was stated recurrently. Hence, jointly, it was shown that a complete implementation is theoretically possible, can be applied to real-world problems, and contributes to solving them.

6.3 Evaluation of Prototypical Implementation (Eval 3)

Compared to the other two evaluation steps, in this work, Eval 3 is much more comprehensive. This is not only due to the fact that many of the evaluations already required for Eval 1 and 2 were carried out in the course of the work, but also that in this step the focus is primarily placed on the developed solution and its instantiation. According to [SV12b], the latter refers mainly to a prototypical implementation. Extensive evaluations of the generated components have already been carried out in the previous sections and chapters. However, it is important to note that this was mostly done on an isolated basis and specific to the component. Furthermore, implementations were rarely undertaken. Section 6.3.1 will, therefore, first evaluate the instantiation of the DECIDE framework, which contains both the end-to-end process and the individual components. The aim is to be able to make essential statements about the applicability, usability, robustness and the dimensions mentioned in Figure 6.1. The following section 6.3.2 then deals with meticulously conducted expert interviews, which are intended to provide an additional assessment of the developed solution. In addition to specific questions about the prototype itself, the focus is again put on essential points that deal with the process, the individual components and the DECIDE framework as such. As a result, the observations made in Eval 1 and Eval 2 can be externally validated once again *ex-post*.

6.3.1 Project-based Application

During the development of the individual components, extensive tests were carried out on each of them before. to further assure the quality of the developed artifact, in the third evaluation phase, a prototypical instantiation was validated in an artificial setting. Specifically, this involves the end-to-end process mentioned in sections 3.3 and 5.6. Due to the problem that many documented and published projects rarely disclose detailed information (cf. section 4.2), resources were used that had already been used in the context of the work. Specifically, these are extensively documented use case descriptions found, inter alia, in scientific articles. The comprehensiveness was extensively tested in advance, using a modified version of the template from [CG18] (cf. [VST+20]. This resulted in 48 use cases that are suitable for testing the end-to-end process as such. Since the requirements are partly defined in detail in these and do not only contain the pure idea of the project, they constitute an adequate basis. In concrete terms, three different UCs were randomly selected, each belonging to a different SUC and using technologies that are already mapped in the knowledge base. Namely, these are, [ZZW+18; YMR+17; SKC16]

Due to the already given project idea and further details in each of those contributions, the UC comparison component is not required; thus, the first combined fragment in the sequence diagram from Figure 5.24 is skipped. The same applies to the subsequent modeling and deployment, as these have already been tested on their own in advance,

and treated within the process as optional. Hence, only the core process was focused. The literature often addresses the problem that DSS cannot be evaluated very well since they attempt to solve semi-structured and unstructured problems for which an answer can usually only be seen as a recommendation and not as an absolute execution directive. The continuous bias that already arises during the input of data into the system also has an influence on the results of the system. Accordingly, a classification of the latter should rather be carried out in good, bad or reasonable [RR08].

Despite this, an attempt was made to collect all information on the corresponding FRs and NFRs as comprehensively as possible. Especially concerning the NFRs, this was not trivial. It was already described in advance that the characteristics of each individual UC were collected and later used as a basis for calculating the average value of each NFR of the SUCs (cf. section 4.2.4). Although a corresponding classification was carried out with the addition of Table A.5, there may still be a certain bias. Nevertheless, an overview of these, according to the ratings from one (lowester) to five (highest), is presented in Table 6.1

T/NFR	**AV**	**CC**	**C**	**DO**	**FS**	**FT**	**IM**	**RE**	**R**	**SY**	**SC**	**S**	**UI**
[ZZW+18]	5	5	5	5	5	3	5	3	4	5	5	3	5
[YMR+17]	5	5	5	2	2	4	5	4	4	5	5	2	5
[SKC16]	5	5	5	5	5	5	5	5	3	5	5	3	5

Table 6.1: Mapping of the UCs and fulfillment of the NFRs

The corresponding input matrices, necessary for calculating the MCDM (cf. section 4.4.2), were created using Table 4.11. All investigated use cases are listed in the appendix due to their size (cf. Table A.10). Typically, this transformation step is omitted because the user performs the comparisons independently. Furthermore, all functional requirements were collected through the meticulous analysis of the related articles. A summary of all of them is shown below, in Table 6.2. The input was created using the given flow as presented in section 5.6.

Overall, the technology combination identified in the contributions could not be found in the exact same form for any of the given use cases. This may be due to several reasons. On the one hand, there is the possibility that errors occurred in the inputs with respect to (i) NFRs as well as FRs, (ii) the developed algorithms and stored data are incorrect, or (iii) that the decisions made in the contributions represented only one possible expression.

A closer comparison of the determined solutions and the results provided here revealed the latter. For example, a whole range of big data technologies was used in the project described by [SKC16]. Specifically, these are Giraph, Hadoop, Hive, Mahout, Pig, Spark, Lucene , MLib, and Spark.

Article	SUC	FR	Technologies
[ZZW+18]	1	RT,ST,F,SU,SS,BP,EP,CL, V,SL,CU,RP,P,DM,SS,ML	Kafka, Storm, HBase, Spark, Redis
[YMR+17]	3	ST,F,SU,SD,BP,MH,CL,V, CU,P,DS,CF,SP,SS,ML	Hadoop, Spark, GraphX, MLib, Giraph, Mahout
[SKC16]	8	RT,ST,SU,SD,BP,CL, CU,MO,CF,DM,SS,ML	Giraph, Hadoop, Hive, Mahout, Pig, Spark, Lucene, MLib, Spark, Neo4j

Table 6.2: NFR ratings of the individual use cases

In contrast, the system provides a wide range of possible combinations. Among them are often those that also include the ones used by the paper´s authors [SKC16]. For example, Pig and Kinesis (1.766%) were identified as the combination with the fewest possible technologies. The value in parentheses corresponds to the recommendation by the use of the AHP. Furthermore, Storm and Pig (1.7%), Hadoop, Hive Flink, and Infosphere (1.7%), as well as Girpah, Pig and Neo4j (1.69%), were additionally identified. A direct comparison of the individual tools and their degree of fulfillment of FRs clearly shows that the comprehensive solutions Kinesis, Infosphere, and Neo4j cover a large number of functionalities and thus represent an adequate replacement for many combinations (cf. Table A.6).

At the same time, it should also be mentioned that the NFRs determined here refer mainly to interpreting the given information and sometimes include only very few pairwise comparisons. The authors might have preferred other NFRs in each investigated case, which would make other solutions much more prominent. This is, for instance, the case for preferences that have a very high value for the cost and accordingly prefer technologies that do not impose further monetary expenses. Extensive platforms, such as Neo4j would be much less considered.

Nevertheless, similar observations were also noted in the other two considered cases [ZZW+18] and [YMR+17]. Kafka, Storm, HBase, Spark, and Redis were used in the former. Alternatively to these, the combination of HBase, Pentaho, Spark (1.725%) and Hadoop, Infosphere, Neo4j, and Spark (1.806%) could be identified. In [YMR+17] a lot of specified solutions were used, more precisely Hadoop, Spark, GraphX, MLIb, Giraph, and Mahout, a combination of Hadoop, Nifi, Pentaho, and Storm (1.923%) was considered equivalent.

Although the examination of the individual cases showed that in no instance an exactly replicated technology combination was proposed by the system, various combinations with a small number of technologies produced a similar result. If certain technologies would be preconditioned already at the beginning, the same recommendations are to be expected, as

presented here in the contributions. Although the functionality is not a complex problem, it is not yet included in the prototype. However, it could be a possible addition for future research. Furthermore, it should be mentioned that the found combinations only constitute possible options. For each of the use cases, about 30 different combinations could be found. This is also the reason for the partly low recommendation value, which in sum always results in 100%. For better clarity, these values could be normalized in the future.

6.3.2 Expert Interviews

In the previous section, a successful evaluation of the overall applicability of the developed prototype was showcased. Although this was mainly performed focusing on the main steps of the end-to-end procedure, covering only the most essential components, the efficiency, validity, generality, completeness, consistency, and accuracy were proven so far. To analyze the remaining criteria beyond that, external opinions were ascertained. Through semi-structured interviews, an external assessment of the developed framework, components, and the developed prototype was conducted. In doing so, the focus was primarily on the remaining criteria (cf. Figure 6.1), including the *structure*, *environment*, and *evolution dimension*.

However, related questions regarding the overall design and setup were also asked during the conducted expert interviews. Apart from the overall appearance and visual representation of the developed prototype, this included the questions related to the construction of the single components and used methodologies.

The overall setup, including the formulation, use, structuring, and realization of the interviews, was heavily oriented on best practices, given by [KPJ+16; Ada15; HA05]. A total of five interviews was conducted and recorded face-to-face via Zoom with experts from a wide range of fields. All of them were very different not only in their areas of responsibility but also in their professional experience and previous careers. Besides one person with a Ph.D., having the highest level of education, who is now active in consulting, company founders, a digitization officer for SME, and experts in systems and big data engineering were interviewed.

Many of the candidates had additional experience with the selected tools, which were directly related to the developed prototype (e.g., Vaadin). While some had a large amount of big data experience and are working in this field, others dealt selectively with individual big data technologies or had no knowledge at all. The goal was to evaluate the usefulness, methodology, and solution as comprehensively as possible. All information about the candidates was obtained from the first question, which targeted general information about their background and current role in their company (cf. 6.4).

ID	Highest Degree in Academia	Position	Exp.	Main Area of Expertise
A	Doctor of Engineering	Consultant	9 years	Consulting, System Engineering, Consulting, Science
B	Diploma	Consultant	20 years	Digitization Consultant, SME, Company Founder
C	-	System Architect	11 years	BDE, SE, Product Manager, Company Founder
D	Master of Science	Big Data Architect	9 years	BDE, SE, Project Management, Consulting Science
E	Master of Science	Big Data Architect	15 years	Web Development (Vaadin), Software Engineering, BDE, SE

Table 6.3: Overview of the interviewees

Candidate A demonstrated the highest level of a formal education with a Ph.D. in Computer Science. In addition to many years of practical experience in his current employment as a consultant, the candidate also has an extensive scientific history. Scientifically and now practically, large intersection points were determined, which concern the general applicability, the conceptual design by means of the DSR, and the individual components.

Candidate B has a diploma and the longest professional experience, with currently more than 20 years, which is also reflected by the field of activity. He already has extensive experience with the selection and implementation of software solutions, especially enterprise resource planning tools. In addition, in his current position, he is a consultant for digitization topics, especially for medium-sized companies. He is the only one who does not have a connection to big data, but thoroughly inherents expertise in the last topic mentioned (SME).

Candidate C has a wide variety of work activities, all related to software development, web development, and product management. He is the only one without a degree, which comes from the fact that he was already recruited away during his studies. In addition to numerous freelance activities, this candidate has also founded his own company using Infrastructure as Code (IaC) concepts. With IaC, architectures can be created and deployed automatically using various tools and machine-processable files [ABD+17]. In particular, according to the candidate, Docker and also Ansible played a big role. The current role as a system architect with a solid connection to big data projects is also about selecting and using individual technologies.

Candidate D has a similar experience as Candidate A, both in practice and in academia. In the current role as a big data architect and project manager, this candidate is responsible for system planning and conceptual design in big data projects in the insurance industry. Previously, he worked as a consultant in one of the largest consulting companies

in the world. Due to his extensive prior knowledge and professional experience, many areas of the built DSS could be covered and also comprehensively assessed.

Candidate E, like D, has a Master of Science. With 15 years of professional experience, especially in software and systems engineering, he has the second longest professional career. At the current time he, like candidate D, is employed as a big data architect. Furthermore, he has extensive experience in software engineering, in which web development has also played a major role, and he has gained experience in the use of various tools that were also used for the direct development of the prototype. Besides MongoDB and PlantUML, he is the only one with extensive experience with Vaadin.

While care was taken in the selection of candidates to ensure that each would cover certain areas of expertise to some extent, this could not be fully guaranteed. For example, none of the candidates had any experience or prior knowledge of ontologies, which limited them to rely on the information given in the presentation.

Initially, a running time of approximately 60-90 minutes was estimated for each of the planned interviews. However, the actual length varied greatly. While the shortest interview with candidate B lasted only 73 minutes, the longest with candidate D was 125 minutes long. These strong deviations can be explained not only by the time spent on the initial presentation, in which some of the candidates already had follow-up questions but also by the varying complexity of the answers during the questioning. Some of the candidates answered relatively quickly and succinctly, whereas others needed more time for deliberations. In the latter cases, most of the answers were more extensive.

The addressed presentation was sent to the participants in advance of each interview so that they could already get a comprehensive picture. Among other things, it contained essential background information on motivation, problem definition, research methodology, and background. Furthermore, all components that were scientifically investigated in advance were presented extensively. All 42 slides can be found in Appendix D, specifically Figure A.4 - A.7. In the last part, starting from the 23rd slide (cf. Figure A.6), numerous overviews and descriptions of the developed prototype were shown. During the presentation, this part was replaced with a live demo. All functionalities, the created knowledge base, and parts of the source code were shown extensively.

After the presentation was finished, the semi-structured interview took place. Unlike the presentation slides, the questions were not sent out in advance and should ideally be answered unfiltered in a direct face-to-face conversation. The interview included a total of 27 questions and is shown in Table 6.4. Each of these questions is assigned to a specific category in order to avoid thematic jumps. The semi-structured nature of the interview sometimes left room for follow-up questions. This was done whenever it was appropriate to formulate follow-up questions to specific responses. Regarding the structure of the questionnaires, some *opening questions* were asked first, which are often recommended to prepare the interviewee for further ones. Besides the already mentioned information about the person, the experience in the field of big data was asked.

Area	Question	Type
Opening Questions	1. Could you please elaborate your background and current role? 2. What is, in your opinion, big data? 3. How much experience do you have with big data technologies? 4. How many projects did you conduct so far that utilized big data technologies?	Open
Warm-up Questions	5. What are the most critical points when conducting a big data project? Think of all aspects, including technical, organization, and even human-related facts. 6. Do you use any additional tools or external assistance (e.g. experts) for the planning of your projects? If yes, which in particular? 7. Would you personally use and/or recommend supplementary solutions to increase the likelihood of potential project success? Please elaborate.	Open
Artifact Questions	8. Does the end-to-end decision support procedure cover all relevant steps in a sensible order? Please elaborate. 9. Are you performing any quick checks to identify whether big data technologies might be required for a specific project? If yes, what is your opinion about the created application check component? Otherwise, would you use the described component in the future? 10. Would you say that an ontology is a suitable depiction of the knowledge base? Do you miss any information located there? 11. Do you think all relevant (functional/non-functional) requirements are covered yet for selecting big data technologies? Are you missing any? 12. Would you check the proposed standard use case descriptions and rely on the given information to get an idea about similar solutions for your own endeavor? Are you missing any? 13. Would you say that the FRs, NFRs, their severity, and an MCDM approach are sensible for identifying suitable technologies and their combination? Do you miss anything here? 14. Would you use the modeling component? If so, what purpose would it have? Do you miss anything here?	Open

Table 6.4 continued from previous page

Area	Question	Type
	15. Would you use a deployment solution to create sandbox environments for your project? If so, what purpose would it have? Do you miss anything here?	
Prototype Questions	On a scale of one to five, where one means that the observed aspect is not satisfied at all and a five, as the highest value, indicates that this aspect is very satisfied. How would you rate the following: 16. The system fulfills everything to support the realization of a big data project. 17. The system offers an excellent visual representation. 18. The system provides excellent decision support. 19. The system provides lots of interaction points to the user. 20. The menu navigation is intuitive. 21. The system provides thorough feedback to everything. 22. The provided information within the prototype are useful. 23. The system delivers enough information to use it confidentially.	Closed/ Rating
Feedback Questions	24. Are there any critical components or functionalities you are missing in a system like that? Also, think about auxiliary functions that could be generally sensible and not exclusively for your purpose/company. 25. Would you recommend the standalone usage of developed components and their structured use in the end-to-end procedure? 26. Which potential users could be interested in using the framework and the developed prototype? Think of single roles, job descriptions, companies, and organizational forms. 27. Do you have any other comments regarding the process, framework, system, or the development?	Open

Table 6.4: Questions in the conducted interviews

Following this, general *warm-up questions* were raised, with the help of which the interviewees were supposed to give the first impression and introduce them to the investigated topic. The focus was on the general usefulness of the created solution. The third section dealt specifically with the *artifact* of the work. In a structured way, essential ques-

tions were asked about each component, not only regarding the general applicability but also the implementation. Furthermore, in most cases, due to the semi-structured nature, it was already attempted to determine to what extent certain aspects were missing in each of the developed components or could be improved otherwise. During the questioning, the theoretical contents on the slides as well as the practical implementation using the prototype, were shown alternately.

As already highlighted, stated in section 6.3.1, the evaluation of DSS is commonly a sophisticated endeavor, where given recommendations mostly depend on a high number of criteria. In this case, apart from numerous other information, the user's preferences are considered. Typically, these have a certain bias. For that reason, the questions listed in the fourth block contain *closed rating questions*, mainly focusing on the aforementioned criteria from Figure 6.1, at which the interviewees had to provide a rating, which ranges similar to the NFRs used in this work (cf. section 4.4.2), between one and five. While *one* expresses the lowest possible satisfaction of a given fact, a *five* represents the best possible implementation.

Finally, in the fifth part of the interview, concluding *feedback questions* were asked that once again dealt comprehensively with the content taught. Here, the interviewees were asked to provide information on the extent to which an expansion and change would be useful, whether additional content should be created, and what aspects they consider to be particularly critical. In connection with this, another referral was made to the applicability. In particular, potential stakeholders had to be identified and the likelihood of a possible application assessed. The last question allowed each of the interviewees to draw a final conclusion. Ultimately, the first and last questions validated key aspects that formed the baseline of the DSR as it is shown in Figure 1.1 and highlighted why and for whom the results produced here are actually helpful. Before starting with the first interview, assessments and tests regarding the formulated question, structure, and estimated time were performed [KPJ+16].

Question 1 – Interviewee Information

Through the use of this question, general information about each of the interviewed experts were obtained.

Question 2 – Definition of Big Data

The second question was aimed to investigate the basic understanding of the topic of big data by each of the interviewees. Although everyone had at least some sense of what big data is, potential definitional approaches varied widely, leading to significant disagreements among the respondents. All candidates were aware of this, which caused them to go for *"best practices"* at this point. Candidates B and C referred here to *the 6 known Vs, of which 3 are particularly important and provide a plethora of new challenges.* Implicitly, candidates A, D and E have also stressed this. The given definitions by the experts were, thus, similar at least to one of the approaches described in Table 2.1.

Questions 3 & 4 – Big Data Experience

With regard to the third and fourth questions, all but candidates A and B stated that they already had experience in dealing with big data technologies and dedicated big data projects. However, there were also fluctuations here, ranging from a few months for candidate C to a few years for candidate D. However, it also turned out that there was a disagreement as to when a big data project was actually being referred to. This problem was addressed by candidates A, C, and E. Candidate C stated that the projects already implemented, despite the use of tools, were possibly not big data projects *"although the tooling says something different"*. In this context, the reasonableness component was, therefore, positively addressed independently several times here, which can serve as a kind of *"quick check"* to identify whether it could be a big data project and a combined use of big data technologies would make sense.

Question 5 – Critical Points

The fifth question, which deals holistically with the critical aspects in such projects, was answered differently by each candidate. Candidate C generally emphasized the *"extremely difficult plannability"* of such projects. In his opinion, *"agile concepts"* in particular are challenging to apply, because the complexity means that well thought-out planning is essential *"in which it is already clear at the beginning what the end result should be and what the intermediate steps look like"*. A holistic requirements analysis is, therefore, necessary from the very beginning. At the same time, *"communication between the business and technical sides is complicated"*, which is mainly due to the complexity and scope of the domain. Often there is an *"extreme knowledge gap"*, complicating the communication and implementation. On the technical side, the deployment is also addressed here, which is particularly complex. Ready-made *"pipelines"* and also *"solutions where no increased planning is necessary in advance [...] and can be deployed easily would be great"*. The problem of adequately identifying, configuring, and deploying the technologies was also reflected in this way by candidates A, B, and D. In particular, according to candidate D, determining the feasibility of individual functionalities (as identified in Table A.6) is only possible with a great deal of effort and detached prior knowledge. Eventually, he sees missing knowledge as one of the biggest problems.

Questions 6 & 7- Assistance

No additional tools addressed by the 6th question regarding the project planning were used by any of the interviewees. Candidate D carries out a similarly structured and systematic procedure in the company, with which architectural components are successively determined, and individual FR are taken into account, but this is carried out purely manually and without other computer-aided solutions. Nevertheless, every decision made there is meticulously documented to allow better traceability later on. Although no such solution

is used by any of them, all respondents to the 7th question agree that the prototype developed here is a valuable and welcome option for project implementation. Candidate B also independently mentions SMEs that particularly benefit from it. Candidates A and D emphasize the holistic implementation, which is meaningful for decision-makers who do not have extensive knowledge. In addition to these aspects, candidate E also mentions the need for the given recommendations to be *"manually cross-checked"* once again.

Question 8 – End-to-End Procedure

As mentioned above, for artifact-related questions slides and the the prototype were shown. Concerning the end-to-end processes and thus the 8th question, the BPMN diagram from Figure 3.2 was displayed. Candidate C once again stated that this one is *"pretty cool and well thought out and makes perfect sense from my perspective"*. In this context, Candidate E even mentioned that the corresponding process is used one-to-one in his company. Candidate B also *"didn't find any gaps worth mentioning"*. Only candidate A criticized that despite its rigorous creation, it can sometimes become too confusing. A sequence diagram would help here, but this has already been developed independently and not discussed (cf. Figure 5.24). According to candidate D, another aspect that would be useful for future process iterations would be, *"to feed the decision back into the knowledge base to trigger a re-training"*. While the inclusion in a catalog of already made decisions makes sense, concrete retraining is currently not planned. This is mainly because, currently, no ML approaches are implemented.

Question 9 – Big Data Technology Application Check

Some of the identified and implemented components had to be described in more detail. The reason for this can be different. Although the slides were sent to each user prior to the interview, they may not have been reviewed in too much detail in advance. At least candidates A, C, and E stated this directly. Nevertheless, this allowed expansions of the individual questions. With regard to the 9th question, for example, the conception and the individual characteristics of the developed hexagon could be dealt with in more detail (cf. Figure 4.2). In this respect, the general idea was praised and the approach was considered useful by all candidates. However, candidate C pointed out that it can be difficult if it is not clear where a big data project starts and where it ends. This could be the case *"when all characteristics are very high, and volume is low"*. However, individual technologies can still make sense here, as not all are aimed solely at mass data processing. However, as *"a first starting point to making sense, this sounds like a reasonable approach"* (candidate C).

Question 10 – BDTOnto

The ontology as a knowledge base, which is addressed with question 10, was commended by everyone. Above all, the basic idea of the extensibility, regarding the technologies and further domain knowledge that can be brought in there, was called *"elegant"* (candidate

C) for such a solution. Parallel to the corresponding slide, Protégé was opened, as well as visually shown and discussed using the internal plugins OntoGraf and OWLViz. The candidates A, C, and E were especially interested and wanted to get additional information about the general creation of such ontologies. At least candidate A knew about Jena from previous work and emphasized the usefulness of integrating the ontology into the prototype (cf. Figure 5.1). However, none of the candidates had a deeper understanding of it. As a result of the extensive presentation, answering the questions took much more time than originally expected. However, there was no concrete idea for an addition that the candidates made.

Question 11 – FRs & NFRs

A similar observation could be made about all FRs and NFRs, as asked by the 11th question. Each participant indicated that these were very comprehensive and had no gaps that could be identified ad hoc. Candidate B indicated, however, *"that possible changes could emerge if increased engagements are made in organizations"*. These could be then even more specific compared to the given ones. This is also implicitly addressed by Candidate D, who also asked distinct questions with regard to his company. In connection with this, it indicated that certain requirements could be specified further and perhaps also be decisive with the technology selection. This includes, for example, the costs, which are at the moment only covered by the related NFR and thus individual weightings. Candidate A would also like to see a finer degree of detail.

Question 12 – SUC Comparison

The SUC comparison component was seen by all five candidates as a good addition in the event that there is little to no experience in the area of big data and information about the potential project. Candidates A and B indicated that they would like to see additional enhancements, for example, the integration of projects that have already been carried out. In complete contrast to this, candidate D pointed out that too much information could possibly have a deterrent and obstructive effect. A balanced information content is therefore essential here. Notwithstanding that, they all highlighted the sensibility of such an approach, especially in earlier stages, when no clear understanding is available.

Question 13 – Inference Engine - MCDM

The inference engine, which is responsible for the identification and possible combination of the individual technologies, was considered by all participants as a good idea to consider both FRs and NFRs. As stated by candidate D in an earlier question, in his company they use a similarly structured procedure, however, neglecting the application of valid scientific methodologies. The other candidates appreciated as well the usage of a MCDM method. In the future, as a useful extension, the pre-limitation of candidates C and D was suggested. In concrete terms, this means that individual technologies could be pre-selected if,

for example, they are already in use in the company. In this context, only recommendations would come that already consider individual technologies. According to candidate C, the inclusion of this setting could thus also greatly increase the circle of users in the future. System architects who are permanently under time pressure and could only think a little about potential system alternatives or their extension would presumably have an increased interest in it. This also applies independently to the deployment component. As an additional extension of the inference engine, candidate D further recommends the extension of the given decision support. Specifically, for example, a skills matrix could be delivered, which might be used, among other things, to prepare human resource management and trainings. Overall, during the interview, initial ideas for future developments were given. Candidates C, D and E also picked up on the community idea, which would be desirable for future developments with regard to deployments.

Question 14 – Modeling

The additional conceptualized and integrated components that can be used to model and deploy the given recommendation were seen by all users as an important addition that ultimately completes the end-to-end process. The candidates often emphasized that additional diagrams can be helpful, most of all for large and complex integrations. Especially when it comes to coordination and communication, *"such architectures are often much more helpful than plain text"* (Candidate D). While candidates A, B, C, and E did not see any additions to this component as necessary and in some cases, described it as *perfect*, D still provided some ideas for enhancement. Specifically, it was about *"using more color for management"*, furthermore, according to him, it would also be desirable to integrate the previously defined categories that further structured the FRs, such as using *data analysis* or *data preparation* (cf. Table 4.10).

Question 15 – Rapid Deployment

The component developed for deployment received at least the same approval. Candidate B highlighted among other things that his *"would reduce initial barriers often encountered during implementation"*. Especially *"SMEs are often confronted with the problem here because the necessary know-how and people are missing"*. This was also stressed by Candidate E, who mentioned *"that not every company has DevOps employees"*, particularly concerning the idea of the playbooks and the external registry. Candidate C also sees the applicability by big data engineers and system architects, who often have little to no time for deployment. Rapid deployments can be put to good use here for initial testing. Although the solution is seen as helpful and good by all, candidate C, for example, pointed out the problem of configuration, which is not fully covered here. Instead, individual tools are only loosely coupled. Candidates C, D, and E also pointed out the backtracking and creation of more specific solutions, where these particular configurations of the tools are already covered, coming close to concrete architectures. Candidate D also expressed the

desire to integrate special tool versions and compatibilities, as there are often large variations in the range of functions and connectivity options. However, he sees the complexity and the effort that might be required.

Questions 16-23 - Prototype Fulfillment

Questions 16.-23. were primarily aimed at the above-mentioned missing criteria. Thus, not only the individual components as such but also the prototypical implementation should be assessed as well as the general perception of it. The numerical values for each question were summarized for each candidate in Table 6.5. While for most of the questions, only the values were given by the candidates, for questions such as related to navigation (20.), some indicated that individual users could be overloaded with information. Candidate E criticized the number of tabs, which could perhaps play an important role on mobile device role and reduce general usability. However, at the end of this statement he said: *"who use this on a smartphone? In many cases laptops or large tablets would be used"*. D emphasized additionally to question 22 that the addressed role concept makes sense here *"in order to not overload the users"*. However, for users in academia, this can be very helpful. Overall, it was found that good to very good results could be achieved with the exception of navigation.

Questions/Candidate Response	**A**	**B**	**C**	**D**	**D**	∅
16. Support BD Projects	4	5	4	5	5	4.6
17. Visual Representation	5	5	3	3	5	4.2
18. Overall Decision Support	4	4	4	4	5	4.2
19. Interaction Points	3	5	5	5	5	4.6
20. Menu Navigation	5	5	3	2	4	3.8
21. Provide Feedback	5	5	4	4	4	4.4
22. Information Usefulness	5	5	4	5	5	4.8
23. Confident Interaction	4	5	4	4	4	4.2

Table 6.5: Overview of the given rating for each question by each candidate

Question 24 - Missing Functionalities

During the previous questions, there was extensive feedback and some hints for potential functionalities, which made the final answers relatively short. Nevertheless, there were once again some relevant aspects here that might be of interest for future developments. These include, among others, the increased integration of data flows within the architectures to be implemented. Candidates A, C, and D requested this, for example, in the modeling and deployment components. Candidate B referred once again to the future community idea, pointing here to the potential extension of the knowledge base, the prototype, and the playbooks. D again highlighted the use of potential ML methods for future training and deployments.

Question 25 - Stakeholders

After all content related to the created solutions as well as the system itself had been comprehensively presented, the interviewees should identify possible stakeholders. Although some of the statements were shared by each candidate, the answers were sometimes very different. Candidate E addressed almost all groups of people *"who have to do with requirements in projects"*, including DevOps, admins, and architectures. He highlighted that startups and SMEs, in particular, would benefit the most here. Candidates B and C shared this view and emphasized the importance of SMEs. Candidate B went into further details about individual positions such as *"IT-affine managing directors or business process managers who deal with data exploitation"*. Candidates C and D also referred more specifically to large companies, where internal IT can also benefit, and system architects, for example, represent a large target group. D mentioned *"consulting firms"* and, in general, the role of the consultant, who, as a kind of solution manager, can already present initial approaches. In sum, according to each expert, many different people can benefit from the multitude of functions, as previously identified in Figure 1.1.

Question 26 - Comments

In the last question, the expert had the chance to provide some final remarks. In most cases, the solution as such was once again appreciated. Candidate B stressed another time the utilization of established concepts, which are familiar to many people, e.g., UML and GitHub. Only candidate E stated that functionalities for stress tests in the sense of load balancing would be helpful in the future. Cloud environments would also be desirable in context of this. Candidate D highlighted this already in advance.

Overall, it was possible to obtain the views of a wide range of experts by conducting the interviews. At the same time, useful comments and enhancements were ascertained as to how they could be expanded in the future and introduced into the existing prototype implementation. Consequently, especially the missing criteria focusing on *structure*, *environment*, and *evolution* were successfully evaluated.

6.4 Summary

The present chapter has dealt extensively with the evaluation of all artifacts created in this work. Initially, a brief outline of evaluation criteria and approaches was given, which were also used here. In particular, the presented Eval patterns were applied here that contributed significantly to the structure of this chapter. Gradually, all steps proposed by Sonnenberg and vom Brocke, with exception of the fourth, could be successfully evaluated. Although all components were already extensively tested throughout this work, additional experiments and expert interviews were conducted for the holistic test in Eval 3. Not only could the general functionality be demonstrated, but also other criteria, as shown in particular in Figure 6.1.

7 Concluding Remarks

In this thesis, a comprehensive end-to-end approach was presented, which can support the implementation of big data projects and focuses primarily on the creation of the required systems and the associated technology selection. By means of the DSR, a comprehensive component-based framework was successively created, prototypically implemented, and holistically evaluated. In accordance with the overarching research methodology, an initial motivation and deduction of the corresponding goals took place in the first chapter, in addition to the presentation of this methodology. Based on the problem that there are fewer experts than required that have extensive knowledge in big data, while more and more technologies, paradigms, and architectures constantly emerge, it was tried to find a structured solution that thoroughly helps with the corresponding selection process. Due to the comprehensiveness of the domain, the abundance of information, the necessary preliminary considerations, comparisons, and other peculiarities, a structured end-to-end sequence was assumed to be the most sensible, especially for non-experts. In this context, a purely manual execution was excluded, and the necessity of a computer-supported solution was determined. Subsequently, the objective was to answer the following main research question and its sub-research questions: *"How can end-to-end decision support be facilitated concerning BDE activities that assists decision-makers with selecting, combining, and deploying big data technologies in their projects?"*.

By harnessing also existing knowledge in this regard, essential information on SE, big data, ontologies, and decision support systems was presented in the second chapter. In doing so, not only basic theoretical information were recited here but also substantial findings already embedded, as they could be determined by numerous published contributions in advance. In particular, this concerns BDE, as it is decisively necessary for the execution of the conceptualization and creation of relevant systems. Starting from the information presented there and the previously formulated objectives of the thesis, the third chapter provided initial preliminary considerations as to how potential support for system design and thus the implementation of big data projects could be achieved. In addition to specifying the general idea of an end-to-end approach, further research was conducted to investigate, differentiate, and discuss other works in which potential solutions were suspected. However, despite a meticulous search, no approach could be found that entirely solved the addressed problem.

Although many of the ideas provided in the individual contributions appeared relevant

and could thus be used implicitly or explicitly, there was little detailed knowledge that was directly applicable. As a result of this paucity, an idea for such a structured process and the role of a potential system was identified step-by-step based on standard practices. By identifying and bundling necessary activities on the part of the user and the potential functionalities provided by the system, different components were determined to be further investigated.

In the fourth chapter, the referred components were then further investigated. Based on numerous scientific contributions, which were already published in the context of this thesis, each of the components was examined, and the foundation for a further application was built. Throughout the entire chapter, details about potential technologies were shared, and relevant requirements were identified to facilitate a selection of those. In the end, the DECIDE framework was created, which not only helps with the structured support and implementation of big data projects but also with the creation of a decision support system.

To verify both aspects, the end-to-end approach and the decision support system, a prototype was conceptualized and developed that serves as an instantiation of the DECIDE framework. This prototype was designed and developed according to common characteristics of such systems, as they were identified in advance. The structure, the selected technologies as well as the individual components were then comprehensively described in the fifth chapter. Although only a few Java code fragments were shown over the course of this chapter, references back to the classes and created methods regularly took place. For detailed information, the extensively commented prototype is to be mentioned, and the comprehensive documentation, which is provided digitally, besides the doctoral thesis.

During the identification and creation of the individual components, employing different scientific methods, extensive evaluations were already included in the fourth chapter of the work. Nevertheless, in the sixth chapter, a detailed evaluation of the solution in its entirety took place. The focus was not only on the DECIDE framework and the individual components but also on the prototypical implementation itself. In addition to the practical application of the instantiated framework utilizing the prototype, the latter was also evaluated qualitatively through the observation of third parties in the form of researchers and practitioners that are active in the big data domain. Five semi-structured expert interviews were conducted face-to-face that focused on the framework and its components, as well as the prototype. The initially formulated hypotheses were proven correct. The same applies to the deduced design decisions concluded from them, which were also successfully validated. Thus, it has been shown that the developed DECIDE framework, its components, and the developed prototype constitute a sufficient solution to the given problem, despite that the evaluation in an organizational context is still pending (Eval 4).

7.1 Discussion

In this work, the fundamental goal was to determine how the implementation of big data projects can be fully supported by means of suitable processes, best practices, and a computer-based solution. For this purpose, two hypotheses were formulated dealing with the end-to-end realization of such projects and the general applicability of DSS. In order to investigate these hypotheses, different research questions were defined. In the end, these SRQs should answer the following overarching question *"How can end-to-end decision support be facilitated concerning big data engineering activities that assist decision-makers with selecting, combining, and deploying big data technologies?"*

(i) "Which steps are required that support decision-makers in realizing their big data endeavors?"
Based on established processes in the field of data mining, SE, and the developed BDEP (cf. Figure 2.13), numerous activities were identified that are necessary for implementing big data projects. The investigation of preliminary scientific work and the formulation of a potential scenario led to an end-to-end process that must be assisted by a computer-supported solution due to the knowledge required therein (cf. chapter 3).

(ii) "Which requirements need to be considered when it comes to the selection of appropriate technologies for a big data system?"
Basically, two different types of requirements were identified that play a role here. By conducting a literature review and use case analysis, 29 FRs and 13 NFRs were identified and further categorized for the sake of a better understandability. By checking official documentation, research articles, and non-per reviewed documents, the degree of fulfillment of 58 big data technologies regarding those requirements was investigated (cf. section 4.2.4).

(iii) "What elements are required to create a computer-supported solution to provide decision support for big data projects?"
Based on the end-to-end process and general characteristics that a DSS should fulfill, numerous FRs were identified (cf. Table A.4). Due to the complexity and coherence of many of these, six components were defined, which were successively examined in chapter 4. Namely, these are the *Technology Application Check Component*, the *Use Case Comparison Component*, *Multi-Criteria Decision-Making Component for a Technology Selection*, the *Modeling Component*, the *Automatic Deployment Component*, and a *Knowledge Base*. Eventually, those were combined and aligned in an overarching framework called DECIDE that was proposed in section 4.7.

(iiii) "How could a computer-supported solution be designed and utilized that allows a semi-automated selection and deployment of big data technologies in related projects?"
While the DECIDE framework denotes the composition of the general components along the end-to-end procedure and, thus, the loosely coupled architecture, no specific implementation details are shared. The design for the computer-supported solution was partially described in chapter 3, specifying the important characteristics. The instantiation of the

DECIDE framework, concurrently following these, was proposed in chapter 6. Here, particular technologies, a deployment diagram, as well as the implementation of all of these are described.

All in all, it was possible to answer all research questions and the initially formulated hypotheses satisfactorily. The conducted DSR project, whose environment was previously set up and described in Figure 1.1, was also successfully evaluated. All points addressed in it were confirmed implicitly and explicitly by tests and interviews. In particular, for the basics and methodologies necessary in the knowledge base outlined there, it was possible to refer back to them constantly and their necessity in the course of the work.

Despite the rigorous and structured approach, it was impossible to fully implement all of the requirements discussed in chapter 3, which are particularly necessary for a prototypical implementation. These include, among other things, the FRs derived from the DSS characteristics, such as the role concept (cf. Table 3.3), which was also requested several times by the interviewed experts. During these discussions, it became clear that numerous suggestions for improvement at some points and functionalities would also still be desirable, which relate primarily to the modeling and deployment components. Many of the experts referred to the possibility of potential changes that may arise in a direct company application. At the same time, this points to the missing Eval 4 and thus long-term enterprise applications. Although the system could be successfully evaluated by means of extensive tests and interviews, this step is still needed, at least for the full evaluation. In the course of this, further optimizations and comparisons would also be interesting and conceivable, for example, concerning the benchmarking of different MCDM approaches. Despite these and the shortcomings highlighted in the previous chapter, numerous contributions could be achieved. Specifically, these are:

- A comprehensive overview of the domain of big data and the engineering of the related systems while revealing dependencies to adhering domains.
- An end-to-end procedure that comprehensively supports the realization of big data projects and the engineering of related systems in a structured way.
- The identification, investigation, conceptualization, and creation of various components emerging from scientific artifacts. These artifacts are related to relevant BDE activities and thus the end-to-end procedure, covering: the sensemaking of a big data technology application, SUCs, structuring and classification of pertinent information using an ontology, requirements engineering for big data projects, identification, selection and combination of big data technologies, as well as relevant FRs and NFRs of related technologies, a potential modeling approach, and an automated way to deploy the technologies.
- A framework for a component-based architecture of a DSS fulfilling the core activities of an end-to-end procedure.
- A prototypical implementation of a DSS that denotes an instantiation of the developed framework.

The description of a DSR project usually turns out to be very extensive, as is already the case with the work summary in this chapter. At any time, a balancing act between problem and solution space must be created, and all necessary considerations must be carried out sequentially and interactively. As a result of the complexity involved, it is easy to lose sight of what is essential [VM19]. Based on this problem, vom Brocke and Maedche proposed the idea of a DSR Grid, which highlights the most important points in a one-pager to provide a good overview to potentially interested parties. They distinguish the six dimensions for improved communication: *Problem Description*, *Research Process Solution*, *Input Knowledge*, *Concepts*, and *Output Knowledge*. Following those, a comprehensive DSR grid for the doctoral thesis at hand is described in Figure 7.1.

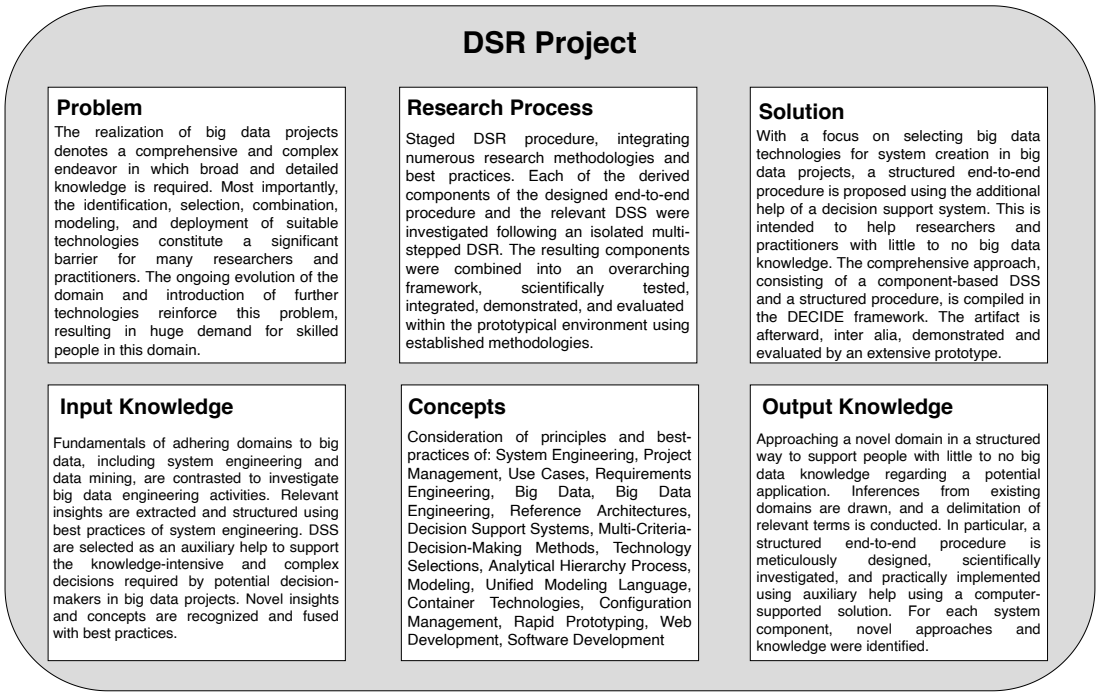


Figure 7.1: DSR grid based on the approach by [VM19]

7.2 Future Research

The derived SUCs in the area of big data represent a comprehensive picture of potential scenarios, denoting a short description, detailed information to each UC as well as relevant FRs and NFRs. However, some further information is not yet the subject of the work and, thus, also not the subject of the ontology. In particular, this concerns the individual technologies as well as the descriptions of the associated architectures. Recently, these SUCs were examined in another article [VSS+22] in more detail. Some of the results were already partially included in this work (cf. Table 1.1 - No. 20). In addition to the FRs and NFRs, individual technologies as well as concrete architectures were derived, which could be sensible for future integrations, such as in the context of case-based reasoning

approach to obtain technological recommendations alternatively.

Depending on a given problem of a potential user, not only similar use cases could be examined, as have already been implemented with the comparison component, but also possible technology combinations could be derived. Based on the degree of modification of these *tailor-made* architectures, they would then be added to the pool of existing solutions. Some of the interviewed experts shared similar ideas. For instance, created architectures could be traced back to the system as best practices for further deployments or future technology selections. Regarding the latter, one of the interviewees even proposed the idea of a machine learning approach for a technology selection once enough technology combinations were built, tested, and included in the system.

Depending on how the DSS is implemented, it would be sensible to include new information directly emerging from a particular organization, covering their specificities, or to open the accessibility in a cross-organizational way. With a focus on the latter, users from various domains could contribute their solutions back. Such a crowdsourced approach would possibly make sense and be conceivable in an open-source community but presumably be unthinkable for many organizations, at least if everybody would get access to all information. Above all this would result from the closedness and associated preservation of internal company data. The search has already shown this for corresponding use cases in sections 4.2 and 5.4.2. At this point, an elaborated role concept may help, limiting and regulating the accessibility to certain tools within the prototype, independently whether applied locally or globally. During the semi-structured interviews, it turned out that most interviewees support this idea and believe that it provides better usability, mainly originating from the reduction of unnecessary information that might be required for novice users.

The research on the individual big data technologies and their integration into the knowledge base of the prototype took considerable time and represented a general problem using ontologies. As discovered during the investigation of the different types of the big data technologies (cf. Table A.6 and Table A.7), most of these are compatible with each other, directly or indirectly, using additional libraries, adapters, or specific APIs. However, interoperability cannot always be guaranteed. To cover specific interconnections within the ontology, particular annotation and data properties were already assigned and exemplarily tested (cf. section 4.3.2). For the complete setup, sheer endless investigations, tests, and evaluations would be required. In case a crowdsourced ontology extension is planned, as mentioned before, more than one researcher could contribute to this. In doing so, also particular architectures could be prebuilt or used from existing reference architectures, such as the mentioned Lambda, Kappa, or Bolster architectures. Notably, an approach was created but not incorporated into the existing framework, facilitating decision support for the selection of big data reference architectures using the AHP as an MCDM [VBB+19a]. By amalgamate the particular consideration of big data technology combinations, the integration of further domain knowledge in the ontology, the extension of the inference

engine, and the aforementioned contribution, concrete architectures could be deployed, which do not need to be manually adjusted in terms of the interfaces. Hence, those cannot only be used for testing and rapid prototyping. Instead, these could potentially also be harnessed for more sophisticated implementations.

One cumbersome task, as highlighted, is denoted by the complexity of the ontology and the required effort of the extension. The comprehensive representability of complex relations comes at the expense of the integrability of the new data. Although a rudimentary editor is already integrated within the prototype to add new information about relevant technologies, this could be further extended in the future. Similar considerations apply to mechanisms for acquiring the information in general. Research can sometimes be very time-consuming and tedious. An automated or community-based approach seems to be desirable. In the case of the former, web-scraping tools such as Apache Nutch could be helpful in automatically searching and handling information received by web crawling. Regarding the latter, interested parties and users of the ontology could contribute additional content to all big data technologies but also the domain in general. This concerns, among other things, e.g., terms, descriptions, data characteristics, and architectures (cf. section 2.2). With the identification of the individual technologies by means of the inference engine, it was determined that with the recommendation also additional references to development or deployment tools could be given. In principle, the question arises whether a general extension to include non-big data technologies would be useful in the sense of future research, with the help of which even more comprehensive recommendations can be given. In this regard, however, further FRs and NFRs, as well as renewed assessments of already integrated technologies, are required.

In addition to other obvious enhancements to the prototype, like integration further MCDM approaches, even more fine-grained configuration options for the user, a complete roles and permissions concept, and visual adjustments are conceivable. One very important aspect most of the interviewees have addressed is the option to preselect technologies that need to be incorporated. Using this option during the multi-stepped MCDM approach, existing parts of the environments could be *simulated* and strongly required technologies always preselected. Another idea could be the creation of microservices for each of the developed components and their loose coupling. This would allow them to be integrated into workflows, larger approaches, and architectures. An implementation in *continuous integration and continuous development* (CI/CD) pipelines would also be sensible, whereby the individual components make automated decisions and convert these directly into execution directives. In the sense of test-driven developments, as they are described in detail in [SVN+19a; SHT19; SVL+21], furthermore the opportunity arises to use and extend the system in such a way that the individual tests are carried out automatically by means of the *deployment manager* and the created system is made available afterward.

The testing of all previously mentioned steps and further developments would then have to take place in a similar framework as it has already been carried out here in the contribution at hand. This applies especially for the Eval 4 step, which requires a meticulous and long-term application in an organizational context [SV12a], is something that could be investigated in the future. It is expected that in a real-world context, certain requirements regarding the prototypical development will arise once again, which have not been considered so far. Although all interviewees gave a positive picture of both the framework and the prototype, the actual application would presumably provide improvement and revision ideas. Consequently, an actual product could be developed and used from the prototype in the future.

A Appendix

A.1 Appendix A - Standard Use Case for Big Data Projects

No.	UC	Aim	Relevant Features (cf. Table A.2)	Changes
1	2, 3, 5, 6, 8, 14, 37, 39, 44, 45	Improve the analysis algorithms	high velocity; data used for analysis; real-time (near-real-time) processing; data fusion; unstructured data; permanent data; dynamic data; basic statistics; search/query/indexing; classification	Case 14 was added
2	1, 9, 13	Analyze the data from (IoT) sensors	data from different devices/sensors; visualization; data used for analysis; unstructured data; batch processing; data shared between users/applications; classification; clustering algorithms	Cases left from the initial third cluster
3	4, 19, 21, 23, 30, 35, 47	Realize smart city concepts	data fusion; visualization; unstructured data; real-time (near-real-time) processing; personally identifiable information; permanent data; data shared between users/applications; basic statistics	Merged out of the third and fourth cluster
4	15, 28, 29	Integrate heterogeneously structured data for multi-leveled problems	structured and unstructured data; real-time processing; data fusion; personally identifiable information; permanent data; transient data; data shared between different users/devices; NoSQL; dynamic data; deep learning; basic statistics; search/query/indexing; data classification	Former fifth cluster

Table A.1 continued from previous page

No.	UC	Aim	Relevant Features (cf. Table A.2)	Changes
5	7, 27, 42, 43,	Improve the analysis quality through additional data	data coming from different institutions; unstructured data; valuable data; batch-processing; data mining; personally identifiable information; HDFS; permanent repository; basic statistics; search/-query/indexing	First group of the former sixth cluster
6	10, 11, 26, 48	Link data from different sources	data cleaning; batch-processing; unstructured data; permanent data; dynamic data; HDFS; NoSQL; basic statistics; classification;	Second group of the former sixth cluster
7	18, 31, 38	Enable decision-making	real-time processing; data fusion; unstructured data; pre-processing; text; images; personally identifiable information; Permanent data; basic statistics; search/query/indexing; data mining; classification	First group of the former seventh cluster
8	12, 20, 24, 32, 36, 41, 46	Enable real-time analysis for data, incoming with high-speed	real-time processing; data fusion; visualization; structured and unstructured data; personally identifiable information; data shared between users; permanent data; invaluable data; search/-query/indexing; classification; basic statistics	Second group of the former seventh cluster
9	17, 22, 25	Optimize existing processes	high-speed data; both real-time and batch processing; structured and unstructured data; data mining (recommenders); visualization techniques; data fusion	Manually built cluster
Outliers		16, 33, 34, 40	-	

Table A.1: Derived clusters after qualitative analysis [VST+20]

No.	Feature	Number of Occurrences
1.	Unstructured Data	40
2.	Semi-structured Data	4
3.	Structured Data	16
4.	Heterogeneous Data	40
5.	Historical Data	11
6.	Dynamic Data	43
7.	Real-time Data	32
8.	Event Processing	4
9.	Stream Processing	6
10.	Batch Processing	19
11.	High Velocity	20
12.	Multiple Sources	39
13.	Data Cleaning	5
14.	Data Pre-Processing	13
15.	Data Fusion	41
16.	Parallel Processing	7
17.	Parallel File System	1
18.	Distributed Computing	7
19.	NoSQL	21
20.	IoT	15
21.	GIS	11
22.	Statistical Calculations	40
23.	Machine Learning	16
24.	Data Mining	16
25.	Structured and Unstructured Data	12
26.	High Performance Computing	4
27.	Big Data Analysis	34
28.	Map Reduce	20
29.	Hadoop	23
30.	Spark	8

Table A.2: A list of all features and their occurrences [VST+20]

U/F	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	●	○	○	●	●	●	○	○	○	●	●	●	○	●	●	○	○	○	○	●	○	●	○	●	○	○	●	○	○	○
2	●	○	○	●	●	●	○	○	●	○	●	●	○	●	●	○	○	○	●	○	○	●	○	●	○	○	●	●	○	●
3	●	○	○	●	○	●	●	○	○	○	●	●	○	●	●	○	○	○	○	●	○	●	●	●	○	○	●	●	○	●
4	●	○	○	●	●	●	●	○	○	○	●	●	○	○	●	○	○	○	●	●	○	○	○	○	○	○	○	○	○	○
5	●	○	○	●	●	●	●	○	●	○	●	●	○	○	●	●	○	○	○	●	○	●	●	○	○	○	●	○	●	○
6	●	○	○	●	●	●	●	○	●	○	●	●	●	●	●	○	○	○	○	○	○	●	●	○	○	○	●	○	●	○
7	●	○	○	●	○	●	○	●	○	●	○	●	○	○	●	●	○	○	○	○	○	●	●	●	○	○	●	●	●	○
8	●	○	○	●	●	●	●	○	○	●	●	●	○	○	●	○	○	○	○	●	○	●	●	●	○	○	●	●	●	●
9	●	○	○	●	○	●	○	○	○	●	○	●	○	○	●	○	○	○	●	●	○	●	●	●	○	○	●	○	○	○
10	●	○	○	○	○	●	○	○	○	●	○	○	●	●	○	○	○	○	●	○	○	●	●	●	○	○	●	●	●	○
11	●	○	●	●	○	●	○	○	○	●	○	●	●	●	●	○	○	○	●	○	○	●	○	○	○	○	○	○	●	○
12	●	○	●	●	○	●	●	○	○	○	○	●	○	○	●	○	○	○	●	○	○	●	○	●	○	○	●	○	●	○
13	●	○	○	●	○	●	●	○	○	●	●	●	○	○	●	○	○	○	○	●	●	●	○	○	○	○	○	●	○	○
14	●	○	○	●	○	●	●	○	○	○	○	●	○	○	●	○	○	○	●	○	○	○	○	○	○	○	○	○	○	○
15	●	○	●	●	●	●	●	○	●	●	○	●	○	○	●	○	○	●	●	○	○	●	●	○	○	○	●	○	●	○
16	○	○	●	○	○	●	●	○	○	○	○	○	○	○	●	○	○	○	●	○	○	●	○	○	○	○	○	○	○	○
17	●	○	●	●	○	●	●	○	○	●	●	●	○	●	●	○	○	○	●	●	○	●	●	○	●	○	●	○	○	○
18	●	○	●	●	○	●	●	○	○	●	○	●	○	●	●	○	○	●	○	○	○	●	○	●	●	○	●	○	○	○
19	●	○	○	●	○	●	●	○	○	○	○	●	○	○	●	○	○	○	○	●	●	●	○	○	○	●	●	●	●	●
20	●	●	●	●	○	●	●	○	○	○	●	●	○	○	●	●	○	○	○	○	●	●	○	○	●	○	●	●	●	●
21	●	●	○	●	○	●	●	○	○	○	○	●	○	○	●	○	○	○	○	●	●	●	○	○	○	○	●	○	○	○
22	○	○	○	○	●	●	●	○	○	●	●	●	○	●	●	●	○	○	○	○	●	●	○	○	○	●	●	○	○	○
23	○	○	○	●	○	●	●	○	○	○	●	●	○	○	●	○	○	○	●	●	●	●	○	○	○	○	○	○	○	○
24	●	○	●	●	○	●	●	○	○	○	●	○	○	○	●	○	○	○	●	○	○	●	●	●	●	○	●	●	●	●
25	●	○	●	●	○	●	●	●	○	●	○	●	○	○	●	○	○	○	○	●	●	●	○	●	●	○	●	○	○	○
26	●	○	○	●	○	●	●	○	○	●	○	●	○	○	●	○	○	●	○	○	○	●	○	○	○	○	●	●	●	○
27	●	○	●	●	○	●	○	○	○	●	●	●	○	○	●	○	○	○	○	○	○	●	○	○	●	○	○	●	●	○
28	●	●	●	●	○	●	○	○	○	○	○	●	○	○	●	○	○	○	●	○	○	●	●	○	●	○	●	○	●	○

Table A.3 continued from previous page

U/F	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
29	●	●	●	●	○	●	●	○	○	○	●	●	○	○	●	○	○	○	●	●	○	●	○	○	●	○	●	○	○	○
30	●	○	○	●	○	●	●	●	○	●	○	●	○	○	●	○	○	○	●	●	●	●	○	○	○	○	●	●	○	○
31	●	○	○	●	○	●	●	○	○	○	●	●	○	●	●	●	●	○	○	○	○	●	○	●	○	○	●	○	●	○
32	●	○	●	●	○	●	●	○	●	●	○	●	○	○	●	○	○	○	●	○	●	●	●	●	●	○	●	●	●	●
33	●	○	○	●	○	●	○	●	○	○	○	●	○	○	●	○	○	○	●	○	○	●	○	○	○	○	○	○	○	○
34	●	○	○	●	○	●	●	○	○	○	●	●	○	○	●	○	○	●	●	○	○	●	●	○	○	○	○	●	○	●
35	●	○	○	●	○	●	●	○	○	○	○	●	○	○	●	○	○	○	●	●	●	●	○	○	○	○	●	○	○	○
36	●	○	○	●	○	●	●	○	○	○	○	●	○	○	●	○	○	●	●	○	●	●	●	●	○	○	●	●	●	○
37	●	○	○	●	●	●	●	○	○	○	●	●	●	●	●	○	○	○	○	○	○	●	●	○	○	●	●	●	○	○
38	●	○	●	●	○	●	●	○	○	○	○	●	○	●	●	○	○	○	○	○	○	●	●	●	●	○	●	●	●	○
39	●	○	○	●	○	●	●	○	○	○	●	●	●	●	●	○	○	○	○	○	○	●	○	○	○	○	●	●	●	○
40	●	○	●	●	●	●	●	○	○	○	○	●	○	○	●	●	○	●	○	○	○	●	○	●	●	○	●	●	●	○
41	●	○	●	●	●	●	●	○	●	●	●	●	○	○	●	○	○	●	●	○	○	●	○	○	●	○	●	●	●	○
42	●	○	○	●	○	●	○	○	○	●	○	○	○	○	○	●	○	○	○	○	○	●	○	○	○	●	●	○	●	○
43	●	○	○	●	○	●	○	○	○	●	○	●	○	○	●	○	○	○	○	○	○	○	○	○	○	○	●	○	●	○

Table A.3: Input matrix showing the occurrences of each feature (F) in the respective use cases (U) [VST+20]

A.2 Appendix B - Big Data-related Requirements

NFR	**Availability (AV)**	This non-functional requirement refers to the overall availability of the tool. The higher the value is, the more robust the system is against outtages of single nodes.
	Computational Complexity (CC)	This non-functional requirement concerns the subjectively perceived complexity of the calculation that can be carried out with the prospective tool. Along with this, it also addresses the necessary resources for the fulfillment of this calculation. As a result, the higher the value, the better the resource utilization, even when complex computations are performed.
	Cost (C)	This non-functional requirement covers the monetary expenditures that are required to make use of all of the available functionalities. The higher the value is, the less money it cost.
	Documentation and Support (DO)	This non-functional requirement focuses on the overall chance to receive support for the respective solution. This is, inter alia, represented by official documentations, product descriptions, community platforms, support hotlines, and mails. The higher the value is the more of the previously mentioned support opportunities are existing.
	Flexibility and Scalability (FS)	This non-functional requirement focuses on the overall chance to add additional components to the existing environment. This is not only reflected by additional tools but also to the scaling of the solution itself. The higher the value, the easier a potential extension can be realized by additional resource and/or tools.
	Installation and Maintenance (IM)	This non-functional requirement refers to the overall effort that is required for the installation and maintenance of the chosen solution. At its best, an easy implementation and connection to other tools is facilitated and automatic updates and an admin panel given.
	Regulation (RE)	This non-functional requirement covers all of the aspects, which might be relevant for legal constraints and, thus, to the overall degree of their fulfillment, application, extension and modification. In context of this licensing plays an important role. The opener the tool and is application the higher the value. A very low rating indicates a proprietary software which offers no leeway in the previously referred context.
	Fault Tolerance (FT)	This non-functional requirement refers to the overall capability to handle interruptions, failures or other kind of errors that may occur and have a negative influence on the overall operability of the tool/system or provided services. A high fault tolerance is given when the system can work as intended and no long-term data loss occurs, if a problem will occur.
	Reliability (R)	This non-functional requirements covers all aspects that are related to the overall reliability of the system and therefore all tasks can be done in a certain amount of time under the given conditions.
	Security (SY)	This non-functional requirement focuses on all of the related aspects that can be attributed to the security. This includes, inter alia, the ability to manage the access, to keep track of performed activities in a historical manner (log), prevent any kind of disclosure and protect the communication as good as possible.
	Storage Capacity (SC)	The property reflects how large the storage requirement are that results from both the installation and its effective use on the intended system.
	Sustainability (S)	This non-functional requirement reflects whether and to what extent a development of the solution is continued or can be expected. The lowest value is assigned if the targeted solution has already been discontinued.
	User Interface (UI)	This non-functional requirement refers to the overall availability of a graphical user interface and its offered functionalities. The severity of this non-functional requirement is very low if no GUI is available or can be added without greater effort. An example in that case would be the communication via command-line.

	Automation Acting (AA)	The functionality refers to the general capability to automatize different tasks, e.g., within a process, without the necessity of human interaction.
	Batch Processing (BP)	This functionality addresses the ability to process sets of data (batches) in a recurring sequence without the need for constant human interaction. This processing can be triggered by fulfilling certain conditions, e.g., time-related conditions.
	Cluster Management (CM)	This functionality refers to the general functionalities of managing computer clusters.
	Consistency Preservation (CP)	This functionality refers to general functionalities that are used to preserve the consistency of the data. In doing so, an attempt is made that all data is in a consistent status across all related nodes, stores, or tools.
	Data Aggregation (AG)	This functionality refers to functionalities that facilitate the aggregation and, thus, the combination of different data. The data itself may also originate from various data sources.
	Data Classification (CF)	This functionality refers to the general capability to investigate the differently structured data and categorize them based on different properties or meta information.
	Data Cleaning (CL)	This covers functionalities of improving data quality to an appropriate level, applicable to the chosen analysis techniques. This can be the selection of subsets of the data, the insertion of proper default values, or more sophisticated techniques such as estimating missing data through modeling.
	Data Clustering (CU)	This functionality refers to the general capability of applying clustering algorithms to the available data. Those are typically aligned to the domain of machine learning and used to identify
	Data Formatting (F)	This functionality refers to primarily syntactic modifications made to the data that do not change its meaning but might be required by the modeling tool.
	Data Mining Algorithm Support (DM)	The functionality refers to the overall capability to enable and apply known data mining algorithms.
	Data Pipelining (P)	This functionality refers to general functionalities that allow a simple or event based automatized data transfer between tools.
	Data Selection (DS)	This refers to general functionalities used to support the data selection process, including, for instance, particular data types or data limits.
	Data Streaming (ST)	This refers to the functionality of handling continuous data streams. It focuses on the overall data handling rather than the particular processing, denoted by the stream processing.
	Data Visualization (V)	This functionality targets an information processing objective achieved by a data visualization algorithm execution process. A planned process that creates images, diagrams, or animations from the input data.
FR	**Event-Data Processing (EP)**	This refers to the general functionalities that allow the handling of complex events. Apart from the sole detection of events, the handling, filtering, and further data-related techniques are also considered.
	Machine Learning (ML)	This functionality refers to the general capability to perform any kind of machine learning techniques on the data at hand. This includes, e.g., algorithms in reinforcement, supervised and unsupervised learning.
	Message Handling (MH)	This functionality describes the overall capability between various systems or particular tools that can handle messages, including sensing and receiving.
	Monitoring (MO)	Refers to monitoring activities that allow the inspection of the internal processes and thus their current status. In doing so, vital information can be recorded and displayed over time.
	Near Real Time Processing (NP)	This functionality refers to the overall capability to acquire and process the targeted in near-real time.
	Parallel Processing (PP)	This functionality refers to the overall capability to parallel process larger task, using specific methods or the ability to optimize the use of the technical foundation of the system environment.
	Real Time Processing (RT)	This functionality refers to the overall capability to acquire and process the targeted in real time.
	Recovery Mechanics (RC)	This functionality covers all of the existing methods and techniques relevant to the recovery of the system, its components, or attached elements. In doing so, services and data are also included here.
	Reporting (RP)	This functionality refers to the overall capability to create reports from the conducted analysis or rather the achieved results.
	Resource Management (RM)	This refers to all functionalities used to manage the given computational resources.
	Store Semi-Structured Data (SS)	This functionality refers to the general capability to store unstructured data, such as audio or video files. It typically goes hand in hand with NoSQL database solutions.
	Store Structured Data (SD)	This functionality refers to the general capability to store structured data, revealing inner structures and coming in known formats. It typically goes hand in hand with relational databases or other kinds of solutions, at which data in standardized formats can be stored.
	Store Unstructured Data (SU)	This functionality refers to the general capability to store unstructured data, such as audio or video files. It typically goes hand in hand with NoSQL database solutions.
	Streaming Processing (ST)	This functionality refers to the ability to acquire and process the targeted in streaming mode.
	Support Scripting Languages (SL)	This functionality refers to the general capability to support scripting languages to either add, extend or modify existing functionalities. Inter alia, this may also include implicitly other functionalities like machine learning, the support of data mining algorithms, and others.
	System Deployment (SD)	This functionality refers to the general capability to distribute and install software packages on local or remote IT resources.

Table A.4: Description of all FRs and NFRs

NFR	One	Three	Five
AV	The loss of either the tool or even one node will result in a complete system crash. Single point of failure.	Updates can be performed during runtime. Minor response and connection errors may occur.	A high-availability can be achieved. Whenever options, modifications or other changes are required, this can be done without any downtime.
CC	The tool has a low scope of functions. Even smaller tasks require a high computational effort. Along with this, resource utilization is bad.	The tool is capable of fulfilling complex computations. However, no sophisticated approaches for resource utilization are used, and the tool is very demanding.	Complex computations are possible. The resource utilization is still very good.
C	Initial cost together with ongoing subscription/service cost.	Reasonable initial or subscription cost.	No cost at all.
DO	No documentation and support at all	A documentation exists on the project website. In there, the most relevant information is included.	Easily accessible support and documentation. Very comprehensive and delivers everything that needs to be known. Apart from that, a mail list, communities and/or even a hotline exist.
FS	No scaling and extension at all.	The current solution can be easily extended and scaled. However, problems may occur, which result, at least, in an eventually consistent state. Apart from that, additional steps might be required to get everything fully working after the extension.	The existing solution can be easily extended, and further tools and connections added, during runtime, primarily through the sophisticated scaling possibilities. No downtime or other larger issues can be expected.
IM	Everything needs to be done manually. This includes the installation, updates, and everything else.	The installation and updates have to be done in a mixed way. Some of the activities need to be done manually, others are automated. However, scripts are partially (and officially) exist that help at some point.	The installation is done via a specific assistant and is mostly automatized. All updates can be easily automated, and all interconnection to other tools are done without any manual configuration effort.
RE	Proprietary software that does not allow any kind of extension and modification. The use is restricted and bounded so specific conditions.	The tool is free to use, but no extensions and modifications are allowed.	The tool is free to use, the further development in terms of extensions and modifications is supported.
FT	The tool offers no functionalities at all that are related to fault tolerance. As soon as a problem or outtake may occur, the overall operability of the underlying system is interrupted. Human interaction is strongly required.	The tool offers some functionalities for fault tolerance.	The tool is robust again any unforeseen outtakes, either in the core or additional nodes/instance. The service or rather its functionalities, can be provided without greater interruptions.
R	The tool works really unstable. It cannot be guaranteed that constant operability can be assured. A general operation is mostly bonded on many different conditions.	The tool should work most of the time without any problems. All of the functions are comprehensible, and the different steps for the user are visible (e.g., logs for debugging). Known issues are discussed, and potential help is provided.	The tool is constantly maintained and working fine (claimed). There are no known problems or any kind of issues.
SY	No security mechanics at all.	Some mechanics, such as Logging, user management, transaction encryption.	Puts massive effort into fulfilling as many security aspects as possible.
SC	The capacity for the tool is very high (could be biased and must be seen in comparison to the others), and some storage needs to be reserved before.	Only the tool is required, and options for outsourcing are given. E.g., endpoints for the data to be processed as well as the results.	The required storage for the tool is shallow. Additional configurations can be set up to attach further storage solutions, e.g., for the raw data and processed results.
S	The tool is deprecated. No further effort is put into development and maintenance.	The tool is still maintained, and updates are most probably coming.	The tool is currently maintained and further extended. There is constant support and also guaranteed for the next three years.
UI	No graphical user interface at all.	A GUI is supported by additional tools & libraries.	Perfectly working GUI that can be locally used, via web etc., and also offers many details and a broad number of options.

Table A.5: Used criteria for all NFRs to identify the severity for each technology

T/FR	AA	AP	BM	CP	DA	DC	CL	CU	F	DM	P	DS	ST	V	EP	EL	MH	MO	NP	PP	RT	RC	RP	RM	SS	SD	SU	SP	SL
Airflow	○	○	○	○	○	○	○	○	○	○	●	○	○	●	○	●	●	●	○	○	○	○	○	○	○	○	○	○	●
Ambari	●	●	●	●	○	○	○	○	○	○	○	○	○	○	○	○	○	●	○	○	○	○	●	●	○	○	○	○	●
Athena	○	○	○	●	○	○	○	○	○	○	●	●	○	●	○	○	○	○	○	○	○	○	●	○	●	●	●	○	●
Avro	○	○	○	●	○	○	●	○	●	○	○	○	○	○	○	○	●	○	○	○	○	○	○	○	○	●	○	○	●
BigML	○	○	○	○	○	●	○	○	○	○	○	○	○	●	○	○	○	●	○	○	○	○	○	●	○	○	○	○	●
Bokeh	○	○	○	○	○	○	○	○	○	○	○	○	●	●	○	○	○	○	○	○	○	○	○	○	○	○	○	●	●
Cassandra	○	○	○	●	○	○	○	○	○	○	○	●	●	○	○	○	●	●	○	●	○	●	●	○	●	●	●	●	○
Chukwa	○	○	●	○	○	○	○	○	○	○	●	●	○	●	○	○	○	●	○	○	○	○	○	○	○	○	○	○	○
CouchBase	○	○	○	●	●	○	○	○	○	○	●	●	○	○	○	○	○	○	○	○	○	●	○	○	●	●	●	○	●
CouchDB	○	○	○	●	●	●	●	●	●	●	●	●	○	○	○	○	○	○	○	○	○	●	○	○	●	●	●	○	●
DataCleaner	○	○	○	○	●	●	●	●	●	○	○	●	○	●	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●
Drill	○	○	○	●	○	○	○	○	○	○	○	●	○	○	○	○	○	○	○	○	●	○	○	○	●	●	●	○	●
Elasticsearch	○	○	○	○	●	●	●	●	●	○	○	●	○	●	○	●	○	●	●	○	●	●	●	○	○	○	○	●	●
Flink	●	●	○	○	●	●	●	●	○	○	●	●	○	○	●	●	○	○	●	●	●	○	○	●	○	○	○	●	●
Giraph	○	○	○	○	●	○	○	●	○	○	○	○	○	○	○	○	○	○	●	○	●	●	○	○	○	○	○	○	○
GraphLab	○	○	○	●	○	●	○	●	○	●	○	●	○	○	○	●	○	○	○	●	○	○	○	○	●	●	●	○	●
GraphX	○	○	○	○	●	●	●	●	●	○	●	○	○	●	○	○	○	○	○	○	●	○	○	○	○	○	○	○	●
GridGain	○	○	●	●	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●	○	●	●	●	●	●	○	○	○	●
Hadoop	○	●	●	○	○	○	○	○	○	○	○	○	●	○	○	○	○	○	●	○	●	○	○	○	●	○	○	○	○
Hama	○	○	○	○	●	●	●	●	●	○	○	○	○	○	○	●	●	●	○	●	●	○	○	○	○	○	○	○	●
HANA	○	○	○	○	●	●	●	●	●	●	○	●	●	○	○	○	○	●	○	●	●	●	○	○	○	○	○	●	●
HBase	●	○	●	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●	○	○	○	●	●	●	○	●
Heron	○	○	●	○	○	○	○	○	○	○	○	○	●	●	○	○	●	●	○	●	○	○	○	●	○	○	○	●	●
Hive	○	●	○	○	●	●	●	●	●	●	●	○	○	○	○	○	○	○	○	○	●	○	○	○	●	●	○	○	○
HPCC	○	●	○	○	●	●	●	●	●	●	●	●	○	○	○	○	○	○	○	●	○	○	○	●	○	○	○	○	●
Hstore	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●	●	●	○	●
InfoSphere	○	○	●	●	○	○	●	○	○	●	●	○	●	●	○	○	○	●	○	●	●	○	●	●	●	●	●	●	●
Kafka	○	●	○	●	○	○	○	○	○	○	●	○	○	○	●	○	●	○	○	●	○	○	○	○	○	○	○	●	○
Kinesis	●	○	○	○	○	○	○	○	○	○	●	○	●	○	○	●	●	●	●	○	●	○	○	●	●	●	●	●	●
Lumify	○	○	○	○	●	●	●	●	●	○	○	○	○	●	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○

Table A.6 continued from previous page

T/FR	AA	BP	CM	CP	DA	DC	CL	CU	F	DM	P	DS	ST	V	EP	ML	MH	MO	NP	PP	RT	RC	RP	RM	SS	SD	SU	SP	SL
Mahout	●	○	○	○	○	●	●	●	●	●	○	●	○	○	○	●	○	○	○	○	○	○	○	○	○	○	○	○	○
MLib	●	○	○	○	○	●	●	●	●	●	●	●	○	○	○	●	○	○	○	○	○	○	○	○	○	○	○	○	●
MongoDB	●	○	●	○	●	○	○	○	○	○	○	○	○	●	○	○	○	○	○	○	○	○	○	○	●	●	○	○	●
Neo4j	●	○	○	○	●	●	●	●	●	●	●	●	○	○	○	●	○	●	○	○	○	○	○	●	○	●	○	○	●
Nifi	●	○	●	○	●	●	●	●	●	○	●	●	●	○	○	○	○	●	○	○	○	○	○	○	●	●	●	●	●
OpenRefine	○	○	○	○	○	○	●	○	●	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●	●	●	○	○
OrientDB	●	○	○	●	○	○	○	○	○	○	●	○	○	○	○	○	○	○	●	○	○	●	○	○	○	○	○	○	●
Pentaho	●	○	○	○	●	●	●	●	●	●	●	●	○	●	○	○	○	○	○	○	○	○	●	○	●	●	●	○	○
Pig	○	●	○	○	●	●	●	●	●	●	●	●	●	○	○	○	○	○	○	●	○	○	○	○	●	●	●	○	●
Pregel	○	○	●	○	●	○	○	○	○	○	○	○	○	○	○	○	●	●	○	○	○	○	○	●	○	○	○	○	●
Presto	○	○	○	○	●	●	●	●	●	●	●	○	○	○	○	○	○	○	○	○	●	○	●	●	○	○	○	○	○
Qubole	●	○	○	○	○	○	○	○	○	●	●	○	●	○	○	●	○	○	○	○	●	○	○	○	○	○	○	●	○
R	○	○	○	○	●	○	○	○	○	○	○	●	●	●	○	●	○	○	○	○	○	○	○	○	○	○	○	○	●
Rapidminer	○	○	○	○	○	●	●	○	●	●	○	●	○	●	○	●	○	○	○	○	○	○	○	○	○	●	○	○	●
RavenDB	●	○	●	●	○	○	○	○	○	○	○	○	○	○	○	○	○	●	○	○	○	○	○	●	○	○	○	○	○
Redis	○	○	○	●	○	○	○	○	○	○	○	●	●	○	○	●	●	○	○	○	●	○	○	○	○	●	○	○	●
Redshift	●	○	○	○	○	○	○	○	●	○	●	○	○	●	○	●	○	○	●	●	●	○	○	○	●	●	●	○	●
SAMOA	○	○	○	○	○	●	○	●	○	●	○	○	●	○	○	●	○	○	○	○	○	○	○	○	○	○	○	●	●
Scylla	○	○	○	○	○	○	○	●	○	○	○	○	●	○	○	○	○	○	○	○	●	●	○	○	○	○	○	●	●
Silk	○	○	○	○	○	○	●	○	●	○	○	○	○	○	○	○	○	○	○	●	○	○	○	○	●	●	○	○	●
Spark	○	●	○	○	○	○	○	○	○	●	●	○	●	○	●	●	○	●	○	●	○	○	○	●	○	○	○	●	●
Storm	●	○	○	○	●	●	●	●	●	●	●	●	●	○	○	●	●	●	○	○	●	●	○	○	○	○	○	●	○
Tableau	●	○	○	○	●	○	●	○	○	○	○	○	○	●	○	●	○	●	○	○	○	○	○	●	○	○	○	○	○
TensorFlow	○	○	○	○	●	●	●	●	●	○	●	○	○	○	○	●	○	○	○	○	○	○	○	○	○	○	○	●	○
Tez	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	●	○	○	○	●	○	○	○	○	○
VoltDB	○	○	○	○	○	○	○	○	○	○	○	○	●	○	○	●	○	○	○	○	●	○	○	○	○	○	○	●	○
ZooKeeper	●	○	●	●	○	○	○	○	○	○	○	○	○	○	○	○	○	●	○	○	○	●	●	●	○	○	○	○	○

Table A.6: Mapping of the technologies (T) and fulfilled FRs

T/NFR	AV	CC	C	DO	FS	FT	IM	RE	R	SY	SC	S	UI
Ambari	3	3	5	2	2	3	2	5	2	5	3	1	4
Athena	4	4	3	5	5	4	4	1	5	4	4	4	4
Avro	2	3	5	3	2	1	4	5	3	3	2	3	1
BigML	3	5	3	5	5	5	5	1	4	4	5	4	5
Bokeh	2	3	5	4	4	1	5	5	5	3	4	3	1
Cassandra	3	5	5	5	5	5	4	5	5	5	3	5	1
Chukwa	2	3	5	3	4	1	2	5	2	2	2	1	4
CouchBase	5	5	1	5	4	5	5	2	5	5	5	5	4
CouchDB	4	4	5	1	4	4	3	5	5	3	4	4	4
DataCleaner	2	3	2	4	4	3	5	5	5	2	3	3	5
Drill	4	5	5	4	5	3	4	5	3	1	2	4	1
Elasticsearch	5	3	3	5	5	5	4	1	5	5	3	4	1
Flink	4	4	5	5	5	5	5	5	4	3	4	4	4
Giraph	5	4	5	3	3	1	3	3	3	3	1	3	1
GraphLab	5	4	5	3	4	1	5	3	3	1	1	1	3
GraphX	5	3	5	5	1	1	5	3	3	1	1	5	3
GridGain	5	5	5	4	5	5	4	5	4	5	3	4	4
Hadoop	5	3	3	4	5	5	2	4	3	4	5	3	4
Hama	1	4	4	5	4	1	3	4	3	2	4	2	3
HANA	5	3	2	5	5	5	3	2	3	5	2	5	5
HBase	4	1	5	3	4	4	3	4	4	4	4	5	4
Heron	1	5	5	5	5	3	4	4	5	1	3	5	5
Hive	3	4	5	4	2	3	3	5	4	3	3	5	5
HPCC	5	3	4	5	5	5	3	3	4	4	2	5	1
Hstore	1	1	1	1	1	1	1	1	1	1	1	1	1
InfoSphere	5	5	2	5	5	5	4	5	5	5	2	5	4
Kinesis	3	5	2	5	5	3	3	3	5	3	5	5	3
Lumify	1	2	5	1	2	1	3	5	1	2	1	1	4
Mahout	1	3	5	5	4	2	3	5	2	3	3	3	1
MLib	5	5	5	5	2	2	4	5	3	3	5	5	2

Table A.7 continued from previous page

T/NFR	AV	CC	C	DO	FS	FT	IM	RE	R	SY	SC	S	UI
MongoDB	5	5	3	5	5	3	3	3	5	4	2	5	5
Neo4j	5	2	4	5	4	3	5	5	4	2	3	5	5
Nifi	1	5	5	5	5	1	5	5	4	5	5	5	5
OpenRefine	1	5	5	5	3	4	4	5	3	1	4	5	2
OrientDB	5	4	5	5	4	4	5	4	4	5	3	5	5
Pentaho	5	3	5	5	5	5	5	5	5	3	3	5	3
Pig	4	3	5	5	3	5	4	5	5	2	4	3	2
Pregel	1	3	4	1	5	5	1	2	3	2	5	3	1
Presto	1	3	5	5	3	1	2	5	2	4	4	5	5
Qubole	5	5	5	5	3	3	4	3	4	4	5	5	5
R	5	3	2	4	3	4	3	3	3	5	3	5	2
Rapidminer	5	3	5	5	5	3	4	5	5	5	5	5	3
RavenDB	5	3	4	5	5	5	5	1	5	5	5	4	5
Redis	5	5	5	5	5	5	3	5	5	5	4	5	3
Redshift	5	5	2	5	5	5	4	2	4	5	3	5	5
SAMOA	1	1	5	4	4	1	3	5	1	1	5	4	1
Scylla	5	5	5	5	5	5	5	2	5	5	1	5	5
Silk	1	3	5	2	3	2	1	4	2	1	2	1	1
Spark	5	3	3	4	5	3	4	5	3	5	3	5	4
Storm	5	5	5	3	4	4	4	5	3	5	2	4	5
Tableau	5	4	2	5	5	5	3	2	3	4	4	5	5
TensorFlow	1	3	5	5	5	3	3	3	4	4	3	3	3
Tez	1	4	5	5	2	2	3	5	4	2	5	5	5
VoltDB	5	3	4	5	5	3	2	5	4	1	4	5	5
ZooKeeper	5	4	5	4	5	4	3	5	4	4	3	2	5
Airflow	3	3	5	5	5	2	4	4	3	5	4	5	5
Kafka	5	4	5	3	4	4	2	4	4	5	3	5	1

Table A.7: Mapping of the technologies (T) and fulfilled NFRs

A.3 Appendix C - Exemplary Modeled Deployment Diagrams

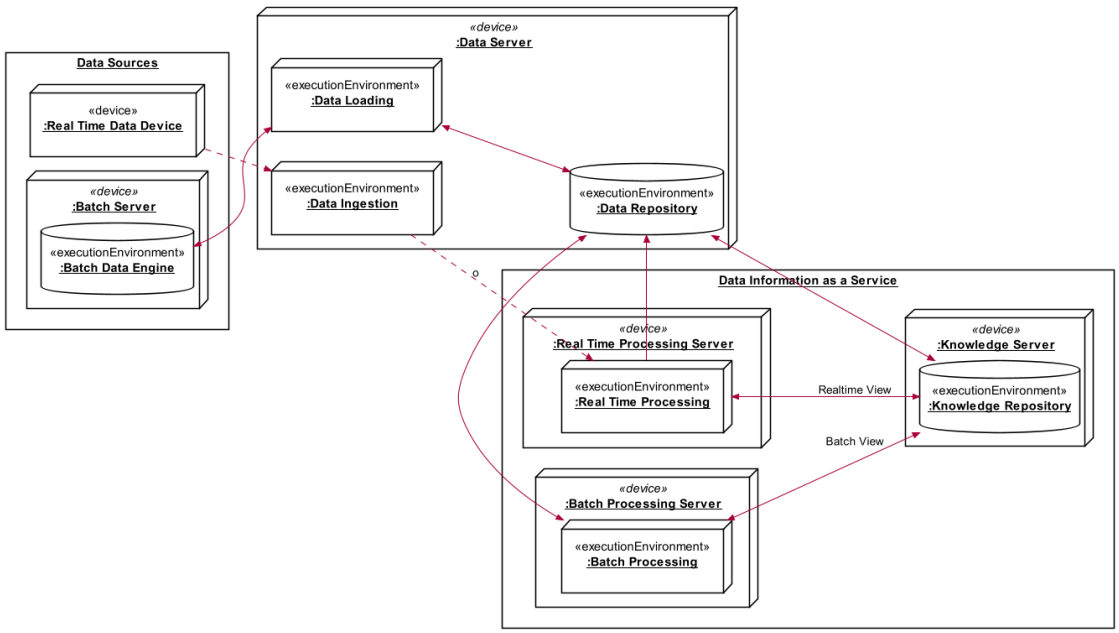


Figure A.1: Automatically created deployment diagram for the evaluation based on the description of [LPB16]

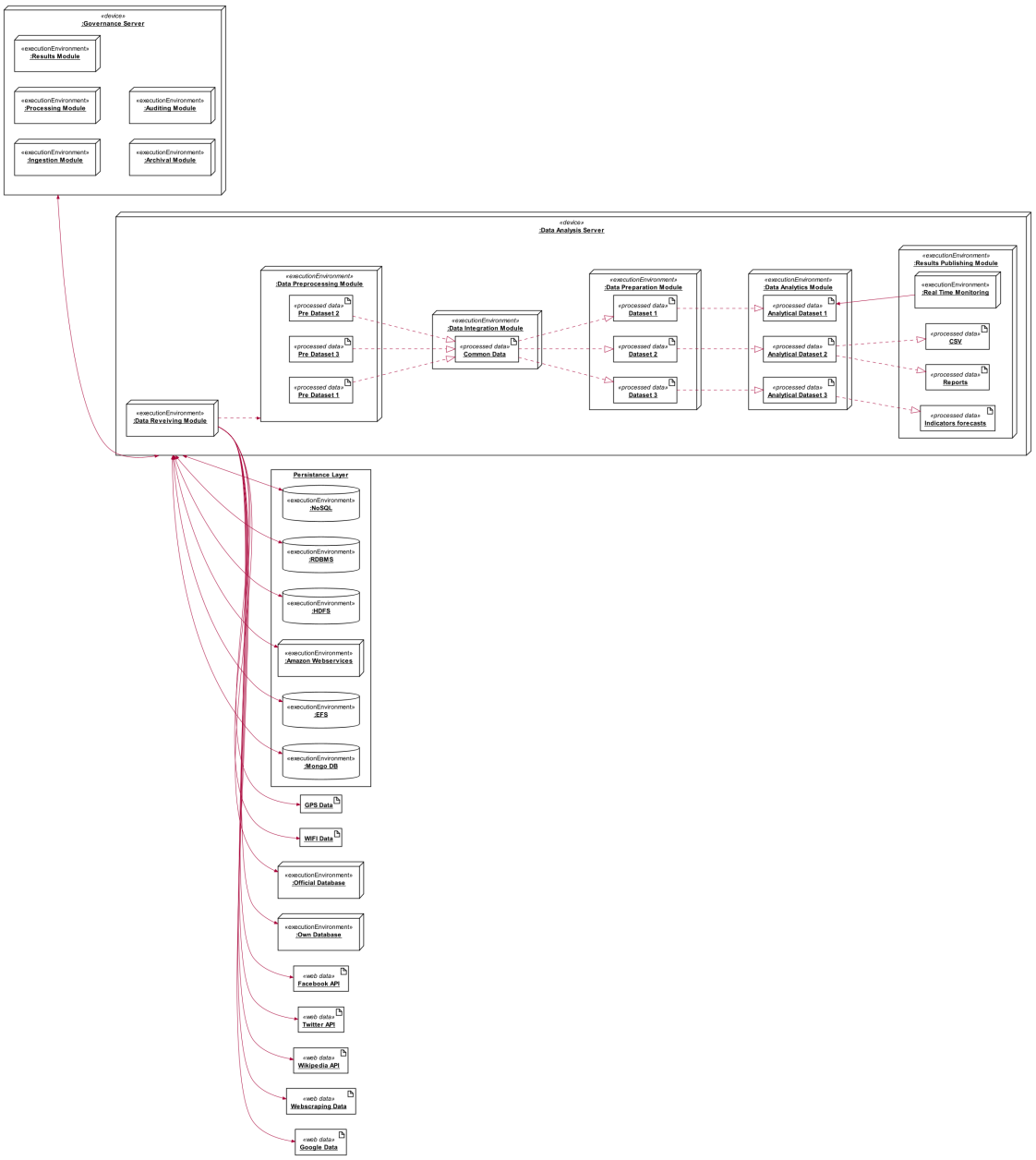


Figure A.2: Automatically created deployment diagram for the evaluation based on the description of [BD18]

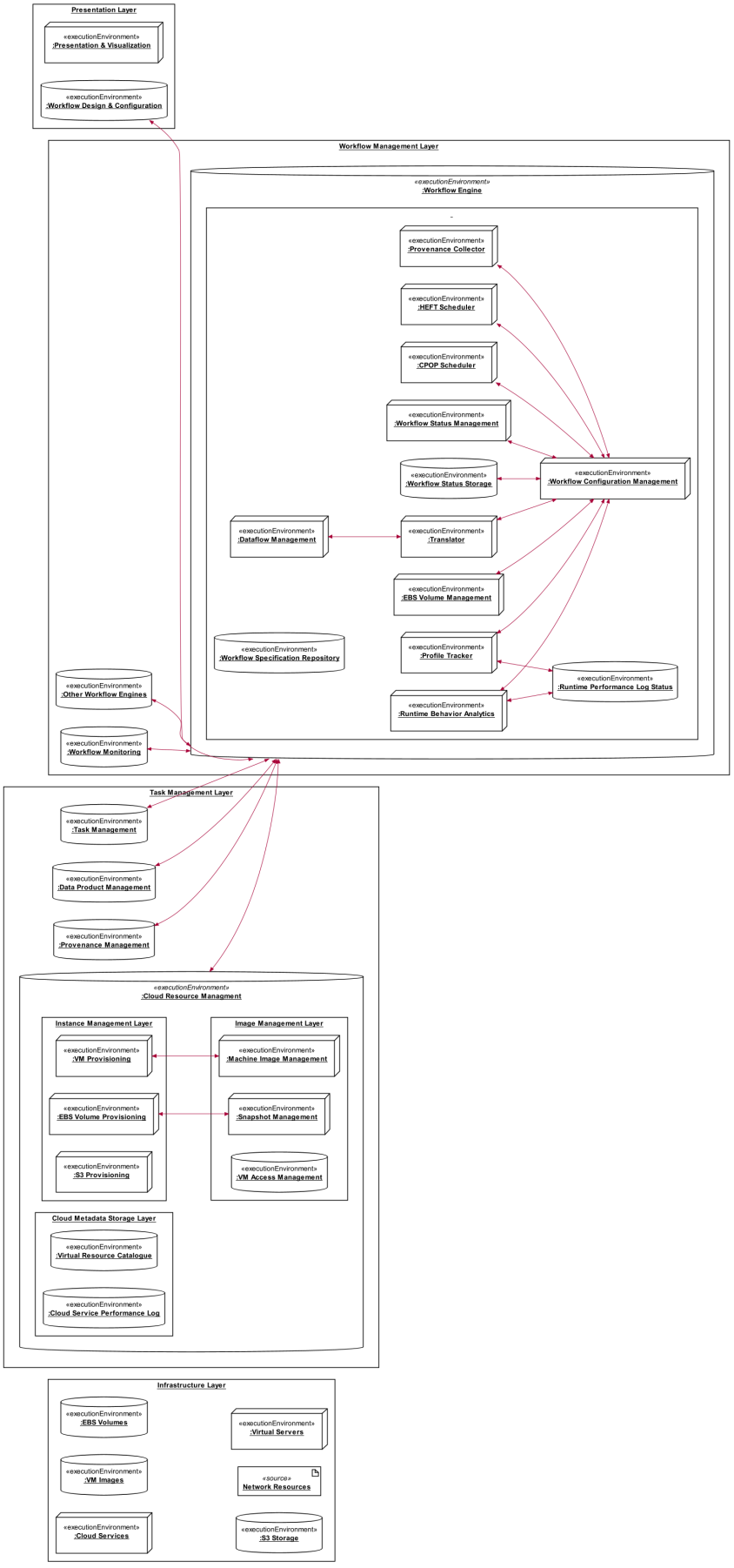


Figure A.3: Automatically created deployment diagram for the evaluation based on the description of [KL14]

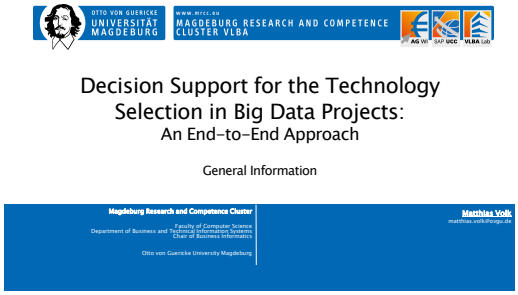
A.4 Appendix D - Evaluation

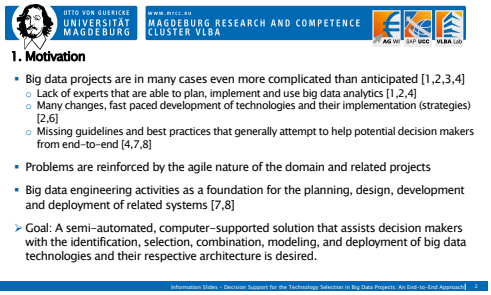
[ZZW+18]	UI	IM	FS	C	CC	RE	SC	SY	DO	FT	AV	S	R
UI	1.0	1.0	1.0	1.0	1.0	6.0	1.0	6	3.0	1.0	1.0	6	1.0
IM	1.0	1.0	1.0	1.0	1.0	6.0	1.0	6	3.0	1.0	1.0	6	1.0
FS	1.0	1.0	1.0	1.0	1.0	6.0	1.0	6	3.0	1.0	1.0	6.0	1.0
C	1.0	1.0	1.0	1.0	1.0	6.0	1.0	6	3.0	1.0	1.0	6.0	1.0
CC	1.0	1.0	1.0	1.0	1.0	6.0	1.0	6	3.0	1.0	1.0	6.0	1.0
RE	1/6	1/6	1/6	1/6	1/6	1.0	1/6	1.0	1/3	1/6	1/6	1.0	1/6
SC	1.0	1.0	1.0	1.0	1.0	6.0	1.0	6.0	3.0	1.0	1.0	6.0	1.0
SY	1/6	1/6	1/6	1/6	1/6	1.0	1/6	1.0	1/3	1/6	1/6	1.0	1/6
DO	1/3	1/3	1/3	1/3	1/3	3.0	1/3	3.0	1.0	1/3	1/3	3.0	1/3
FT	1.0	1.0	1.0	1.0	1.0	6.0	1.0	6.0	3.0	1.0	1.0	6.0	1.0
AV	1.0	1.0	1.0	1.0	1.0	6.0	1.0	6.0	3.0	1.0	1.0	6.0	1.0
S	1/6	1/6	1/6	1/6	1/6	1.0	1/6	1.0	1/3	1/6	1/6	1.0	1/6
R	1.0	1.0	1.0	1.0	1.0	6.0	1.0	6.0	3.0	1.0	1.0	6.0	1.0

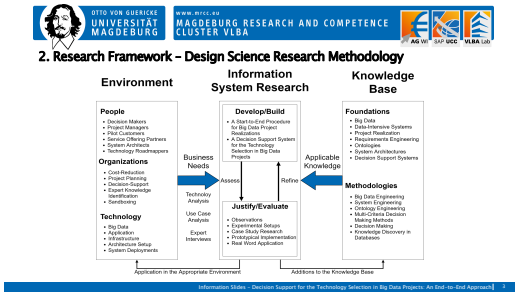
[YMR+17]	UI	IM	FS	C	CC	RE	SC	SY	DO	FT	AV	S	R
UI	1.0	1.0	1.0	8.0	8.0	3.0	1.0	3.0	3.0	1.0	1.0	8.0	1.0
IM	1.0	1.0	1.0	8.0	8.0	3.0	1.0	3.0	3.0	1.0	1.0	8.0	1.0
FS	1.0	1.0	1.0	8.0	8.0	3.0	1.0	3.0	3.0	1.0	1.0	8.0	1.0
C	1/8	1/8	1/8	1.0	1.0	1/6	1/8	1/6	1/6	1/8	1/8	1.0	1/8
CC	1/8	1/8	1/8	1.0	1.0	1/6	1/8	1/6	1/6	1/8	1/8	1.0	1/8
RE	1/3	1/3	1/3	6.0	6.0	1.0	1/3	1.0	1.0	1/3	1/3	6.0	1/3
SC	1.0	1.0	1.0	8.0	8.0	3.0	1.0	3.0	3.0	1.0	1.0	8.0	1.0
SY	1/3	1/3	1/3	6.0	6.0	1.0	1/3	1.0	1.0	1/3	1/3	6.0	1/3
DO	1/3	1/3	1/3	6.0	6.0	1.0	1/3	1.0	1.0	1/3	1/3	6.0	1/3
FT	1.0	1.0	1.0	8.0	8.0	3.0	1.0	3.0	3.0	1.0	1.0	8.0	1.0
AV	1.0	1.0	1.0	8.0	8.0	3.0	1.0	3.0	3.0	1.0	1.0	8.0	1.0
S	1/8	1/8	1/8	1.0	1.0	1/6	1/8	1/6	1/6	1/8	1/8	1.0	1/8
R	1.0	1.0	1.0	8.0	8.0	3.0	1.0	3.0	3.0	1.0	1.0	8.0	1.0

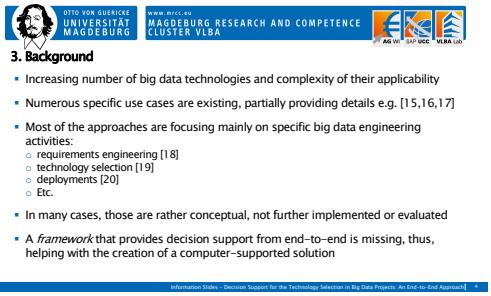
[SKC16]	UI	IM	FS	C	CC	RE	SC	SY	DO	FT	AV	S	R
UI	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6	1.0	1.0	6	1.0
IM	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6	1.0	1.0	6	1.0
FS	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6	1.0	1.0	6	1.0
C	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6	1.0	1.0	6	1.0
CC	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6	1.0	1.0	6	1.0
RE	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6	1.0	1.0	6	1.0
SC	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6	1.0	1.0	6	1.0
SY	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6	1.0	1.0	6	1.0
DO	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1.0	1/6	1/6	1.0	1/6
FT	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6.0	1.0	1.0	6	1.0
AV	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6.0	1.0	1.0	6	1.0
S	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1/6	1.0	1/6	1/6	1.0	1/6
R	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	6.0	1.0	1.0	6.0	1.0

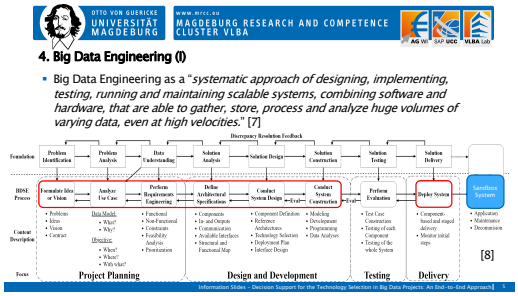
Table A.10: The matrices for the NFRs comparison, conducted for Eval 3

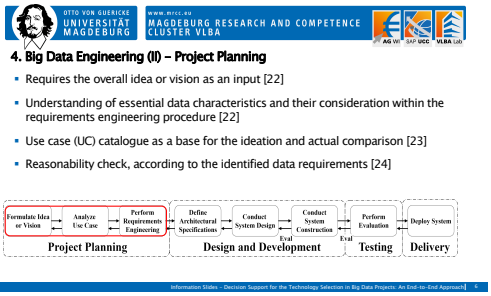


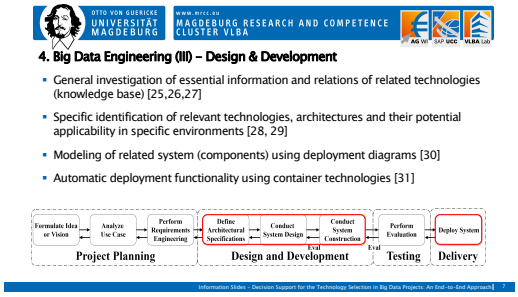


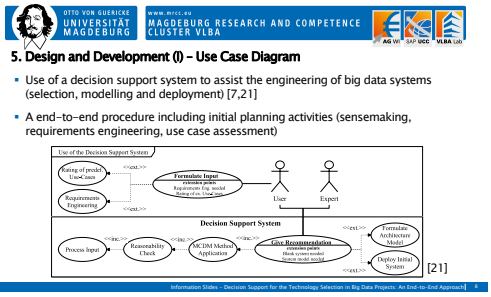


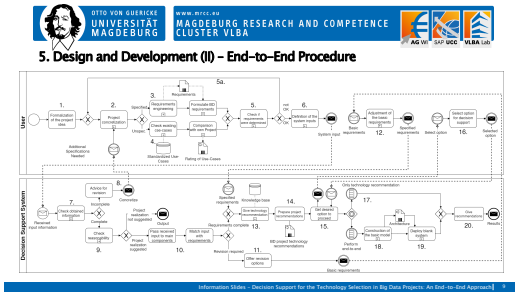












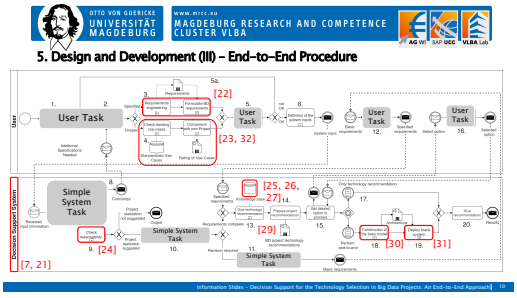


Figure A.4: Presentation for the interview, slides 1-10

5. Design and Development (IV) – Component Identification for a System

Name	Functionality
Sensemaking Component	Based on the given input information this component identifies the general sensibility of a big data project realization, referring to a combined use of big data technologies. It provides a general recommendation whether further specifications appear to be reasonable or not. In doing so it shall sensibilize potential decision-makers and allow them a *quick-check* for their planned endeavor.
Use Case Comparsion Component	The processed input information by the user are used for acquiring knowledge about a possible implementation of big data technologies. In the case that a user has no concrete idea about the project or its structuring, in terms of necessary requirements, this component can be used. By means of a simple comparison of a short description, of established use cases, as well as their requested functionality, a rough understanding of the size and complexity of such an undertaking can be empathized.
MCDM-Component	The selection of a technology can depend on a variety of information that must be taken into account. In addition to simple binary criteria with regard to the functionalities the underlying system shall fulfill, these include non-functional requirements. By means of a multi-criteria decision making (MCDM) approach, the latter can be considered and compared with each other. As a result, different recommendations are provided in a ranked order that describe sensible constellations of single or multiple big data technologies.
Modeling Component	The visualization and representation of the components of a system architecture is a major challenge, especially for decision makers. Particularly in the context of big data, special aspects must be taken into account that not only increase the size but also the complexity of possible diagrams. With the help of this component, it should be possible to create and distribute big data diagrams in a simple manner in order to provide a visual overview of a imaginable implementation.
Automatic Deployment Component	The goal of discussing individual big data technologies and their combination is in most cases not only of a conceptual nature. In the aftermath of the decision support, the recommended technologies are often to be implemented and tested to get a better impression of their actual usability and functionality. With the help of this component, individual solutions are deployed automatically without the decision maker having to demonstrate a profound understanding of the installation.
Knowledge Base	All the necessary information for dealing with big data and its technologies must be stored in a comprehensive knowledge base. In addition to the pure functionality of being able to provide information in a targeted manner, modifications must also be permitted that allow sustainable and extensive use.

5. Design and Development (V) – Sensemaking Component

- General identification of relevant characteristics to improve the requirements engineering activities
- Use of the arithmetic mean to calculate the magnitude index of a potential project
- Identification of three core characteristics and three additional ones
- Use of a NULL layer for characteristics, which are not addressed in a project
- Assessment value>1.33 generally recommended use of big data technologies

$$\text{Assessment value} = \bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$
$$\text{with} \begin{cases} x = \text{assigment of the characteristics} \\ n = \text{length of the vector} \\ x_i = i_{th}\text{ entry of the given vector} \end{cases}$$

[22,24]

5. Design and Development (VI) – Use Case Component

- Tremendous amount of use case descriptions, with huge differences in terms of information density and complexity
- Varying in scope, relevant requirements and used systems
- Thoroughly conducted literature research and use case analysis to identify comprehensive big data use case description
- Feature engineering, clustering and qualitative analysis created to obtain overarching use case clusters

→Creation of standard use case description of big data projects covering detailed information to particular cases and relevant (functional/non-functional) requirements

[23,32]

5. Design and Development (VII) – Use Case Component

- **Non-functional requirements**: UI: User Interface; IM: Installation and Maintenance Effort; DS: Documentation and Support; FS: Flexibility and Scalability; FT: Fault Tolerance; C: Cost; CC: Computational Complexity; RE: Regulations; SC: Storage Capacity; SY: Security; PM: Processing Method; AV: Availability; S: Sustainability; R: Reliability
- **Functional requirements**: AA: Automation Acting; BP: Batch Processing; CM: cluster management; CP: Consistency Preservation; A:Data Aggregation; CF: Data Classification; CL: Data Cleaning; CU: Data Clustering; F: Data Formatting; DM: Data Mining Algorithm Support; P: Data Pipelining; DS: Data Selection; ST: Data Streaming; Data Visualization; EP: event-data processing; ML: Machine Learning, MH: Message Handling; MO: Monitoring; NP: Near Real Time Processing; PP: Parallel Processing, RT: Real-Time Processing, RC: Recovery Mechanics; RP: Reporting; RM: Resource Management; SS: Store Semi-Structured Data; SD: Store Structured Data; SU: Store Unstructured Data; SP: Streaming Processing; SL: Support Scripting Language
- Binary or non-binary identification for each Standard Use Case (SUC)

№	Title	Non-Functional Requirements (NFR)											Functional Requirements (FR)			
		UI	IM	FS	C	CC	RE	SC	SY	DS	FT	AV	S	R	DI	DP	DA	DD	SO
1.	Data Analysis Improvement	5	3	5	4	4	3	5	3	3	3	5	2	5	EP, MH,P,ST-SD,SS,SU	AG,CF,CL, CU,F	MH,NP, PP,RT,SP, DM, ML	RP, V	CM,MO, RC,RM, SL
2.	Batch Mode Sensor Data Analysis	5	4	5	2	2	2	5	2	2	2	3	2	5	SD,SU	AC,CF,CL, U,F	BP,PP, DM, ML	RP, V	-
3.	Smart City	5	5	5	4	3	4	5	4	3	4	5	3	5	P,DS,SD,SS,SU	CF,CL, CU,F	MH,NP, RT, ML	RP, V	CM, SL
4.	Multi-Level Problems	3	3	5	3	3	3	3	5	3	5	4	3	4	EP,P,ST,SU	-	BP,RT, DM, ML	-	MO,RC
5.	Expand Data Sourcing	5	5	5	3	4	2	5	5	3	5	5	2	5	DS,SD,SS,SU	CL,F	BP, NP	RP,V	CM,MO, RC,RM, SL
6.	Data Connection	5	5	5	3	3	3	5	2	3	2	5	4	5	EP,P,DS,SD,SU	CF,CL, CU,F	DM, ML, BP	RP,V	AA, RC
7.	Decision Support	5	4	5	3	3	3	5	5	4	5	5	2	5	DS,SU,SD	CF,CL, CU,F	BP,RT, DM, ML	RP	-
8.	High-Speed Analysis	5	4	5	3	5	4	5	5	3	5	5	3	5	DS,ST,SD,SU	CF,CL, CU,F	PP,BP,RT, DM, ML	RP	-
9.	Process Optimization	3	5	5	5	5	5	5	5	2	5	5	2	5	EP,MH,DS,SS,SU	-	BP, RT DM	-	MO,RC

5. Design and Development (IX) – Use Case Component

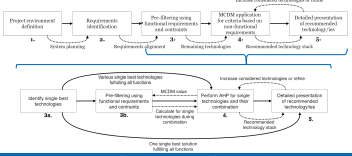
- Tremendous amount of use case descriptions, with huge differences in terms of information density and complexity
- Varying in scope, relevant requirements and used systems
- Thoroughly conducted literature research and use case analysis to identify comprehensive big data use case description
- Feature engineering, clustering and qualitative analysis created overarching clusters of cases

→Creation of standard use case description of big data projects covering detailed information to particular cases and relevant (functional/non-functional) requirements

[23,32]

5. Design and Development (V) – MCDM Component – Inference Engine

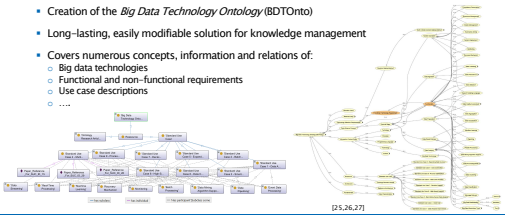
- Multi-stepped procedure that incorporates binary (FR) and non-binary (NFR) decisions for the selection and combination of big data technologies
- Use of the AHP for the identification and ranking of potential recommendations



[29]

5. Design and Development (V) – Knowledge Base Component

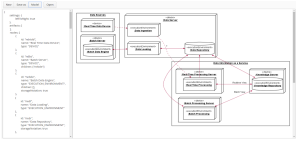
- Creation of the *Big Data Technology Ontology* (BDTOnto)
- Long-lasting, easily modifiable solution for knowledge management
- Covers numerous concepts, information and relations of:
  - Big data technologies
  - Functional and non-functional requirements
  - Use case descriptions
  - ....



[25,26,27]

5. Design and Development (V) – Modeling Component

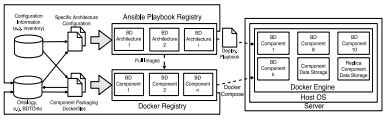
- Creation of deployment diagram profile that is automatically build using PlantUML and JSON files.
- The files (and diagrams) can easily be created, shared and modified



[30]

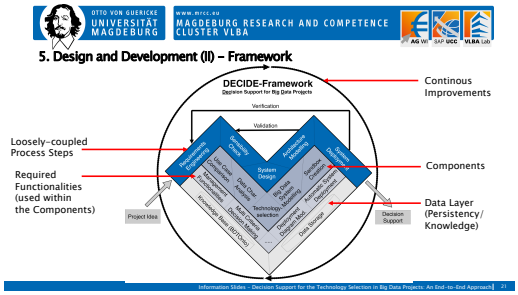
5. Design and Development (V) – Automatic Deployment Component

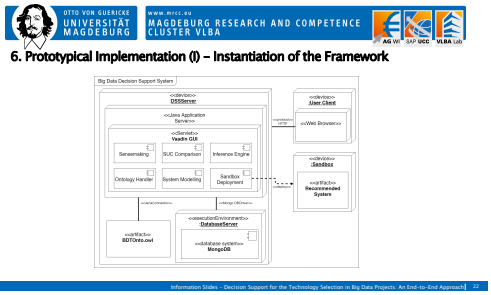
- Developed a concept for the creation and deployment of big data architectures
- Use of Ansible and Docker for the compound deployment of technologies
- Additional use of the developed ontology for the mapping of related information and needed files, e.g. playbooks.

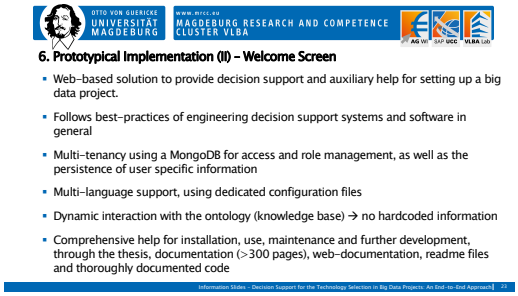


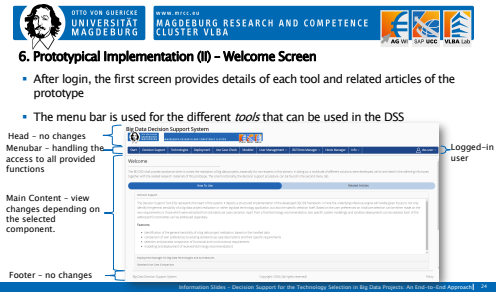
[31]

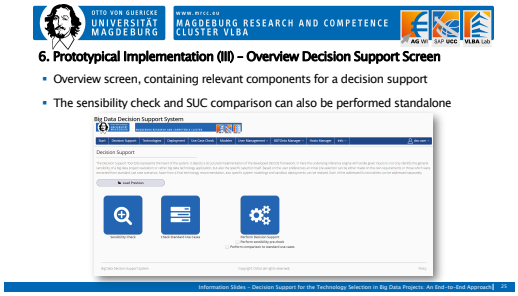
Figure A.5: Presentation for the interview, slides 11-20

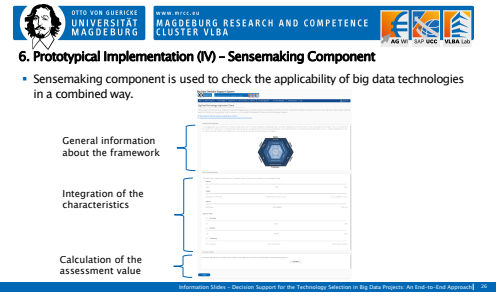


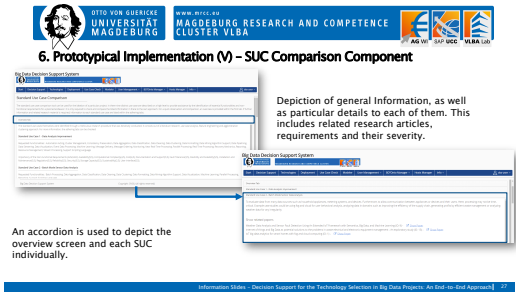


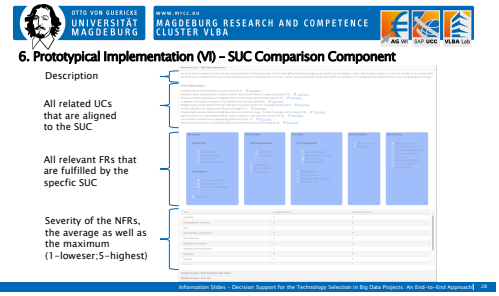


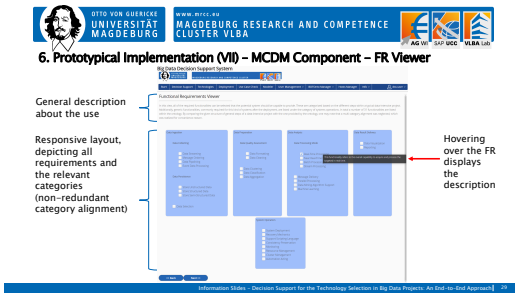












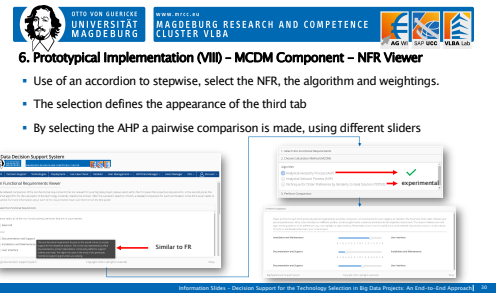


Figure A.6: Presentation for the interview, slides 21-30

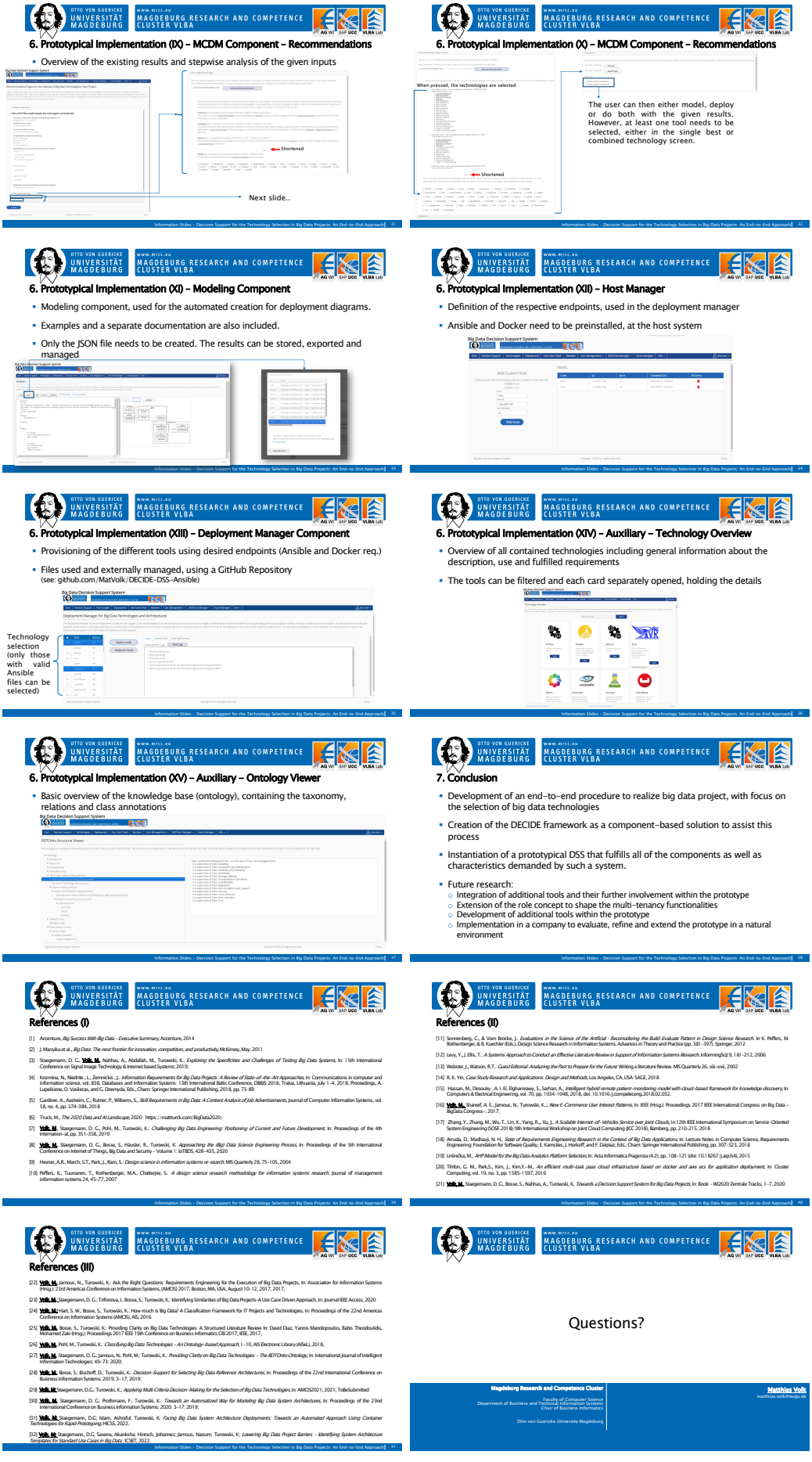


Figure A.7: Presentation for the interview, slides 31-42

Bibliography

[ABC+19] Abedjan, Z. et al. “Data Science in Healthcare: Benefits, Challenges and Opportunities.” In: *Data Science for Healthcare*. Ed. by Consoli, S., Reforgiato Recupero, D., and Petković, M. Cham: Springer International Publishing, 2019, pp. 3–38. DOI: 10.1007/978-3-030-05249-2_1.

[ASH+15] Abusharekh, A., Stewart, S. A., Hashemian, N., and Abidi, S. S. R. “H-DRIVE: A Big Health Data Analytics Platform for Evidence-Informed Decision Making.” In: *2015 IEEE International Congress on Big Data (BigData Congress)*. Ed. by Carminati, B. Piscataway: IEEE, 2015, pp. 416–423. DOI: 10.1109/BigDataCongress.2015.68.

[Ada15] Adams, W. C. “Conducting Semi-Structured Interviews.” In: *Handbook of Practical Program Evaluation*. Ed. by Newcomer, K. E., Hatry, H. P., and Wholey, J. S. Essential Texts for Nonprofit and Public Leadership and Management Ser. s.l.: Jossey-Bass, 2015, pp. 492–505. DOI: 10.1002/9781119171386.ch19.

[AIS+14] Agrawal, R., Imran, A., Seay, C., and Walker, J. “A Layer Based Architecture for Provenance in Big Data.” In: *Big Data (International Conference on Big Data)* (2014), pp. 29–31.

[AGH+16] Aguilera, A., Grunzke, R., Habich, D., Luong, J., Schollbach, D., Markwardt, U., and Garcke, J. “Advancing a Gateway Infrastructure for Wind Turbine Data Analysis.” In: *Journal of Grid Computing* 14.4 (2016), pp. 499–514. DOI: 10.1007/s10723-016-9376-9.

[AGM+15] Aguilera, A., Grunzke, R., Markwardt, U., Habich, D., Schollbach, D., and Garcke, J. “Towards an Industry Data Gateway: An Integrated Platform for the Analysis of Wind Turbine Data.” In: *IWSG 2015*. Piscataway, NJ: IEEE, 2015, pp. 62–66. DOI: 10.1109/IWSG.2015.8.

[ABD+19] Ahmad, A., Babar, M., Din, S., Khalid, S., Ullah, M. M., Paul, A., Goutham Reddy, A., and Min-Allah, N. “Socio-cyber network: The potential of cyber-physical system to define human behaviors using big data analytics.” In: *Future Generation Computer Systems* 92 (2019), pp. 868–878. DOI: 10.1016/j.future.2017.12.027.

[AKP+18] Ahmad, A., Khan, M., Paul, A., Din, S., Rathore, M. M., Jeon, G., and Choi, G. S. “Toward modeling and optimization of features selection in Big Data

based social Internet of Things." In: *Future Generation Computer Systems* 82 (2018), pp. 715–726. DOI: 10.1016/j.future.2017.09.028.

[ABS19] Alaei, A. R., Becken, S., and Stantic, B. "Sentiment Analysis in Tourism: Capitalizing on Big Data." In: *Journal of Travel Research* 58.2 (2019), pp. 175–191. DOI: 10.1177/0047287517747753.

[AKT+03] Albani, A., Keiblinger, A., Turowski, K., and Winnewisser, C. "Domain Based Identification and Modelling of Business Component Applications." In: Springer, Berlin, Heidelberg, 2003, pp. 30–45. DOI: 10.1007/978-3-540-39403-7_5.

[AIS+19] Alfian, G., Ijaz, M. F., Syafrudin, M., Syaekhoni, M. A., Fitriyani, N. L., and Rhee, J. "Customer behavior analysis using real-time data processing." In: *Asia Pacific Journal of Marketing and Logistics* 31.1 (2019), pp. 265–290. DOI: 10.1108/APJML-03-2018-0088.

[ACD+19] Alkahtani, M., Choudhary, A., De, A., and Harding, J. A. "A decision support system based on ontology and data mining to improve design using warranty data." In: *Computers & Industrial Engineering* 128 (2019), pp. 1027–1039. DOI: 10.1016/j.cie.2018.04.033.

[AGP+19] Alpoim, Â., Guimarães, T., Portela, F., and Santos, M. F. "Evaluation Model for Big Data Integration Tools." In: *New Knowledge in Information Systems and Technologies*. Ed. by Rocha, Á., Adeli, H., Reis, L. P., and Costanzo, S. Vol. 932. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, 2019, pp. 601–610. DOI: 10.1007/978-3-030-16187-3_58.

[ANW15] Alshboul, Y., Nepali, R., and Wang Yong. "Big Data LifeCycle: Threats and Security Model." In: *SIGSEC (Information Systems Security, Assurence and Privacy)* (2015).

[ANN+17] Altarturi, H. H., Ng, K.-Y., Ninggal, M. I. H., Nazri, A. S. A., and Ghani, A. A. A. "A requirement engineering model for big data software." In: *IEEE 2017 Conference on Big*. 2017, pp. 111–117. DOI: 10.1109/ICBDAA.2017.8284116.

[Alt76] Alter, S. *Computer aided decision making in organizations: a decision support systems typology*. 1976.

[APA+15] Alzate Calderón, W., Pomares Quimbaya, A., A. Gonzalez, R., and Muñoz, O. M. "BigTexts - A Framework for the Analysis of Electronic Health Record Narrative Texts based on Big Data Technologies." In: *Proceedings of the 1st International Conference on Information and Communication Technologies for Ageing Well and e-Health*. SCITEPRESS - Science and and Technology Publications, 2015, pp. 129–136. DOI: 10.5220/0005434101290136.

[AHN+20] Amanullah, M. A., Habeeb, R. A. A., Nasaruddin, F. H., Gani, A., Ahmed, E., Nainar, A. S. M., Akim, N. M., and Imran, M. "Deep learning and big data technologies for IoT security." In: *Computer Communications* 151 (2020), pp. 495–517. DOI: 10.1016/j.comcom.2020.01.016.

[AGP17] Amini, S., Gerostathopoulos, I., and Prehofer, C. "Big data analytics architecture for real-time traffic control." In: *5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems*. Piscataway, NJ: IEEE, 2017, pp. 710–715. DOI: 10.1109/MTITS.2017.8005605.

[Anc12] Ancona, D. "Framing and Acting in the Unknown." In: *The Hnadbook for Teaching Leadership*. Ed. by Snook, S., Nohira, N., and Khurana, R. Sage Publications, 2012, pp. 198–217.

[AAA+20] Anthony Jnr, B., Abbas Petersen, S., Ahlers, D., and Krogstie, J. "Big data driven multi-tier architecture for electric mobility as a service in smart cities." In: *International Journal of Energy Sector Management* 14.5 (2020), pp. 1023–1047. DOI: 10.1108/IJESM-08-2019-0001. URL: https://www.emerald.com/insight/content/doi/10.1108/IJESM-08-2019-0001/full/pdf?skipTracking=trueUR%20-%20https://www.emerald.com/insight/content/doi/10.1108/IJESM-08-2019-0001/full/html?skipTracking=true.

[Av09] Antoniou, G. and van Harmelen, F. "Web Ontology Language: OWL." In: *Handbook on Ontologies*. Ed. by Staab, S. and Studer, R. Springer, Berlin, Heidelberg, 2009, pp. 91–110. DOI: 10.1007/978-3-540-92673-3_4.

[Apa22a] Apache. *Apache Hadoop*. 28.03.2022. URL: https://hadoop.apache.org/ (visited on 05/04/2022).

[Apa22b] Apache. *Apache Jena - Home*. 23.04.2022. URL: https://jena.apache.org/ (visited on 05/21/2022).

[Apa] Apache Software Foundation. *Bigtop – Apache Bigtop*. URL: https://bigtop.apache.org/ (visited on 04/18/2022).

[AM18] Arruda, D. and Madhavji, N. H. "State of Requirements Engineering Research in the Context of Big Data Applications." In: *Requirements Engineering: Foundation for Software Quality*. Ed. by Kamsties, E., Horkoff, J., and Dalpiaz, F. Vol. 10753. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 307–323. DOI: 10.1007/978-3-319-77243-1_20.

[AM19] Arruda, D. and Madhavji, N. H. "QualiBD: A Tool for Modelling Quality Requirements for Big Data Applications." In: *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 9.12.2019 - 12.12.2019, pp. 5977–5979. DOI: 10.1109/BigData47090.2019.9006294.

[ABD+17] Artac, M., Borovssak, T., Di Nitto, E., Guerriero, M., and Tamburri, D. A. "DevOps: Introducing Infrastructure-as-Code." In: *2017 IEEE/ACM 39th International Conference on Software Engineering companion*. Piscataway, NJ: IEEE, 2017, pp. 497–498. DOI: 10.1109/ICSE-C.2017.162.

[ACB+15a] Assunção, M. D., Calheiros, R. N., Bianchi, S., Netto, M. A., and Buyya, R. "Big Data computing and clouds: Trends and future directions." In: *Journal of Parallel and Distributed Computing* 79-80 (2015), pp. 3–15. DOI: 10.1016/j.jpdc.2014.08.003.

[ACB+15b] Assunção, M. D., Calheiros, R. N., Bianchi, S., Netto, M. A., and Buyya, R. "Big Data computing and clouds: Trends and future directions." In: *Journal of Parallel and Distributed Computing* 79-80 (2015), pp. 3–15. DOI: 10.1016/j.jpdc.2014.08.003.

[AL20] Ataei, P. and Litchfield, A. "Big Data Reference Architectures, a systematic literature review." In: *ACIS 2020 Proceedings* (2020). URL: https://aisel.aisnet.org/acis2020/30.

[ACD+18] Avvenuti, M., Cresci, S., Del Vigna, F., Fagni, T., and Tesconi, M. "CrisMap: a Big Data Crisis Mapping System Based on Damage Detection and Geoparsing." In: *Information Systems Frontiers* (2018). DOI: 10.1007/s10796-018-9833-z.

[AS08] Azevedo, A. and Santos, M. F. "KDD, SEMMA and CRISP-DM: a parallel overview." In: *IADIS European Conf. Data Mining*. 2008.

[AAR+16] Azimi, I., Anzanpour, A., Rahmani, A. M., Liljeberg, P., and Salakoski, T. "Medical warning system based on Internet of Things using fog computing." In: *2016 International Workshop on Big Data and Information Security (IWBIS)*. Piscataway, NJ: IEEE, 2016, pp. 19–24. DOI: 10.1109/IWBIS.2016.7872884.

[BRA+18] Babar, M., Rahman, A., Arif, F., and Jeon, G. "Energy-harvesting based on internet of things and big data analytics for smart health monitoring." In: *Sustainable Computing: Informatics and Systems* 20 (2018), pp. 155–164. DOI: 10.1016/j.suscom.2017.10.009.

[BZA+19] Bahri, S., Zoghlami, N., Abed, M., and Tavares, J. M. R. S. "BIG DATA for Healthcare: A Survey." In: *IEEE Access* 7 (2019), pp. 7397–7408. DOI: 10.1109/ACCESS.2018.2889180.

[BD20] Barham, H. and Daim, T. "The use of readiness assessment for big data projects." In: *Sustainable Cities and Society* 60 (2020), p. 102233. DOI: 10.1016/j.scs.2020.102233. URL: https://www.sciencedirect.com/science/article/pii/S2210670720302201.

[Bar14] Bartlomiej Twardowski, D. R. "Multi-agent architecture for real-time Big Data processing." In: *IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)* 3 (2014), pp. 333–337.

[BS11] Baxter, G. and Sommerville, I. "Socio-technical systems: From design methods to systems engineering." In: *Interacting with Computers* 23.1 (2011), pp. 4–17. DOI: 10.1016/j.intcom.2010.07.003.

[Bec17] Becker, D. K. "Predicting outcomes for big data projects: Big Data Project Dynamics (BDPD): Research in progress." In: *2017 IEEE International Conference on Big Data*. Ed. by Nie, J.-Y., Obradovic, Z., Suzumura, T., Ghosh, R., Nambiar, R., and Wang, C. Piscataway, NJ: IEEE, 2017, pp. 2320–2330. DOI: 10.1109/BigData.2017.8258186.

[BJG14] Bedi, P., Jindal, V., and Gautam, A. "Beginning with big data simplified." In: *2014 International Conference on Data Mining and Intelligent Computing (ICDMIC)*. IEEE, 2014, pp. 1–7. DOI: 10.1109/ICDMIC.2014.6954229.

[BJC16] Bello-Orgaz, G., Jung, J. J., and Camacho, D. "Social big data: Recent achievements and new challenges." In: *Information Fusion* 28 (2016), pp. 45–59. DOI: 10.1016/j.inffus.2015.08.005.

[BHL+01] Berners-Lee, T., Hendler, J., Lassila, O., et al. "The semantic web." In: *Scientific american* 284.5 (2001), pp. 28–37.

[BMR+16] Bersani, M. M., Marconi, F., Rossi, M., and Erascu, M. "A tool for verification of big-data applications." In: *Proceedings of the 2nd International Workshop on Quality-Aware DevOps*. Ed. by Ardagna, D. ACM Digital Library. New York, NY: ACM, 2016, pp. 44–45. DOI: 10.1145/2945408.2945419.

[BOQ+16] Bilal, M., Oyedele, L. O., Qadir, J., Munir, K., Ajayi, S. O., Akinade, O. O., Owolabi, H. A., Alaka, H. A., and Pasha, M. "Big Data in the construction industry: A review of present status, opportunities, and future trends." In: *Advanced Engineering Informatics* 30.3 (2016), pp. 500–521. DOI: 10.1016/j.aei.2016.07.001.

[BBL18] Billot, R., Bothorel, C., and Lenca, P. "Introduction to Big Data and Its Applications in Insurance: Chapter 1." In: (2018), pp. 1–25. (Visited on 09/02/2019).

[BO09] Birkmeier, D. and Overhage, S. "On Component Identification Approaches – Classification, State of the Art, and Comparison." In: Springer, Berlin, Heidelberg, 2009, pp. 1–18. DOI: 10.1007/978-3-642-02414-6_1.

[BIT12] BITKOM. *Big Data im Praxiseinsatz - Szenarien, Beispiele, Effekte*. Ed. by BITKOM. Berlin, 2012.

[BIT14] BITKOM. *Big-Data-Technologien–Wissen für Entscheider*. Ed. by BITKOM. Berlin, 2014. (Visited on 03/05/0202).

[BD18] Blazquez, D. and Domenech, J. "Big Data sources and methods for social and economic analyses." In: *Technological Forecasting and Social Change* 130 (2018), pp. 99–113. DOI: 10.1016/j.techfore.2017.07.027.

[BS05] Blomqvist, E. and Sandkuhl, K. "Patterns in Ontology Engineering: Classification of Ontology Patterns." In: *Proceedings / ICEIS 2005, Seventh International Conference on Enterprise Information Systems*. Ed. by Weghorn, H. Setúbal: INSTICC, 2005, pp. 413–416. DOI: 10.5220/0002518804130416.

[BT15] Boci, E. and Thistlethwaite, S. "A novel Big Data architecture in support of ADS-B data analytic." In: *ICNS (Integrated Communication, Navigation and Surveillance Conference)* (2015), pp. C1-1 - C1–8.

[Boo12] Boone, D. "Analyzing Likert Data." In: *The Journal of Extension* 50.2 (2012). URL: https://tigerprints.clemson.edu/joe/vol50/iss2/48.

[BSH16] Borodo, S. M., Shamsuddin, S. M., and Hasan, S. "Big Data Platforms and Techniques." In: *Indonesian Journal of Electrical Engineering and Computer Science* 1.1 (2016), pp. 191–200. DOI: 10.11591/ijeecs.v1.i1.pp191-200.

[BHA+17] Boutkhoum, O., Hanine, M., Agouti, T., and Tikniouine, A. "A decision-making approach based on fuzzy AHP-TOPSIS methodology for selecting the appropriate cloud solution to manage big data projects." In: *International Journal of System Assurance Engineering and Management* 8.2 (2017), pp. 1237–1253.

[Bra19] Brady, H. E. "The Challenge of Big Data and Data Science." In: *Annual Review of Political Science* 22.1 (2019), pp. 297–323. DOI: 10.1146/annurev-polisci-090216-023229.

[BS16] Brans, J.-P. and Smet, Y. de. "PROMETHEE Methods." In: *Multiple criteria decision analysis*. Ed. by Greco, S., Ehrgott, M., and Figueira, J. R. Vol. 233. International Series in Operations Research & Management Science. New York et al.: Springer, 2016, pp. 187–219. DOI: 10.1007/978-1-4939-3094-4_6.

[Bre00a] Brewer, E. A. *Towards Robust Distributed Systems*. 2000. URL: http://www.cs.berkeley.edu/~brewer/cs262b-2004/PODC-keynote.pdf.

[Bre00b] Brewer, E. A. "Towards robust distributed systems (abstract)." In: *the nineteenth annual ACM symposium*. Ed. by Neiger, G. 2000, p. 7. DOI: 10.1145/343477.343502.

[BK16] Bronson, K. and Knezevic, I. "Big Data in food and agriculture." In: *Big Data & Society* 3.1 (2016). DOI: 10.1177/2053951716648174.

[BDS18] Burmeister, F., Drews, P., and Schirmer, I. "Towards an Extended Enterprise Architecture Meta-Model for Big Data - A Literature-based Approach." In: *AMCIS (Americas Conference on Information Systems)* 24 (2018).

[CSG+17] Campos, J., Sharma, P., Gabiria, U. G., Jantunen, E., and Baglee, D. "A Big Data Analytical Architecture for the Asset Management." In: *CIRP* 64 (2017), pp. 369–374. DOI: 10.1016/j.procir.2017.03.019.

[CFC+18] Canito, A., Fernandes, M., Conceição, L., Praça, I., and Marreiros, G. "A Big Data Platform for Industrial Enterprise Asset Value Enablers." In: *DCAI (International Conference on Distributed Computing and Artificial Intelligence)* 15 (2018), pp. 145–154. DOI: 10.1007/978-3-319-94649-8_18.

[CL18] Casale, G. and Li, C. "Enhancing Big Data Application Design with the DICE Framework." In: *Advances in Service-Oriented and Cloud Computing*. Ed. by Mann, Z. Á. and Stolz, V. Vol. 824. Communications in computer and information science. Cham: Springer International Publishing, 2018, pp. 164–168. DOI: 10.1007/978-3-319-79090-9_13.

[CPC+20] Castellanos, C., Perez, B., Correal, D., and Varela, C. A. "A Model-Driven Architectural Design Method for Big Data Analytics Applications." In: *7281-7415* (2020), pp. 89–94. DOI: 10.1109/ICSA-C50368.2020.00026. URL: https://www.computer.org/csdl/proceedings-article/icsa-c/2020/09095552/1jXvoWD2noQ.

[CGD15] Cato, P., Golzer, P., and Demmelhuber, W. "An investigation into the implementation factors affecting the success of big data systems." In: *Proceedings of 11th International Conference on Innovations in Information Technology (IIT)*. IEEE, 2015, pp. 134–139. DOI: 10.1109/INNOVATIONS.2015.7381528.

[Ceu12] Ceusters, W. *An information artifact ontology perspective on data collections and associated representational artifacts*. 2012.

[CDG+06] Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., Chandra, T., Fikes, A., and Gruber, R. E. "Bigtable: A Distributed Storage System for Structured Data." In: *7th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. 2006, pp. 205–218.

[CG18] Chang, W. L. and Grady, N. *NIST Big Data Interoperability Framework: Volume 3 Use Cases and General Requirements*. 2018. URL: https://doi.org/10.6028/NIST.SP.1500-3r1.

[CG19a] Chang, W. L. and Grady, N. *NIST Big Data Interoperability Framework: Volume 1, Definitions*. 2019. URL: https://doi.org/10.6028/NIST.SP.1500-1r2.

[CG19b] Chang, W. L. and Grady, N. *NIST Big Data Interoperability Framework: Volume 6, Reference Architecture*. 2019. URL: https://doi.org/10.6028/NIST.SP.1500-6r2.

[CCK+00] Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., and Wirth, R. *CRISP-DM 1.0: Step-by-step data mining guide*. 2000.

[CHN12] Chaudron, M. R. V., Heijstek, W., and Nugroho, A. "How effective is UML modeling ?" In: *Software & Systems Modeling* 11.4 (2012), pp. 571–580. DOI: 10.1007/s10270-012-0278-4.

[Che16] Chen, C.-M. "Use cases and challenges in telecom big data analytics." In: *APSIPA Transactions on Signal and Information Processing* 5 (2016). DOI: 10.1017/ATSIP.2016.20.

[CKG+16] Chen, H.-M., Kazman, R., Garbajosa, J., and Gonzalez, E. "Toward big data value engineering for innovation." In: *BIGDSE (International Workshop on Big Data Software Engineering)* 2 (2016), pp. 44–50. DOI: 10.1145/2896825.2896837.

[CKG+17] Chen, H.-M., Kazman, R., Garbajosa, J., and Gonzalez, E. "Big Data Value Engineering for Business Model Innovation." In: *HICSS (Hawaii International Conference on System Sciences)* 50 (2017), pp. 5921–5930.

[CKH16a] Chen, H.-M., Kazman, R., and Haziyev, S. "Agile Big Data Analytics Development: An Architecture-Centric Approach." In: *HICSS (Hawaii International Conference on System Sciences)* 49 (2016), pp. 5378–5387. DOI: 10.1109/HICSS.2016.665.

[CKH16b] Chen, H.-M., Kazman, R., and Haziyev, S. "Agile Big Data Analytics for Web-Based Systems: An Architecture-Centric Approach." In: *IEEE Transactions on Big Data* 2.3 (2016), pp. 234–248. DOI: 10.1109/TBDATA.2016.2564982.

[CKH+15] Chen, H.-M., Kazman, R., Haziyev, S., and Hrytsay, O. "Big Data System Development: An Embedded Case Study with a Global Outsourcing Firm." In: *2015 IEEE/ACM 1st International Workshop on Big Data Software Engineering*. IEEE, 2015, pp. 44–50. DOI: 10.1109/BIGDSE.2015.15.

[CKH16c] Chen, H.-M., Kazman, R., and Haziyev, S. "Strategic Prototyping for Developing Big Data Systems." In: *IEEE Software* 33.2 (2016), pp. 36–43. DOI: 10.1109/MS.2016.65.

[CSK+17] Chen, H.-M., Schütz, R., Kazman, R., and Matthes, F. "How Lufthansa Capitalized on Big Data for Business Model Renovation." In: *MIS Quarterly Executive* 16.1 (2017).

[CCS12] Chen, H., Chiang, R. H., and Storey, V. C. "Business Intelligence and Analytics: From Big Data to Big Impact." In: *MIS Quaterly* 36.4 (2012), p. 1165. DOI: 10.2307/41703503.

[CML14] Chen, M., Mao, S., and Liu, Y. "Big Data: A Survey." In: *Mobile Networks and Applications* 19.2 (2014), pp. 171–209. DOI: 10.1007/s11036-013-0489-0.

[CLC+15] Cheng, B., Longo, S., Cirillo, F., Bauer, M., and Kovacs, E. "Building a Big Data Platform for Smart Cities: Experience and Lessons from Santander." In: *2015 IEEE International Congress on Big Data*. IEEE, 2015, pp. 592–599. DOI: 10.1109/BigDataCongress.2015.91.

[CP16] Comuzzi, M. and Patel, A. "How organisations leverage Big Data: a maturity model." In: *Industrial Management & Data Systems* 116.8 (2016), pp. 1468–1492. DOI: 10.1108/IMDS-12-2015-0495.

[Con14] Conrads, R. "In sieben Schritten zum erfolgreichen Big-Data-Projekt." In: *Informatik-Spektrum* 37.2 (2014), pp. 127–131. DOI: 10.1007/s00287-013-0727-7.

[COR17] Côrte-Real, N., Oliveira, T., and Ruivo, P. "Assessing business value of Big Data Analytics in European firms." In: *Journal of Business Research* 70.C (2017), pp. 379–390.

[CY16] Costa, C. and Yasmina Santos, M. "BASIS: A Big Data Architecture for Smart Cities." In: *SAI (SAI Computing Conference)* (2016), pp. 1247–1256.

[CE97] Cox, M. and Ellsworth, D. "Managing big data for scientific visualization." In: *ACM Siggraph*. Vol. 97. 1997.

[CC05] Cristani, M. and Cuel, R. "A Survey on Ontology Creation Methodologies." In: *International Journal on Semantic Web and Information Systems (IJSWIS)* 1.2 (2005), pp. 49–69. DOI: 10.4018/jswis.2005040103. URL: https://www.igi-global.com/gateway/article/full-text-pdf/2808&riu=true.

[DHO+13] Daas, D., Hurkmans, T., Overbeek, S., and Bouwman, H. "Developing a decision support system for business model design." In: *Electronic Markets* 23.3 (2013), pp. 251–265. DOI: 10.1007/s12525-012-0115-1. URL: https://link.springer.com/article/10.1007/s12525-012-0115-1.

[DVS+22] Daase, C., Volk, M., Staegemann, D., and Turowski, K. "Addressing the Dichotomy of Theory and Practice in Design Science Research Methodologies." In: *Proceedings of the 17th International Conference on Design Science Research in Information Systems and Technology (DESRIST)*. 2022.

[Dar20] Darlan Florencio de Arruda. "Requirements Engineering in the Context of Big Data Software Applications." In: (2020). URL: https://ir.lib.uwo.ca/cgi/viewcontent.cgi?article=9262&context=etd (visited on 04/18/2022).

[DA18] Darwish, T. S. J. and Abu Bakar, K. "Fog Based Intelligent Transportation Big Data Analytics in The Internet of Vehicles Environment: Motivations, Architecture, Challenges, and Critical Issues." In: *IEEE Access* 6 (2018), pp. 15679–15701. DOI: 10.1109/ACCESS.2018.2815989.

[DL20a] Davoudian, A. and Liu, M. "Big Data Systems." In: *ACM Computing Surveys* 53.5 (2020), pp. 1–39. DOI: 10.1145/3408314.

[DL20b] Davoudian, A. and Liu, M. "Big Data Systems: A Software Engineering Perspective." In: *ACM Computing Surveys* 53.5 (2020), pp. 1–39. DOI: 10.1145/3408314.

[DG04] Dean, J. and Ghemawat, S. "MapReduce: Simplified data processing on large clusters." In: (2004).

[DGL+13] Demchenko, Y., Grosso, P., Laat, C. de, and Membrey, P. "Addressing big data issues in Scientific Data Infrastructure." In: *2013 International Conference on Collaboration Technologies and Systems (CTS)*. IEEE, 2013, pp. 48–55. DOI: 10.1109/CTS.2013.6567203.

[DLM14] Demchenko, Y., Laat, C. de, and Membrey, P. "Defining architecture components of the Big Data Ecosystem." In: *2014 International Conference on Collaboration Technologies and Systems (CTS)*. IEEE, 2014, pp. 104–112. DOI: 10.1109/CTS.2014.6867550.

[DNM13] Demchenko, Y., Ngo, C., and Membrey, P. "Architecture Framework and Components for the Big Data Ecosystem." In: (2013).

[DVv+20] Deshai, N., Venkataramana, S., v. d. Sekhar, B. S., Srinivas, K., and Saradhi Varma, G. P. "A Study on Big Data Processing Frameworks: Spark and Storm." In: *Smart Intelligent Computing and Applications*. Ed. by Satapathy, S. C. Vol. 160. Smart Innovation, Systems and Technologies. Singapore: Springer Singapore, 2020, pp. 415–424. DOI: 10.1007/978-981-32-9690-9_43.

[DGK+11] Desic, S., Gvozdanovic, D., Kusek, M., and Huljenic, D. "Advantages of UML-based object-oriented system development." In: *MIPRO Meeting* (2011).

[Dic17] Dick, J. *Requirements Engineering*. 4th ed. 2017. Springer eBook Collection Computer Science. Cham: Springer, 2017. DOI: 10.1007/978-3-319-61073-3.

[Dij13] Dijcks, J.-P. *Oracle: Big Data for the Enterprise*. Ed. by Oracle Corporation. Redwood, 2013.

[DGP+15] Din, S., Ghayvat, H., Paul, A., Ahmad, A., Rathore, M. M., and Shafi, I. "An Architecture to Analyze Big data in the Internet of Things." In: *ICST (International Conference on Sensing Technology)* 9 (2015), pp. 677–682.

[DG16] Dineshreddy, V. and Gangadharan, G. R. "Towards an "Internet of Things" framework for financial services sector." In: *3rd International Conference on Recent Advances in Information Technology (RAIT)*. Ed. by Kumar, C., Banka, H., and Ramesh, D. Piscataway: IEEE, 2016, pp. 177–181. DOI: 10.1109/RAIT.2016.7507897.

[Doc22] Docker. *Explore Docker's Container Image Repository — Docker Hub*. 2022. URL: https://hub.docker.com/search?q= (visited on 05/12/2022).

[Don17] Donoho, D. "50 Years of Data Science." In: *Journal of Computational and Graphical Statistics* 26.4 (2017), pp. 745–766. DOI: 10.1080/10618600.2017.1384734.

[Don13] Dontha, R. "Who came up with the name Big Data?" In: *TechTarget* (2017-01-13). URL: https://www.datasciencecentral.com/who-came-up-with-the-name-big-data/ (visited on 03/15/2022).

[Dox22] Doxygen. *Doxygen*. 15.05.2022. URL: https://www.doxygen.nl/ (visited on 05/22/2022).

[DB15] Dutta, D. and Bose, I. "Managing a Big Data project: The case of Ramco Cements Limited." In: *International Journal of Production Economics* 165 (2015), pp. 293–306. DOI: 10.1016/j.ijpe.2014.12.032.

[Ebe14] Ebert, C. *Systematisches Requirements Engineering: Anforderungen ermitteln, spezifizieren, analysieren und verwalten*. 5., überarb. Aufl. Heidelberg: dpunkt, 2014.

[EJQ17] Eine, B., Jurisch, M., and Quint, W. "Ontology-Based Big Data Management." In: *MDPI Systems* 3 (2017). DOI: 10.3390/systems5030045.

[EEE+18] El Alaoui El Abdallaoui, H., El Fazziki, A., Ennaji, F. Z., and Sadgal, M. "Decision Support System for the Analysis of Traffic Accident Big Data." In: *2018 14th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS 2018)*. Piscataway, NJ: IEEE, 2018, pp. 514–521. DOI: 10.1109/SITIS.2018.00084.

[EH22] EL Haoud, N. and Hali, O. "AHP Approach for Selecting Adequate Big Data Analytics Platform." In: *Intelligent Systems Design and Applications*. Ed. by Abraham, A., Gandhi, N., Hanne, T., Hong, T.-P., Nogueira Rios, T., and Ding, W. Vol. 418. Springer eBook Collection. Cham: Springer International Publishing and Imprint Springer, 2022, pp. 667–675. DOI: 10.1007/978-3-030-96308-8_62.

[ES16] Emmanuel, I. and Stanier, C. "Defining Big Data." In: *BDCA (International Conference on Big Data and Advanced Wireless Technologies)* 5 (2016), pp. 1–6. DOI: 10.1145/3010089.3010090.

[Eom08] Eom, S. B. "Reference Disciplines of Decision Support Systems." In: *Handbook on Decision Support Systems 1*. Ed. by Holsapple, C. W. SpringerLink Bücher. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 141–159. DOI: 10.1007/978-3-540-48713-5_8.

[Erl16] Erl, T. *Big data fundamentals: Concepts, drivers & techniques*. Safari Tech Books Online. Boston: Prentice Hall ServiceTech Press, 2016.

[EBF+21] Ermakova, T., Blume, J., Fabian, B., Fomenko, E., Berlin, M., and Hauswirth, M. *Beyond the Hype: Why Do Data-Driven Projects Fail?* Honolulu, HI: University of Hawai'i at Manoa Hamilton Library, 2021. DOI: 10.24251/HICSS.2021.619.

[EJA17] Escobedo, G., Jacome, N., and Arroyo-Figueroa, G. "Big Data & Analytics to Support the Renewable Energy Integration of Smart Grids - Case Study: Power Solar Generation." In: *IoTBDS 2017*. Ed. by Ramachandran, M., Méndez Muñoz, V., Kantere, V., Wills, G., Walters, R., and Chang, V. Setúbal, Portugal: SCITEPRESS, 2017, pp. 267–275. DOI: 10.5220/0006297502670275.

[FB19] Fahmideh, M. and Beydoun, G. "Big data analytics architecture design—An application in manufacturing systems." In: *Computers & Industrial Engineering* 128 (2019), pp. 948–963. DOI: 10.1016/j.cie.2018.08.004.

[Fal10] Falconer, S. *OntoGraf - Protege Wiki*. 10.04.2010. URL: https://protegewiki.stanford.edu/wiki/OntoGraf (visited on 05/03/2022).

[FJJ+18] Farshidi, S., Jansen, S., Jong, R. de, and Brinkkemper, S. "A decision support system for software technology selection." In: *Journal of Decision Systems* 27.sup1 (2018), pp. 98–110. DOI: 10.1080/12460125.2018.1464821.

[FPS96] Fayyad, U., Piatetsky-Shapiro, G., and Smyth, P. "From Data Mining to Knowledge Discovery in Databases." In: *AI Magazine* 17.3 (1996), p. 37. DOI: 10.1609/aimag.v17i3.1230.

[FV15] Fekete, D. and Vossen, G. "The GOBIA Method: Towards Goal-Oriented Business Intelligence Architectures." In: *undefined* (2015). URL: https://www.semanticscholar.org/paper/The-GOBIA-Method%3A-Towards-Goal-Oriented-Business-Fekete-Vossen/fd64738c9015e8fa6643359b1cad007c806f6368.

[FRM15] Feller, E., Ramakrishnan, L., and Morin, C. "Performance and energy efficiency of big data applications in cloud environments: A Hadoop case study." In: *Journal of Parallel and Distributed Computing* 79-80 (2015), pp. 80–89. DOI: 10.1016/j.jpdc.2015.01.001.

[FFR+15] Felter, W., Ferreira, A., Rajamony, R., and Rubio, J. "An updated performance comparison of virtual machines and Linux containers." In: Piscataway, NJ: IEEE, 2015, pp. 171–172. DOI: 10.1109/ISPASS.2015.7095802.

[FGJ97a] Fernández, M., Gómez-Pérez, A., and Juristo, N. "Methontology: From Ontological Art Towards Ontological Engineering." In: (1997).

[FGJ97b] Fernández-López, M., Gómez-Pérez, A., and Juristo, N. "METHONTOLOGY: From Ontological Art Towards Ontological Engineering." In: *undefined* (1997).

[FLL+16] Fiannaca, A., La Paglia, L., La Rosa, M., Messina, A., Storniolo, P., and Urso, A. "Integrated DB for Bioinformatics: A Case Study on Analysis of Functional Effect of MiRNA SNPs in Cancer." In: *Information technology in bio- and medical informatics*. Ed. by Renda, E. Vol. 9832. Lecture Notes in Computer Science Information systems and applications, incl. Internet/Web, and HCI. Springer, 2016, pp. 214–222. DOI: 10.1007/978-3-319-43949-5_17.

[FGR+13] Figueira, J. R., Greco, S., Roy, B., and Słowiński, R. "An Overview of ELECTRE Methods and their Recent Extensions." In: *Journal of Multi-Criteria Decision Analysis* 20.1-2 (2013), pp. 61–85. DOI: 10.1002/mcda.1482.

[FMR16] Figueira, J. R., Mousseau, V., and Roy, B. "ELECTRE Methods." In: *Multiple criteria decision analysis*. Ed. by Greco, S., Ehrgott, M., and Figueira, J. R. Vol. 233. International Series in Operations Research & Management Science. New York et al.: Springer, 2016, pp. 155–185. DOI: 10.1007/978-1-4939-3094-4_5.

[FSW+18] Filipiak, D., Stróżyna, M., Wecel, K., and Abramowicz, W. "Big Data for Anomaly Detection in Maritime Surveillance: Spatial AIS Data Analysis for Tankers." In: *Zeszyty Naukowe Akademii Marynarki Wojennej* 215.4 (2018), pp. 5–28. DOI: 10.2478/sjpna-2018-0024.

[Fin13] Fink, A. *Conducting research literature reviews: from the Internet to paper*. Sage Publications, 2013.

[FEP+19] Fiore, S. et al. "An Integrated Big and Fast Data Analytics Platform for Smart Urban Transportation Management." In: *IEEE Access* 7 (2019). DOI: 10.1109/ACCESS.2019.2936941.

[Fre18] Fredriksson, C. "Big data creating new knowledge as support in decision-making: practical examples of big data use and consequences of using big data as decision support." In: *Journal of Decision Systems* 27.1 (2018), pp. 1–18. DOI: 10.1080/12460125.2018.1459068.

[FHL14] Fuchs, M., Höpken, W., and Lexhagen, M. "Big data analytics for knowledge generation in tourism destinations – A case from Sweden." In: *Journal of Destination Marketing & Management* 3.4 (2014), pp. 198–209. DOI: 10.1016/j.jdmm.2014.08.002.

[Gaj19] Gajdošík, T. "Big Data Analytics in Smart Tourism Destinations. A New Tool for Destination Management Organizations?" In: *Smart Tourism as a Driver for Culture and Sustainability*. Ed. by Katsoni, V. and Segarra-Oña, M. Springer Proceedings in Business and Economics. Cham: Springer International Publishing, 2019, pp. 15–33. DOI: 10.1007/978-3-030-03910-3_2.

[GH15] Gandomi, A. and Haider, M. "Beyond the hype: Big data concepts, methods, and analytics." In: *International Journal of Information Management* 35.2 (2015), pp. 137–144. DOI: 10.1016/j.ijinfomgt.2014.10.007.

[GGM+02] Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., and Schneider, L. "Sweetening ontologies with DOLCE." In: *International Conference on Knowledge Engineering and Knowledge Management*. 2002, pp. 166–181.

[GSS+16] Gani, A., Siddiqa, A., Shamshirband, S., and Hanum, F. "A survey on indexing techniques for big data: taxonomy and performance evaluation." In: *Knowledge and Information Systems* 46.2 (2016), pp. 241–284. DOI: 10.1007/s10115-015-0830-y.

[GAR+18] Gardiner, A., Aasheim, C., Rutner, P., and Williams, S. "Skill Requirements in Big Data: A Content Analysis of Job Advertisements." In: *Journal of Computer Information Systems* 58.4 (2018), pp. 374–384. DOI: 10.1080/08874417.2017.1289354.

[Gar11] Gartner. *Hype Cycle for Emerging Technologies, 2011*. 2011. URL: https://www.gartner.com/en/documents/1754719/hype-cycle-for-emerging-technologies-2011 (visited on 11/25/2020).

[Gar12] Gartner. *Gartner's 2012 Hype Cycle for Emerging Technologies Identifies*. 2012. URL: https://www.gartner.com/en/documents/2100915 (visited on 07/23/2022).

[Gar15] Gartner. *Gartner's 2015 Hype Cycle for Emerging Technologies Identifies the Computing Innovations That Organizations Should Monitor*. Ed. by Gartner. 2015.

[Gar22a] Gartner. *Definition of Big Data - Gartner Information Technology Glossary*. 5.03.2022. URL: https://www.gartner.com/en/information-technology/glossary/big-data (visited on 03/05/2022).

[Gar22b] Gartner. *Gartner Hype Cycle Research Methodology*. 15.03.2022. URL: https://www.gartner.com/en/research/methodologies/gartner-hype-cycle (visited on 03/15/2022).

[Gar22c] Gartner. *Hype Cycle for Emerging Technologies, 2014*. 11.05.2022. URL: https://www.gartner.com/en/documents/2809728 (visited on 05/11/2022).

[Gee13] Geerdink, B. “A Reference Architecture for Big Data Solutions: Introducing a model to perform predictive analytics using big data technology.” In: *ICITST (International Conference for Internet Technology and Secured Transactions)* 8 (2013), pp. 71–76.

[GG17] Ghantous, G. B. and Gill, A. “DevOps: Concepts, Practices, Tools, Benefits and Challenges.” In: *PACIS 2017 Proceedings* (2017). URL: https://aisel.aisnet.org/pacis2017/96.

[Gha21a] Ghasemaghaei, M. “Understanding the impact of big data on firm performance: The necessity of conceptually differentiating among big data characteristics.” In: *International Journal of Information Management* 57.C (2021).

[Gha21b] Ghavami, P. *Big data management: Data governance principles for big data analytics*. Berlin: De Gruyter, 2021. DOI: 10.1515/9783110664065.

[GGL03] Ghemawat, S., Gobioff, H., and Leung, S.-T. “The Google File System.” In: *ACM SOSP’03*. Bolton Landing, New York, USA, 2003.

[Git22a] GitHub. *GitHub - pixelastic/pokemonorbigdata: Is it Pokemon or Big Data ?* 6.03.2022. URL: https://github.com/pixelastic/pokemonorbigdata (visited on 03/06/2022).

[Git22b] GitHub. *ONT-API Wiki*. 22.05.2022. URL: https://github.com/avicomp/ont-api/wiki (visited on 05/22/2022).

[Goe15] Goes, P. B. “Big Data - Analytics Engine for Digital Transformation: Where is IS?” In: *AMCIS (Americas Conference on Information Systems)* (2015).

[GHK+18] Gohar, M., Hassan, S. A., Khan, M., Guizani, N., Ahmed, A., Rahman, and Arif Ur. “A Big Data Analytics Architecture for the Internet of Small Things.” In: *IEEE Mag.* 56.2 (2018), pp. 128–133. DOI: 10.1109/MCOM.2018.1700273.

[GCA15] Gölzer, P., Cato, P., and Amberg, M. “Data Processing Requirements of Industry 4.0 - Use Cases for Big Data Applications.” In: *ECIS (Conference: European Conference on Information Systems)* 23 (2015).

[Góm01] Gómez-Pérez, A. “Evaluation of ontologies.” In: *International Journal of Intelligent Systems* 16.3 (2001), pp. 391–409. DOI: 10.1002/1098-111X(200103).

[GRS17] Gong, Y., Rimba, P., and Sinnott, R. “A Big Data Architecture for Near Real-time Traffic Analytics.” In: *Companion Proceedings of the10th International Conference on Utility and Cloud Computing - UCC ’17 Companion*. Ed. by Anjum, A., Sill, A., Fox, G., and Chen, Y. New York, New York, USA: ACM Press, 2017, pp. 157–162. DOI: 10.1145/3147234.3151010.

[GS71] Gorry, G. A. and Scott Morton, M. S. *A framework for management information systems.* [Cambridge, M.I.T.], 1971.

[Gra16] Grady, N. W. "KDD meets Big Data." In: *2016 IEEE International Conference on Big Data.* Ed. by Joshi, J. Piscataway, NJ: IEEE, 2016, pp. 1603–1608. DOI: 10.1109/BigData.2016.7840770.

[Gru93] Gruber, T. R. "A translation approach to portable ontology specifications." In: *Knowledge Acquisition* 5.2 (1993), pp. 199–220. DOI: 10.1006/knac.1993.1008.

[GMG+17] Gu, F., Ma, B., Guo, J., Summers, P. A., and Hall, P. "Internet of things and Big Data as potential solutions to the problems in waste electrical and electronic equipment management: An exploratory study." In: *Waste management (New York, N.Y.)* 68 (2017), pp. 434–448. DOI: 10.1016/j.wasman.2017.07.037.

[Gua98] Guarino, N. "Formal ontology and information systems." In: *Proceedings Formal Ontology in Information Systems.* Ed. by IOS Press. Proceedings of FOIS. Trento: IOS Press, 1998, pp. 3–15.

[GOS09] Guarino, N., Oberle, D., and Staab, S. "What Is an Ontology?" In: *Handbook on Ontologies.* Ed. by Staab, S. and Studer, R. Springer, Berlin, Heidelberg, 2009, pp. 1–17. DOI: 10.1007/978-3-540-92673-3_0.

[GAC+20] Guerrero-Prado, J. S., Alfonso-Morales, W., Caicedo-Bravo, E., Zayas-Pérez, B., and Espinosa-Reza, A. "The power of big data and data analytics for AMI data: A case study." In: *Sensors (Switzerland)* 20.11 (2020), pp. 1–27. DOI: 10.3390/s20113289.

[GTT+16] Guerriero, M., Tajfar, S., Tamburri, D. A., and Di Nitto, E. "Towards a model-driven design tool for big data architectures." In: *BIGDSE (International Workshop on Big Data Software Engineering)* 2 (2016), pp. 37–43. DOI: 10.1145/2896825.2896835.

[GH13] Günzel, F. and Holm, A. B. "One size does not fit all : understanding the front-end and back-end of business model innovation." In: *International journal of innovation management* 17.1 (2013).

[HWF+19] Haberfellner, R., Weck, O. L. de, Fricke, E., and Vössner, S. *Systems engineering: Fundamentals and applications.* Cham, Switzerland: Birkhäuser, 2019.

[Hac20] HackerRank. "HackerRank's 2020 Developer Skills Report." In: (2020). URL: https://info.hackerrank.com/rs/487-WAY-049/images/HackerRank-2020-Developer-Skills-Report.pdf (visited on 05/21/2022).

[HPK20] Han, J., Park, S., and Kim, J. "Dynamic OverCloud: Realizing Microservices-Based IoT-Cloud Service Composition over Multiple Clouds." In: *Electronics* 9.6 (2020), p. 969. DOI: 10.3390/electronics9060969.

[HMD17] Haroun, A., Mostefaoui, A., and Dessables, F. "A Big Data Architecture for Automotive Applications: PSA Group Deployment Experience." In: *CCGRID (International Symposium on Cluster, Cloud and Grid Computing)* 17 (2017), pp. 921–928. DOI: 10.1109/CCGRID.2017.107.

[HYA+15] Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., and Ullah Khan, S. "The rise of "big data" on cloud computing: Review and open research issues." In: *Information Systems* 47 (2015), pp. 98–115. DOI: 10.1016/j.is.2014.07.006.

[HIE+18] Hassan, M., I. El Desouky, A., Elghamrawy, S., and Sarhan, A. "Intelligent hybrid remote patient-monitoring model with cloud-based framework for knowledge discovery." In: *Computers & Electrical Engineering* 70 (2018), pp. 1034–1048. DOI: 10.1016/j.compeleceng.2018.02.032.

[Hau02] Haupt, M. ""Data is the New Oil" — A Ludicrous Proposition - Project 2030 - Medium." In: *Project 2030* (2016-05-02). URL: https://medium.com/project-2030/data-is-the-new-oil-a-ludicrous-proposition-1d91bba4f294 (visited on 03/05/2022).

[HZB+21] Heidari, M., Zad, S., Berlin, B., and Rafatirad, S. "Ontology Creation Model based on Attention Mechanism for a Specific Business Domain." In: *IEMTRONICS International Conference, Toronto, Canada*. Ed. by Chakrabarti, S. Piscataway, NJ: IEEE, 2021, pp. 1–5. DOI: 10.1109/IEMTRONICS52119.2021.9422664.

[HEE21] Helmy, S. E., Eladl, G. H., and Eisa, M. "Fuzzy Analytical Hierarchy Process (FAHP) using geometric mean method to select best processing framework adequate to big data." In: *Journal of Theoretical and Applied Information Technology*. 2021.

[HRA16] Herrnansyah, Ruldeviyani, Y., and Aji, R. F. "Enhancing query performance of library information systems using NoSQL DBMS: Case study on library information systems of Universitas Indonesia." In: *2016 International Workshop on Big Data and Information Security (IWBIS)*. Piscataway, NJ: IEEE, 2016, pp. 41–46. DOI: 10.1109/IWBIS.2016.7872887.

[HMP+04] Hevner, A. R., March, S. T., Park, J., and Ram, S. "Design Science in Information Systems Research." In: *MIS Quarterly* 28 (2004), pp. 75–105.

[HAH18] Higgins, J., Al-Jody, T., and Holmes, V. *Rapid Deployment of Bare-Metal and In-Container HPC Clusters Using OpenHPC playbooks*. 2018.

[Hoc15] Hochstein, L. *Ansible: Up & running : automating configuration management and deployment the easy way*. First edition. Sebastopol, CA: O'Reilly & Associates Inc, 2015.

[Hor19] Horridge, M. *OWLViz - Protege Wiki*. 29.03.2019. URL: https://protegewiki.stanford.edu/wiki/OWLViz (visited on 07/29/2019).

[Hot21] Hotz, N. *Why Big Data Science & Data Analytics Projects Fail*. 2021. URL: https://www.datascience-pm.com/project-failures/ (visited on 03/06/2022).

[HA05] Hove, S. E. and Anda, B. "Experiences from Conducting Semi-structured Interviews in Empirical Software Engineering Research." In: *11th IEEE International Software Metrics Symposium*. Los Alamitos, Calif: IEEE Computer Society, 2005, p. 23. DOI: 10.1109/METRICS.2005.24.

[HHR+17] Hruschka, S., Herrero, V., Romero, O., Abelló, A., Franch, X., Vansummeren, S., and Valerio, D. "A software reference architecture for semantic-aware Big Data systems." In: *Information and Software Technology* 90 (2017), pp. 75–92. DOI: 10.1016/j.infsof.2017.06.001.

[HCH17] Hsu, H.-H., Chang, C.-Y., and Hsu, C.-H., eds. *Big data analytics for sensor-network collected intelligence*. Intelligent data-centric systems. London United Kingdom: Academic Press an imprint of Elsevier, 2017.

[HWC+14] Hu, H., Wen, Y., Chua, T.-S., and Li, X. "Toward Scalable Systems for Big Data Analytics: A Technology Tutorial." In: *IEEE Access* 2 (2014), pp. 652–687. DOI: 10.1109/ACCESS.2014.2332453.

[HCJ+15] Huang, Q., Cervone, G., Jing, D., and Chang, C. "DisasterMapper." In: *Proceedings of the 4th International ACM SIGSPATIAL Workshop on Analytics for Big Geospatial Data*. Ed. by Chandola, V. ACM, 2015, pp. 1–6. DOI: 10.1145/2835185.2835189.

[HVN16] Huber, M. F., Voigt, M., and Ngomo, A.-C. N. "Big data architecture for the semantic analysis of complex events in manufacturing." In: *Informatik 2016*. Ed. by Mayr, H. C. and Pinzger, M. Bonn: Gesellschaft für Informatik e.V, 2016, pp. 353–360.

[Hus20] Hussein, A. A. "Fifty-Six Big Data V's Characteristics and Proposed Strategies to Overcome Security and Privacy Challenges (BD2)." In: *Journal of Information Security* 11.04 (2020), pp. 304–328. DOI: 10.4236/jis.2020.114019. URL: https://www.scirp.org/journal/paperinformation.aspx?paperid=103823.

[Hwa81] Hwang, C.-L. *Multiple Attribute Decision Making: Methods and Applications A State-of-the-Art Survey*. Vol. 186. Springer eBook Collection. Berlin and Heidelberg: Springer, 1981. DOI: 10.1007/978-3-642-48318-9.

[IBM13] IBM. "Analytics: The real-world use of big data." In: (2013). URL: https://www.ibmbigdatahub.com/whitepaper/analytics-real-world-use-big-data (visited on 02/12/2020).

[IBM16] IBM. *Analytics Solutions Unified Method*. 2016. URL: http://i2t.icesi.edu.co/ASUM-DM_External/index.htm#cognos.external.asum-DM_Teaser/deliveryprocesses/ASUM-DM_8A5C87D5.html (visited on 02/03/2020).

[IEE12] IEEE. *IEEE Standard for System and Software Verification and Validation*. Piscataway, NJ, USA, 2012. DOI: 10.1109/IEEESTD.2012.6204026.

[IFC20] Iglesias, C. A., Favenza, A., and Carrera, Á. "A Big Data Reference Architecture for Emergency Management." In: *Information* 11.12 (2020), p. 569. DOI: 10.3390/info11120569. URL: https://www.mdpi.com/914052.

[IRH+20] Ijadi Maghsoodi, A., Riahi, D., Herrera-Viedma, E., and Zavadskas, E. K. "An integrated parallel big data decision support tool using the W-CLUS-MCDA: A multi-scenario personnel assessment." In: *Knowledge-Based Systems* 195 (2020), p. 105749. DOI: 10.1016/j.knosys.2020.105749.

[IPO15] Immonen, A., Paakkonen, P., and Ovaska, E. "Evaluating the Quality of Social Media Data in Big Data Architecture." In: *IEEE Access* 3 (2015), pp. 2028–2043. DOI: 10.1109/ACCESS.2015.2490723.

[INC15] INCOSE. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. 4th ed. New York: Wiley, 2015.

[INC20] INCOSE. *Systems Engineering*. 2020. URL: https://www.incose.org/systems-engineering (visited on 01/16/2020).

[IMM+13] Iqbal, R., Murad, M. A. A., Mustapha, A., and Sharef, N. M. "An Analysis of Ontology Engineering Methodologies: A Literature Review." In: *Research Journal of Applied Sciences, Engineering and Technology* 6.16 (2013), pp. 2993–3000. DOI: 10.19026/rjaset.6.3684.

[IZ19] Isah, H. and Zulkernine, F. "A Scalable and Robust Framework for Data Stream Ingestion." In: *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018* (2019), pp. 2900–2905. DOI: 10.1109/BigData.2018.8622360.

[ISO11] ISO. *ISO/IEC/IEEE 29148: 2011(E): ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes –Requirements engineering*. S.l.: IEEE, 2011.

[ISO14] ISO. "ISO/IEC JTC 1 - Big Data - Report." In: (2014). URL: https://www.iso.org/files/live/sites/isoorg/files/developing_standards/docs/en/big_data_report-jtc1.pdf (visited on 03/17/2022).

[ISO] ISO. *42010-2011 ISO/IEC/IEEE Systems and software engineering – Architecture description*. Piscataway, NJ, USA. DOI: 10.1109/IEEESTD.2011.6129467.

[ISO17] ISO. *ISO/IEC 25010:2011 - Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models*. 1.01.2017. URL: http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=35733 (visited on 03/14/2022).

[IS16] Istephan, S. and Siadat, M.-R. "Unstructured medical image query using big data - An epilepsy case study." In: *Journal of biomedical informatics* 59 (2016), pp. 218–226. DOI: 10.1016/j.jbi.2015.12.005.

[IAG+15] Izadi, D., Abawajy, J. H., Ghanavati, S., and Herawan, T. "A data fusion method in wireless sensor networks." In: *Sensors (Basel, Switzerland)* 15.2 (2015), pp. 2964–2979. DOI: 10.3390/s150202964.

[JNL19] Javornik, M., Nadoh, N., and Lange, D. "Data Is the New Oil." In: *Towards User-Centric Transport in Europe*. Ed. by Müller, B. Lecture Notes in Mobility Ser. Cham: Springer, 2019, pp. 295–308. DOI: 10.1007/978-3-319-99756-8_19.

[JJL20] Jin, H. Y., Jung, E. S., and Lee, D. "High-performance IoT streaming data prediction system using Spark: a case study of air pollution." In: *Neural Computing and Applications* 32.17 (2020), pp. 13147–13154. DOI: 10.1007/s00521-019-04678-9.

[JPA+20] Jnr, A. B., Petersen, S. A., Ahlers, D., and Krogstie, J. "Big Data Driven Multi-Tier Architecture for Electric Mobility as a Service in Smart Cities: A Design Science Approach." In: (2020), pp. 1–25.

[JBB14] Jovic, A., Brkic, K., and Bogunovic, N. "An overview of free software tools for general data mining." In: *Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014 37th International Convention on*. IEEE, 2014. DOI: 10.1109/mipro.2014.6859735.

[KPJ+16] Kallio, H., Pietilä, A.-M., Johnson, M., and Kangasniemi, M. "Systematic methodological review: developing a framework for a qualitative semi-structured interview guide." In: *Journal of advanced nursing* 72.12 (2016), pp. 2954–2965. DOI: 10.1111/jan.13031.

[KKP17] Kamilaris, A., Kartakoullis, A., and Prenafeta-Boldú, F. X. "A review on the practice of big data analysis in agriculture." In: *Computers and Electronics in Agriculture* 143 (2017), pp. 23–37. DOI: 10.1016/j.compag.2017.09.037.

[Kan11] Kantardzic, M. *Data mining: Concepts, models, methods, and algorithms*. 2. ed. Piscataway, NJ and Hoboken, NJ: IEEE Press and Wiley, 2011.

[KAK16] Kapil, G., Agrawal, A., and Khan, R. A. "A study of big data characteristics." In: *2016 International Conference on Communication and Electronics Systems (ICCES)*. 2016, pp. 1–4. DOI: 10.1109/CESYS.2016.7889917.

[KL14] Kashlev, A. and Lu, S. "A System Architecture for Running Big Data Workflows in the Cloud." In: *IEEE International Conference on Services Computing* (2014), pp. 51–58. DOI: 10.1109/SCC.2014.16.

[KWG13] Katal, A., Wazid, M., and Goudar, R. H. "Big data: Issues, challenges, tools and Good practices." In: *Sixth International Conference on Contemporary Computing*. Ed. by Parashar. IEEE, 2013, pp. 404–409. DOI: 10.1109/IC3.2013.6612229.

[KDn22] KDnuggets. *The 42 V's of Big Data and Data Science - KDnuggets*. 16.03.2022. URL: https://www.kdnuggets.com/2017/04/42-vs-big-data-data-science.html (visited on 03/16/2022).

[KE17] Kenyon, D. and Eloff, J. "Big data science for predicting insurance claims fraud." In: *2017 Information Security for South Africa*. Ed. by Venter, H. S., Loock, M., and Coetzee, M. Piscataway, NJ: IEEE, 2017, pp. 40–47. DOI: 10.1109/ISSA.2017.8251773.

[KKA20] Ketu, S., Kumar Mishra, P., and Agarwal, S. "Performance Analysis of Distributed Computing Frameworks for Big Data Analytics: Hadoop Vs Spark." In: *Computación y Sistemas* 24.2 (2020). DOI: 10.13053/cys-24-2-3401.

[Kha17] Khalid, F. *Inovalation is driving healthcare transformation with pre-engineered infrastructure and big data analytics*. Bowie, Maryland, 2017. (Visited on 06/01/2020).

[KES+16] Khalifa, S., Elshater, Y., Sundaravarathan, K., Bhat, A., Martin, P., Imam, F., Rope, D., Mcroberts, M., and Statchuk, C. "The Six Pillars for Building Big Data Analytics Ecosystems." In: *ACM Computing Surveys* 49.2 (2016), pp. 1–36. DOI: 10.1145/2963143.

[KAS+18] Khan, N., Alsaqer, M., Shah, H., Badsha, G., Ahmad Abbasi, A., and Salehian, S. "The 10 Vs, Issues and Challenges of Big Data." In: *ICBDE (International Conference on Big Data and Education)* (2018), pp. 52–56. DOI: 10.1145/3206157.3206166.

[KNH+19] Khan, N., Naim, A., Hussain, M. R., Naveed, Q. N., Ahmad, N., and Qamar, S. "The 51 V's Of Big Data." In: *Proceedings of the International Conference on Omni-Layer Intelligent Systems*. ACM Digital Library. New York,NY,United States: Association for Computing Machinery, 2019, pp. 19–24. DOI: 10.1145/3312614.3312623.

[KZV+16] Khazaei, H., Zareian, S., Veleda, R., and Litoiu, M. "Sipresk: A Big Data Analytic Platform for Smart Transportation." In: *First EAI International Summit, Smart City 360°*. Ed. by Leon-Garcia, A. et al. Vol. 166. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Cham and s.l.: Springer, 2016, pp. 419–430. DOI: 10.1007/978-3-319-33681-7_35.

[KMM+15] Kiran, M., Murphy, P., Monga, I., Dugan, J., and Singh Baveja, S. "Lambda Architecture for Cost-effective Batch and Speed Big Data processing." In: *Big Data (IEEE International Conference on Big Data)* (2015), pp. 2785–2792.

[KBB+16] Klein, J., Buglak, R., Blockow, D., Wuttke, T., and Cooper, B. "A reference architecture for big data systems in the national security domain." In: *BIGDSE (International Workshop on Big Data Software Engineering)* 2 (2016), pp. 51–57. DOI: 10.1145/2896825.2896834.

[KND+16] Koley, S., Nandy, S., Dutta, P., Dhar, S., and Sur, T. "Big Data Architecture with Mobile Cloud in CDroid Operating System for Storing Huge Data." In: *CAST* (2016), pp. 12–17.

[KML+16] Kontopoulos, E., Martinopoulos, G., Lazarou, D., and Bassiliades, N. "An ontology-based decision support tool for optimizing domestic solar hot water system selection." In: *Journal of Cleaner Production* 112 (2016), pp. 4636–4646. DOI: 10.1016/j.jclepro.2015.08.088.

[KBP18] Koren, O., Binyaminov, M., and Perel, N. "The Impact of Distributed Data in Big Data Platforms on Organizations." In: *FTC (Proceedings of the Future Technologies Conference)* (2018), pp. 1024–1036. DOI: 10.1007/978-3-030-02683-7_76.

[KNZ18] Kozmina, N., Niedrite, L., and Zemnickis, J. "Information Requirements for Big Data Projects: A Review of State-of-the-Art Approaches." In: *Databases and Information Systems*. Ed. by Lupeikiene, A., Vasilecas, O., and Dzemyda, G. Vol. 838. Communications in computer and information science. Cham: Springer International Publishing, 2018, pp. 73–89. DOI: 10.1007/978-3-319-97571-9_8.

[Kre14] Kreps, J. *Questioning the Lambda Architecture: The Lambda Architecture has its merits, but alternatives are worth exploring.* 2014. URL: https://www.oreilly.com/ideas/questioning-the-lambda-architecture (visited on 01/21/2020).

[Kru16] Krumwiede, K. *2016 Building a Team to Capitalize the Promise of Big Data*. Ed. by Institute of Management Account. 2016. URL: http://www.

imanet.org/-/media/0727a1f482ab42a49b3e948fa7a31771.ashx (visited on 11/26/2020).

[LAM+17] L. Carnevale, A. Celesti, M. Fazio, P. Bramanti, and M. Villari. "How to enable clinical workflows to integrate big healthcare data." In: *2017 IEEE Symposium on Computers and Communications (ISCC)*. 2017, pp. 857–862. DOI: 10.1109/ISCC.2017.8024634.

[Lan01a] Laney, D. "3D data management: Controlling data volume, velocity and variety." In: *META group research note* 6.70 (2001), p. 1.

[Lan01b] Laney, D. *3D Data Management: Controlling Data Volume, Velocity, and Variety*. Ed. by Gartner. 2001.

[Lan12] Laney, D. *Information Economics, Big Data and the Art of the Possible with Analytics*. 2012. URL: https://www-950.ibm.com/events/wwe/grp/grp037.nsf/vLookupPDFs/Gartner_Doug-%20Analytics/$file/Gartner_Doug-%20Analytics.pdf (visited on 02/19/2016).

[LPB16] Le Dinh, T., Phan, T.-C., and Bui, T. "Towards an Architecture for Big Data-Driven Knowledge Management Systems." In: *SIGODIS (Intelligence And Intelligent Systems)* (2016), pp. 1–10.

[Lea10] Leavitt, N. "Will NoSQL Databases Live Up to Their Promise?" In: *Computer* 43.2 (2010), pp. 12–14. DOI: 10.1109/MC.2010.58.

[Lee17] Lee, I. "Big data: Dimensions, evolution, impacts, and challenges." In: *Business Horizons* 60.3 (2017), pp. 293–303. DOI: 10.1016/j.bushor.2017.01.004.

[LFV16] Lehmann, D., Fekete, D., and Vossen, G. *Technology selection for big data and analytical applications: Working Papers, ERCIS - European Research Center for Information Systems*. Münster, 2016.

[LJ06] Levy, Y. and J. Ellis, T. "A Systems Approach to Conduct an Effective Literature Review in Support of Information Systems Research." In: *Informing Science: The International Journal of an Emerging Transdiscipline* 9 (2006), pp. 181–212. DOI: 10.28945/479.

[LXT+18] Li, J., Xu, L., Tang, L., Wang, S., and Li, L. "Big data in tourism research: A literature review." In: *Tourism Management* 68 (2018), pp. 301–323. DOI: 10.1016/j.tourman.2018.03.009.

[LTO16] Li, Y., Thomas, M. A., and Osei-Bryson, K.-M. "A snail shell process model for knowledge discovery via data analytics." In: *Decision Support Systems* 91 (2016), pp. 1–12. DOI: 10.1016/j.dss.2016.07.003.

[LRT+26] Lin, D.-y., Rayavarapu, S. N., Tadjeddine, K., and Yeoh, R. "Beyond financials: Helping small and medium-size enterprises thrive." In: *McKinsey & Company* (2022-01-26). URL: https://www.mckinsey.com/industries/public-and-social-sector/our-insights/beyond-financials-helping-small-and-medium-size-enterprises-thrive (visited on 03/06/2022).

[Liu18] Liu, D. "Big Data Analytics Architecture for Internet-of-Vehicles Based on the Spark." In: *ICITBS (International Conference on Intelligent Transportation, Big Data & Smart City)* (2018), pp. 13–16. DOI: 10.1109/ICITBS.2018.00011.

[LDW+10] Liu, S., Duffy, A. H. B., Whitfield, R. I., and Boyle, I. M. "Integration of decision support systems to improve decision support performance." In: *Knowledge and Information Systems* 22.3 (2010), pp. 261–286. DOI: 10.1007/s10115-009-0192-4. URL: https://link.springer.com/article/10.1007/s10115-009-0192-4.

[Lně15] Lněnička, M. "AHP Model for the Big Data Analytics Platform Selection." In: *Acta Informatica Pragensia* 4.2 (2015), pp. 108–121. DOI: 10.18267/j.aip.64.

[LLV+11] Loos, P. et al. "In-memory Databases in Business Information Systems." In: *Business & Information Systems Engineering* 3.6 (2011), pp. 389–395. DOI: 10.1007/s12599-011-0188-y.

[LX19] Lu, Y. and Xu, X. "Cloud-based manufacturing equipment and big data analytics to enable on-demand manufacturing services." In: *Robotics and Computer-Integrated Manufacturing* 57 (2019), pp. 92–102. DOI: 10.1016/j.rcim.2018.11.006.

[LDK+14] Lv, Y., Duan, Y., Kang, W., Li, Z., and Wang, F.-Y. "Traffic Flow Prediction With Big Data: A Deep Learning Approach." In: *IEEE Transactions on Intelligent Transportation Systems* (2014), pp. 1–9. DOI: 10.1109/TITS.2014.2345663.

[MBB+20] Macak, M., Bangui, H., Buhnova, B., Molnár, A., and Sidló, C. "Big Data Processing Tools Navigation Diagram." In: *Proceedings of the 5th International Conference on Internet of Things, Big Data and Security*. SCITEPRESS - Science and Technology Publications, 2020, pp. 304–312. DOI: 10.5220/0009406403040312.

[MMK15] Madhavji, N. H., Miranskyy, A., and Kontogiannis, K. "Big Picture of Big Data Software Engineering: With Example Research Challenges." In: *2015 IEEE/ACM 1st International Workshop on Big Data Software Engineering*. IEEE, 2015, pp. 11–14. DOI: 10.1109/BIGDSE.2015.10.

[MB18] Majumdar, A. and Bose, I. "Detection of financial rumors using big data analytics: the case of the Bombay Stock Exchange." In: *Journal of Organizational Computing and Electronic Commerce* 28.2 (2018), pp. 79–97. DOI: 10.1080/10919392.2018.1444337.

[MBL+14] Malone, J., Brown, A., Lister, A. L., Ison, J., Hull, D., Parkinson, H., and Stevens, R. "The Software Ontology (SWO): a resource for reproducibility in biomedical data analysis, curation and digital preservation." In: *Journal of Biomedical Semantics* 5.1 (2014), p. 25. DOI: 10.1186/2041-1480-5-25. URL: https://jbiomedsem.biomedcentral.com/articles/10.1186/2041-1480-5-25.

[MCB+11] Manyika, J., Chui, M., Brown, B., Bughin Jacques, Dobbs, R., Roxburgh, C., Byers, and Angela Hung. *Big Data: The next frontier for innovation, competition, and productivity.* Ed. by McKinsey. 2011. URL: https://www.mckinsey.com/~/media/McKinsey/Business%20Functions/McKinsey%20Digital/Our%20Insights/Big%20data%20The%20next%20frontier%20for%20innovation/MGI_big_data_exec_summary.pdf (visited on 10/26/2020).

[Mar24] Marr, B. "Data Is The New Oil: How Shell Has Become A Data-Driven And AI-Enabled Business." In: *Forbes* (2020-04-24). URL: https://www.forbes.com/sites/bernardmarr/2020/04/24/data-is-the-new-oil-how-shell-has-become-a-data-driven-and-ai-enabled-business/?sh=448f2d013170 (visited on 03/07/2022).

[MLR17] Martinez-Mosquera, D., Lujan-Mora, S., and Recalde, H. "Conceptual Modeling of Big Data Extract Processes with UML." In: *INCISCOS (International Conference on Information Systems and Computer Science)* (2017), pp. 207–211. DOI: 10.1109/INCISCOS.2017.18.

[MCA+15] Martínez-Prieto, M. A., Cuesta, C. E., Arias, M., and Fernández, J. D. "The Solid architecture for real-time management of big semantic data." In: *Future Generation Computer Systems* 47 (2015), pp. 62–79. DOI: 10.1016/j.future.2014.10.016.

[MW15] Marz, N. and Warren, J. *Big data: Principles and best practices of scalable real-time data systems.* Shelter Island, NY: Manning, 2015.

[MC13] Mayer-Schönberger, V. and Cukier, K. *Big data: A revolution that will transform how we live, work, and think.* Boston: Houghton Mifflin Harcourt, 2013.

[May10] Mayring, P. "Qualitative Inhaltsanalyse." In: *Handbuch Qualitative Forschung in der Psychologie.* Ed. by Mey, G. and Mruck, K. Wiesbaden: VS Verlag für Sozialwissenschaften, 2010, pp. 601–613. DOI: 10.1007/978-3-531-92052-8_42.

[MHK+15] McCarthy, M. A., Herger, L. M., Khan, S. M., and Belgodere, B. M. "Composable DevOps: Automated Ontology Based DevOps Maturity Analysis." In: *2015 IEEE International Conference on Services Computing (SCC 2015)*. Ed. by Maglio, P. P. Piscataway, NJ: IEEE, 2015, pp. 600–607. DOI: 10.1109/SCC.2015.87.

[MT00] Medvidovic, N. and Taylor, R. N. "A classification and comparison framework for software architecture description languages." In: *IEEE Transactions on Software Engineering* 26.1 (2000), pp. 70–93. DOI: 10.1109/32.825767.

[MK14] Meir-Huber, M. and Köhler, M. *Best Practice für Big Data Projekte - Leitfaden aus #Big Data in #Austria*. 2014. DOI: 10.13140/2.1.3500.8001.

[AA19] Al-Mekhlal, M. and Ali Khwaja, A. "A Synthesis of Big Data Definition and Characteristics." In: *2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*. IEEE, 2019. DOI: 10.1109/cse/euc.2019.00067.

[MSF+21] Mendoza-Moreno, J. F., Santamaria-Granados, L., Fraga Vázquez, A., and Ramirez-Gonzalez, G. "OntoTouTra: Tourist Traceability Ontology Based on Big Data Analytics." In: *Applied Sciences* 11.22 (2021), p. 11061. DOI: 10.3390/app112211061. URL: https://www.mdpi.com/2076-3417/11/22/11061.

[Mer14] Merkel, D. "Docker: lightweight Linux containers for consistent development and deployment." In: *Linux Journal* 239 (2014).

[Mik95] Mike Uschold, M. K. *Towards a Methodology for Building Ontologies*. 1995.

[MKP+21] Mishra, B. K., Kumar, V., Panda, S. K., and Tiwari, P. *Handbook of Research for Big Data: Concepts and Techniques*. Milton: Apple Academic Press Incorporated, 2021.

[Mis17] Mistrík, I., ed. *Software architecture for big data and the cloud*. Cambridge MA: MK an imprint of Elsevier, 2017.

[MK15] Mobus, G. E. and Kalton, M. C. *Principles of Systems Science*. Understanding Complex Systems. New York, NY: Springer, 2015. DOI: 10.1007/978-1-4939-1920-8.

[Mon22] MongoDB. *MongoDB: Die Plattform Für Anwendungsdaten*. 21.05.2022. URL: https://www.mongodb.com/de-de (visited on 05/21/2022).

[MKK15] Morabito, R., Kjallman, J., and Komu, M. "Hypervisors vs. Lightweight Virtualization: A Performance Comparison." In: *Proceedings*. Piscataway, NJ: IEEE, 2015, pp. 386–393. DOI: 10.1109/IC2E.2015.74.

[MWH+16] Morente-Molinera, J. A., Wikström, R., Herrera-Viedma, E., and Carlsson, C. "A linguistic mobile Decision Support System based on fuzzy ontology to facilitate knowledge mobilization." In: *Decision Support Systems* 81 (2016), pp. 66–75. DOI: 10.1016/j.dss.2015.09.001.

[MVM16] Mourtzis, D., Vlachou, E., and Milas, N. "Industrial Big Data as a Result of IoT Adoption in Manufacturing." In: *Procedia CIRP* 55 (2016), pp. 290–295. DOI: 10.1016/j.procir.2016.07.038. URL: http://www.sciencedirect.com/science/article/pii/S2212827116307880.

[MSD+16] Mousannif, H., Sabah, H., Douiji, Y., and Oulad Sayad, Y. "Big data projects: just jump right in!" In: *International Journal of Pervasive Computing and Communications* 12.2 (2016), pp. 260–288. DOI: 10.1108/IJPCC-04-2016-0023.

[MSD+14] Mousannif, H., Sabah, H., Douiji, Y., and Sayad, Y. O. "From Big Data to Big Projects: A Step-by-Step Roadmap." In: *2014 2nd International Conference on Future Internet of Things and Cloud (FiCloud)*. 2014, pp. 373–378. DOI: 10.1109/FiCloud.2014.66.

[Mro18] Mrozek, D. "Foundations of the Hadoop Ecosystem." In: *Scalable Big Data Analytics for Protein Bioinformatics: Efficient Computational Solutions for Protein Structures*. Cham: Springer International Publishing, 2018, pp. 137–150. DOI: 10.1007/978-3-319-98839-9_6.

[MFV18] Müller, O., Fay, M., and Vom Brocke, J. "The Effect of Big Data and Analytics on Firm Performance: An Econometric Analysis Considering Industry Characteristics." In: *Journal of Management Information Systems* 35.2 (2018), pp. 488–509. DOI: 10.1080/07421222.2018.1451955.

[MCS14] Munar, A., Chiner, E., and Sales, I. "A Big Data Financial Information Management Architecture for Global Banking." In: *FiCloud (International Conference on Future Internet of Things and Cloud)* 2 (2014), pp. 385–388. DOI: 10.1109/FiCloud.2014.68.

[MS18] Munir, K. and Sheraz Anjum, M. "The use of ontologies for effective knowledge modelling and information retrieval." In: *Applied Computing and Informatics* 14.2 (2018), pp. 116–126. DOI: 10.1016/j.aci.2017.07.003.

[Mus15] Musen, M. A. "The Protégé Project: A Look Back and a Look Forward." In: *AI matters* 1.4 (2015), pp. 4–12. DOI: 10.1145/2757001.2757003.

[MBR+19] Muthuramalingam, S., Bharathi, A., Rakesh kumar, S., Gayathri, N., Sathiyaraj, R., and Balamurugan, B. "IoT Based Intelligent Transportation System (IoT-ITS) for Global Perspective: A Case Study." In: *Internet of Things and Big Data Analytics for Smart Generation*. Ed. by Balas, V. E., Solanki,

V. K., Kumar, R., and Khari, M. Cham: Springer, 2019, pp. 279–300. DOI: 10.1007/978-3-030-04203-5_13.

[NHR+17] Nadal, S., Herrero, V., Romero, O., Abelló, A., Franch, X., Vansummeren, S., and Valerio, D. "A software reference architecture for semantic-aware Big Data systems." In: *Information and Software Technology* 90 (2017), pp. 75–92. DOI: 10.1016/j.infsof.2017.06.001.

[NMM+16] Nadareishvili, I., Mitra, R., McLarty, M., and Amundsen, M. *Microservice architecture: Aligning principles, practices, and culture.* First edition. Beijing et al.: O'Reilly Media, 2016.

[NKK18] Narain Singh, K., Kumar Behera, R., and Kumar Mantri, J. "Big Data Ecosystem: Review on Architectural Evolution." In: *IEMIS (Emerging Technologies in Data Mining and Information Security)* 2 (2018), pp. 335–345. DOI: 10.1007/978-981-13-1498-8_30.

[NVB+13] Neuhaus, F. et al. "Towards Ontology Evaluation Across the Life Cycle: The Ontology Summit 2013." In: *Appl. Ontology* 8.3 (2013), pp. 179–194. URL: http://dl.acm.org/citation.cfm?id=2594763.2594765.

[NS21a] Nicholas, J. M. and Steyn, H., eds. *Project Management for Engineering, Business, and Technology.* Sixth Edition. Boston: Butterworth-Heinemann, 2021.

[NS21b] Nicholas, J. M. and Steyn, H. "Systems Approach and Systems Engineering." In: *Project Management for Engineering, Business, and Technology.* Ed. by Nicholas, J. M. and Steyn, H. Boston: Butterworth-Heinemann, 2021, pp. 46–82. DOI: 10.1016/B978-0-08-096704-2.50011-9.

[NVM13] Nickerson, R. C., Varshney, U., and Muntermann, J. "A method for taxonomy development and its application in information systems." In: *European Journal of Information Systems* 22.3 (2013), pp. 336–359. DOI: 10.1057/ejis.2012.26.

[Nie11] Nielsen, F. Å. "A new ANEW: Evaluation of a word list for sentiment analysis in microblogs." In: *MSM (Workshop on 'Making Sense of Microposts')* (2011), pp. 47–51.

[NSB+16] Nino, M., Saenz, F., Blanco, J. M., and Illarramendi, A. "Requirements for a big data capturing and integration architecture in a distributed manufacturing scenario." In: *2016 IEEE 14th International Conference on Industrial Informatics (INDIN).* IEEE, 2016, pp. 1326–1329. DOI: 10.1109/INDIN.2016.7819372.

[NAM16] Noorwali, I., Arruda, D., and Madhavji, N. H. "Understanding quality requirements in the context of big data systems." In: *Proceedings of the 2nd International Workshop on BIG Data Software Engineering - BIGDSE '16*. Ed. by Unknown. New York, New York, USA: ACM Press, 2016, pp. 76–79. DOI: 10.1145/2896825.2896838.

[NM01] Noy, N. F. and McGuinness, D. L. *Ontology development 101: A guide to creating your first ontology*. 2001.

[NCP90] Nunamaker, J. F., Chen, M., and Purdin, T. D. "Systems Development in Information Systems Research." In: *Journal of Management Information Systems* 7.3 (1990), pp. 89–106. DOI: 10.1080/07421222.1990.11517898.

[NYC18] NYC OpenData. *2017 Green Taxi Trip Data*. 2018. URL: https://data.cityofnewyork.us/Transportation/2017-Green-Taxi-Trip-Data/5gj9-2kzx (visited on 05/15/2022).

[OMG14] OMG. *Business Process Model and Notation Specification Version 2.0.2*. 2014. URL: https://www.omg.org/spec/BPMN/2.0.2/PDF (visited on 04/06/2022).

[OMG17] OMG. "Unified Modeling Language, v2.5.1." In: (2017), pp. 1–796. (Visited on 08/27/2019).

[OBO+17] Onal, A. C., Berat Sezer, O., Ozbayoglu, M., and Dogdu, E. "Weather data analysis and sensor fault detection using an extended IoT framework with semantics, big data, and machine learning." In: *2017 IEEE International Conference on Big Data*. Ed. by Nie, J.-Y., Obradovic, Z., Suzumura, T., Ghosh, R., Nambiar, R., and Wang, C. Piscataway, NJ: IEEE, 2017, pp. 2037–2046. DOI: 10.1109/BigData.2017.8258150.

[OBA+17] Oussous, A., Benjelloun, F.-Z., Ait Lahcen, A., and Belfkih, S. "Big Data technologies: A survey." In: *Journal of King Saud University - Computer and Information Sciences* (2017). DOI: 10.1016/j.jksuci.2017.06.001.

[PSL08] P. Panov, S. Džeroski, and L. Soldatova. "OntoDM: An Ontology of Data Mining." In: *2008 IEEE International Conference on Data Mining Workshops*. 2008, pp. 752–760. DOI: 10.1109/ICDMW.2008.62.

[PP15] Pääkkönen, P. and Pakkala, D. "Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems." In: *Big Data Research* 2.4 (2015), pp. 166–186. DOI: 10.1016/j.bdr.2015.01.001.

[PC18] Palanivel, K. and Chithralekha, T. "Big Data Reference Architecture for e-Learning Analytical Systems." In: *Int. J. Recent Innov. Trends Comput. Commun* 6.1 (2018).

[PTJ+14] Panahiazar, M., Taslimitehrani, V., Jadhav, A., and Pathak, J. "Empowering Personalized Medicine with Big Data and Semantic Web Technology: Promises, Challenges, and Use Cases." In: *IEEE International Conference on Big Data (Big Data), 2014*. Ed. by Lin, J. Vol. 2015. Piscataway, NJ: IEEE, 2014, pp. 790–795. DOI: 10.1109/BigData.2014.7004307.

[PSK17] Panimalar, A., Shree, V., and Kathrine, V. "The 17 V's Of Big Data." In: *International Research Journal of Engineering and Technology (IRJET)* 9 (2017). URL: https://www.irjet.net/archives/V4/i9/IRJET-V4I957.pdf (visited on 03/16/2022).

[PLH15] Panneerselvam, J., Liu, L., and Hill, R. "Requirements and Challenges for Big Data Architectures." In: *Application of Big Data for National Security*. Elsevier, 2015, pp. 131–139. DOI: 10.1016/B978-0-12-801967-2.00009-4.

[PSD13] Panov, P., Soldatova, L., and Džeroski, S. "OntoDM-KDD: Ontology for Representing the Knowledge Discovery Process." In: Springer, Berlin, Heidelberg, 2013, pp. 126–140. DOI: 10.1007/978-3-642-40897-7_9.

[PP18a] Papathanasiou, J. and Ploskas, N. "AHP." In: *Multiple Criteria Decision Aid*. Ed. by Papathanasiou, J. Vol. 136. SpringerLink Bücher. Cham: Springer International Publishing, 2018, pp. 109–129. DOI: 10.1007/978-3-319-91648-4_5.

[PP18b] Papathanasiou, J. and Ploskas, N. "PROMETHEE." In: *Multiple Criteria Decision Aid*. Ed. by Papathanasiou, J. Vol. 136. SpringerLink Bücher. Cham: Springer International Publishing, 2018, pp. 57–89. DOI: 10.1007/978-3-319-91648-4_3.

[PP18c] Papathanasiou, J. and Ploskas, N. "TOPSIS." In: *Multiple Criteria Decision Aid*. Ed. by Papathanasiou, J. Vol. 136. SpringerLink Bücher. Cham: Springer International Publishing, 2018, pp. 1–30. DOI: 10.1007/978-3-319-91648-4_1.

[PLB] Passlick, J., Lebek, B., and Breitner, M. H. "A Self-Service Supporting Business Intelligence and Big Data Analytics Architecture." In: *Wirtschaftsinformatik 2017*, pp. 1126–1140.

[PA16] Patti, E. and Acquaviva, A. "IoT platform for Smart Cities: Requirements and implementation case studies." In: *Proceedings of 2nd International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*. Piscataway, NJ: IEEE, 2016, pp. 1–6. DOI: 10.1109/RTSI.2016.7740618.

[Pau95] Paul J.H. Schoemaker. "Scenario Planning: A Tool for Strategic Thinking." In: *MIT Sloan Management Review* (1995). URL: https://sloanreview.mit.edu/article/scenario-planning-a-tool-for-strategic-thinking/.

[PB16] Pavlikov, R. and Beisembekova, R. “Architecture and security tools in distributed information systems with Big Data.” In: *AICT (International Conference on Application of Information and Communication Technologies)* 10 (2016).

[PNL02] Pease, A., Niles, I., and Li, J. “The suggested upper merged ontology: A large ontology for the semantic web and its applications.” In: *Working notes of the AAAI-2002 workshop on ontologies and the semantic web*. Vol. 28. 2002, pp. 7–10.

[PRT+12] Peffers, K., Rothenberger, M., Tuunanen, T., and Vaezi, R. “Design Science Research Evaluation.” In: *DESRIST (International Conference on Design Science Research in Information Systems)* (2012), pp. 398–410.

[PTR+07] Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. “A Design Science Research Methodology for Information Systems Research.” In: *Journal of Management Information Systems* 24.3 (2007), pp. 45–77. DOI: 10.2753/MIS0742-1222240302.

[PPP+18] Persico, V., Pescapé, A., Picariello, A., and Sperlí, G. “Benchmarking big data architectures for social networks data processing using public cloud platforms.” In: *Future Generation Computer Systems* 89 (2018), pp. 98–109. DOI: 10.1016/j.future.2018.05.068.

[PZ14a] Philip Chen, C. L. and Zhang, C.-Y. “Data-intensive applications, challenges, techniques and technologies: A survey on Big Data.” In: *Information Sciences* 275 (2014), pp. 314–347. DOI: 10.1016/j.ins.2014.01.015.

[PZ14b] Philip Chen, C. L. and Zhang, C.-Y. “Data-intensive applications, challenges, techniques and technologies: A survey on Big Data.” In: *Information Sciences* 275 (2014), pp. 314–347. DOI: 10.1016/j.ins.2014.01.015.

[PGS+16] Pirttikangas, S., Gilman, E., Su, X., Leppanen, T., Keskinarkaus, A., Rautiainen, M., Pyykkonen, M., and Riekki, J. “Experiences with smart city traffic pilot.” In: *Proceedings of 2016 IEEE International Conference on Big Data*. Ed. by Joshi, J. Piscataway, NJ: IEEE, 2016, pp. 1346–1352. DOI: 10.1109/BigData.2016.7840741.

[Pla12] Plattner, H. *In-Memory Data Management: Technology and Applications*. 2nd ed. Berlin, Heidelberg: Springer Berlin / Heidelberg, 2012.

[PMI08] PMI. *A guide to the project management body of knowledge (PMBOK guide)*. 4th ed. Newtown Square Pa.: Project Management Institute Inc, 2008.

[PR21] Pohl, K. and Rupp, C. *Basiswissen Requirements Engineering: Aus- und Weiterbildung nach IREB-Standard zum Certified Professional for Requirements Engineering Foundation Level*. 5., überarbeitete und aktualisierte Auflage. Heidelberg: dpunkt.verlag, 2021.

[PBT18] Pohl, M., Bosse, S., and Turowski, K. “A Data-Science-as-a-Service Model.” In: *CLOSER 2018*. Ed. by Méndez Muñoz, V., Ferguson, D., Helfert, M., and Pahl, C. Setúbal, Portugal: SCITEPRESS - Science and Technology Publications Lda, 2018, pp. 432–439. (Visited on 03/27/2018).

[PRG+14] Polato, I., Ré, R., Goldman, A., and Kon, F. “A comprehensive view of Hadoop research—A systematic literature review.” In: *Journal of Network and Computer Applications* 46 (2014), pp. 1–25. DOI: 10.1016/j.jnca.2014.07.022.

[PHC17] Poleto, T., Heuer de Carvalho, V. D., and Costa, A. P. C. S. “The Full Knowledge of Big Data in the Integration of Inter-Organizational Information.” In: *International Journal of Decision Support System Technology* 9.1 (2017), pp. 16–31. DOI: 10.4018/IJDSST.2017010102.

[PH15] Poleto, T. and Heuer de Carvalho, Victor DioghoCosta, Ana Paula Cabral Seixas. “The Roles of Big Data in the Decision-Support Process: An Empirical Investigation.” In: *undefined* (2015). URL: https://www.semanticscholar.org/paper/The-Roles-of-Big-Data-in-the-Decision-Support-An-Poleto-Carvalho/b913914acf9b70de6a1f363e54e06514672744dc.

[PLS16] Portela, F., Lima, L., and Santos, M. F. “Why Big Data? Towards a Project Assessment Framework.” In: *Procedia Computer Science* 98 (2016), pp. 604–609. DOI: 10.1016/j.procs.2016.09.094.

[Pow02] Power, D. “Decision Support Systems: Concepts and Resources for Managers.” In: *Faculty Book Gallery* (2002). URL: https://scholarworks.uni.edu/facbook/67.

[Pow08] Power, D. J. “Decision Support Systems: A Historical Overview.” In: *Handbook on Decision Support Systems 1*. Ed. by Holsapple, C. W. SpringerLink Bücher. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 121–140. DOI: 10.1007/978-3-540-48713-5_7.

[PCA14] Prat, N., Comyn-Wattiau, I., and Akoka, J. “Artifact Evaluation in Information Systems Design-Science Research-a Holistic View.” In: *PACIS2014*. 2014.

[PF13] Provost, F. and Fawcett, T. “Data science and its relationship to big data and data-driven decision making.” In: *Big data* 1.1 (2013), pp. 51–59.

[PV17] Ptiček, M. and Vrdoljak, B. “Big Data and New Data Warehousing Approaches.” In: *ICCBDC (International Conference on Cloud and Big Data Computing)* (2017), pp. 6–10. DOI: 10.1145/3141128.3141139.

[RR] Raghupathi, W. and Raghupathi, V. “Big data analytics in healthcare: promise and potential.” In: *Health Information Science and Systems* 2014 (), pp. 1–10. URL: http://www.hissjournal.com/content/2/1/3.

[RA15] Razmak, J. and Aouni, B. “Decision Support System and Multi-Criteria Decision Aid: A State of the Art and Perspectives.” In: *Journal of Multi-Criteria Decision Analysis* 22.1-2 (2015), pp. 101–117. DOI: 10.1002/mcda.1530.

[RGR18] Reinsel, D., Gantz, J., and Rydning, J. *The Digitization of the World - From Edge to Core*. 2018. URL: https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf (visited on 03/05/2020).

[RR08] Rhee, C. and Rao, H. R. “Evaluation of Decision Support Systems.” In: *Handbook on Decision Support Systems 2*. Ed. by Burstein, F. and W. Holsapple, C. International handbooks on information systems. Berlin: Springer, 2008, pp. 313–327. DOI: 10.1007/978-3-540-48716-6_15.

[RFP+18] Rinaldi, S., Flammini, A., Pasetti, M., Tagliabue, L. C., Ciribini, A. L. C., and Zanoni, S. “Metrological issues in the integration of heterogeneous lot devices for energy efficiency in cognitive buildings.” In: *2018 IEEE International Instrumentation & Measurement Technology Conference*. Piscataway, NJ, USA: IEEE, 2018, pp. 1–6. DOI: 10.1109/I2MTC.2018.8409740.

[RVN+19] Roberts, J., Volk, M., Neumann, R., and Turowski, K. “Machine Learning Techniques for Annotations of Large Financial Text Datasets.” In: *AMCIS 2019 Proceedings* (2019). URL: https://aisel.aisnet.org/amcis2019/data_science_analytics_for_decision_support/data_science_analytics_for_decision_support/17.

[RRS+16] Rodríguez-Mazahua, L., Rodríguez-Enríquez, C.-A., Sánchez-Cervantes, J. L., Cervantes, J., García-Alcaraz, J. L., and Alor-Hernández, G. “A general perspective of Big Data: Applications, tools, challenges and trends.” In: *The Journal of Supercomputing* 72.8 (2016), pp. 3073–3113. DOI: 10.1007/s11227-015-1501-1.

[RS12] Rospocher, M. and Serafini, L. “Ontology-centric decision support.” In: CEUR, 2012, pp. 61–72.

[Rus03] Rusche, C. “Digitalisierung: Daten sind das neue Öl.” In: *IWD* (2018-05-03). URL: https://www.iwd.de/artikel/digitalisierung-daten-sind-das-neue-oel-386895/ (visited on 03/07/2022).

[SS18] S. Grandhi and S. Wibowo. “A Multi-criteria Group Decision Making Method for Selecting Big Data Visualization Tools.” In: *undefined* (2018).

[Saa08] Saaty, T. L. “Decision making with the analytic hierarchy process.” In: *International journal of services sciences* 1.1 (2008), pp. 83–98.

[Saa16] Saaty, T. L. "The Analytic Hierarchy and Analytic Network Processes for the Measurement of Intangible Criteria and for Decision-Making." In: *Multiple criteria decision analysis*. Ed. by Greco, S., Ehrgott, M., and Figueira, J. R. Vol. 233. International Series in Operations Research & Management Science. New York et al.: Springer, 2016, pp. 363–419. DOI: 10.1007/978-1-4939-3094-4_10.

[SV13] Saaty, T. L. and Vargas, L. G., eds. *Decision making with the analytic network process: Economic, political, social and technological applications with benefits, opportunities, costs and risks*. 2nd ed. 2013. Vol. 195. International Series in Operations Research & Management Science. Boston, MA: Springer, 2013. DOI: 10.1007/978-1-4614-7279-7.

[SER15] Sabaei, D., Erkoyuncu, J., and Roy, R. "A Review of Multi-criteria Decision Making Methods for Enhanced Maintenance Delivery." In: *Procedia CIRP* 37 (2015), pp. 30–35. DOI: 10.1016/j.procir.2015.08.086. URL: http://www.sciencedirect.com/science/article/pii/S2212827115009403.

[SSK+16] Sachdeva, N., Singh, O., Kapur, P. K., and Galar, D. "Multi-criteria intuitionistic fuzzy group decision analysis with TOPSIS method for selecting appropriate cloud solution to manage big data projects." In: *International Journal of System Assurance Engineering and Management* 7.3 (2016), pp. 316–324. DOI: 10.1007/s13198-016-0455-x.

[SS13] Sagiroglu, S. and Sinanc, D. "Big data: A review." In: *2013 International Conference on Collaboration Technologies and Systems (CTS)*. IEEE, 2013, pp. 42–47. DOI: 10.1109/CTS.2013.6567202.

[SSC17] Saltz, J., Shamshurin, I., and Connors, C. "A Framework for Describing Big Data Projects." In: *Business Information Systems Workshops*. Ed. by Abramowicz, W., Alt, R., and Franczyk, B. Vol. 263. Lecture Notes in Business Information Processing. Cham: Springer, 2017, pp. 183–195. DOI: 10.1007/978-3-319-52464-1_17.

[SS16a] Saltz, J. S. and Shamshurin, I. "Big data team process methodologies: A literature review and the identification of key factors for a project's success." In: *Proceedings of 2016 IEEE International Conference on Big Data*. Ed. by Joshi, J. Piscataway, NJ: IEEE, 2016, pp. 2872–2879. DOI: 10.1109/BigData.2016.7840936.

[SPC+18] Sánchez-Rada, J. F., Pascual, A., Conde, E., and Iglesias, C. A. "A Big Linked Data Toolkit for Social Media Analysis and Visualization Based on W3C Web Components." In: *On the Move to Meaningful Internet Systems*. Ed. by Panetto, H., Debruyne, C., Proper, H. A., Ardagna, C. A., Roman, D., and Meersman, R. Vol. 11230. Programming and Software Engineering.

Cham: Springer International Publishing, 2018, pp. 498–515. DOI: 10.1007/978-3-030-02671-4_30.

[SXV16] Sang, G. M., Xu, L., and Vrieze, P. d. "A Reference Architecture for Big Data Systems." In: *SKIMA (International Conference on Software, Knowledge, Information Management & Applications)* 10 (2016).

[SVS+16] Santos, R. S., Vaz, T. A., Santos, R. P., and Oliveira, J. M. P. de. "Big Data Analytics in a Public General Hospital." In: *Machine learning, optimization, and big data*. Ed. by Pardalos, P. M., Conca, P., Giuffrida, G., and Nicosia, G. Vol. 10122. Lecture Notes in Computer Science. Cham: Springer, 2016, pp. 433–441. DOI: 10.1007/978-3-319-51469-7_38.

[SKL+21] Sayah, Z., Kazar, O., Lejdel, B., Laouid, A., and Ghenabzia, A. "An intelligent system for energy management in smart cities based on big data and ontology." In: *Smart and Sustainable Built Environment* 10.2 (2021), pp. 169–192. DOI: 10.1108/SASBE-07-2019-0087. URL: https://www.emerald.com/insight/content/doi/10.1108/sasbe-07-2019-0087/full/pdf.

[SHB+14] Schermann, M., Hemsen, H., Buchmüller, C., Bitter, T., Krcmar, H., Markl, V., and Hoeren, T. "Big Data." In: *Business & Information Systems Engineering* 6.5 (2014), pp. 261–266. DOI: 10.1007/s12599-014-0345-1.

[SS16b] Schlesner, M. and Schinkel, F. *Big Data Use Case: Genomic Data Research*. Munich, Germany, 2016. URL: https://www.datameer.com/wp-content/uploads/pdf/misc/cs-PF4H-Genome-Research.pdf (visited on 09/04/2019).

[SSB14] Schwarz, C., Schwarz, A., and Black, W. C. "Examining the Impact of Multicollinearity in Discovering Higher-Order Factor Models." In: *Communications of the Association for Information Systems* 34 (2014). DOI: 10.17705/1CAIS.03462.

[SCN+18] Sebaa, A., Chikh, F., Nouicer, A., and Tari, A. "Medical Big Data Warehouse: Architecture and System Design, a Case Study: Improving Healthcare Resources Distribution." In: *Journal of medical systems* 42.4 (2018), p. 59. DOI: 10.1007/s10916-018-0894-9.

[SNC+17] Sebaa, A., Nouicer, A., Chikh, F., and Tari, A. "Big Data Technologies to Improve Medical Data Warehousing." In: *BDCA (international Conference on Big Data, Cloud and Applications)* 2.21 (2017), pp. 1–5. DOI: 10.1145/3090354.3090376.

[SGM18] Sebastio, S., Ghosh, R., and Mukherjee, T. "An Availability Analysis Approach for Deployment Configurations of Containers." In: *IEEE Transactions on Services Computing* (2018), p. 1. DOI: 10.1109/TSC.2017.2788442.

[Sei05] Seiler, D. "How COVID-19 has pushed companies over the technology tipping point—and transformed business forever." In: *McKinsey & Company* (2020-10-05). URL: https://www.mckinsey.com/business-functions/strategy-and-corporate-finance/our-insights/how-covid-19-has-pushed-companies-over-the-technology-tipping-point-and-transformed-business-forever (visited on 03/06/2022).

[Sem04] Semy, Salim K. Pulvermacher, Mary K. Obrst, Leo J. *Toward the use of an upper ontology for US government and US military domains: An evaluation.* 2004.

[SKC16] Seo, W., Kim, N., and Choi, S. "Big Data Framework for Analyzing Patents to Support Strategic R&D Planning." In: ed. by Jin, Q. Piscataway, NJ: IEEE, 2016, pp. 746–753. DOI: 10.1109/DASC-PICom-DataCom-CyberSciTec.2016.131.

[SB16] Seref, B. and Bostanci, E. "Opportunities, Threats and Future Directions in Big Data for Medical Wearables." In: *BDAW (International Conference on Big Data and Advanced Wireless Technologies)* 15 (2016), pp. 1–5. DOI: 10.1145/3010089.3010100.

[SOI15] Sergeevich, K. A., Ovseevna, A. M., and Igor Petrovich, S. "Web-Application For Real-Time Big Data Visualization Of Complex Physical Experiments." In: *SIBCON* (2015).

[SAZ17] Shahin, M., Ali Babar, M., and Zhu, L. "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices." In: *IEEE Access* 5 (2017), pp. 3909–3943. DOI: 10.1109/ACCESS.2017.2685629.

[SDM+18] Shakhovska, N., Duda, O., Matsiuk, O., Bolyubash, Y., and Vovnyanka, R. "Analysis of the Activity of Territorial Communities Using Information Technology of Big Data Based on the Entity-Characteristic Mode." In: *CSIT (International Conference on Computer Science and Information Technologies)* (2018), pp. 155–170. DOI: 10.1007/978-3-030-01069-0_11.

[SDT14] Sharda, R., Delen, D., and Turban, E. *Business intelligence and analytics: Systems for decision support.* 10. ed., global ed. Always learning. Boston, Mass. and Munich: Pearson, 2014.

[Sha16] Sharma, S. "Expanded cloud plumes hiding Big Data ecosystem." In: *Future Generation Computer Systems* 59 (2016), pp. 63–92. DOI: 10.1016/j.future.2016.01.003.

[STG+15] Sharma, S., Tim, U. S., Gadia, S., Wong, J., Shandilya, R., and Peddoju, S. K. "Classification and comparison of NoSQL big data models." In: *International Journal of Big Data Intelligence* 2.3 (2015), p. 201. DOI: 10.1504/IJBDI.2015.070602.

[She00] Shearer, C. "The CRISP-DM Model: The New Blueprint for Data Mining." In: *Journal of Data Warehousing* 5.4 (2000).

[She07] Shenhar, A. J. *Reinventing Project Management: The Diamond Approach To Successful Growth And Innovation*. Boston: Harvard Business Review Press, 2007.

[SK16] Sherimon, P. C. and Krishnan, R. "OntoDiabetic: An Ontology-Based Clinical Decision Support System for Diabetic Patients." In: *Arabian Journal for Science and Engineering* 41.3 (2016), pp. 1145–1160. DOI: 10.1007/s13369-015-1959-4.

[TAG+18] Ta-Shma, P., Akbar, A., Gerson-Golan, G., Hadash, G., Carrez, F., and Moessner, K. "An Ingestion and Analytics Architecture for IoT Applied to Smart City Use Cases." In: *IEEE Internet of Things Journal* 5.2 (2018), pp. 765–774. DOI: 10.1109/JIOT.2017.2722378.

[SIL22] SIL International. *ISO 639 Code Tables — ISO 639-3*. 22.05.2022. URL: https://iso639-3.sil.org/code_tables/639/data/e (visited on 05/22/2022).

[Sim77] Simon, H. A. *The new science of management decision*. Rev. ed. Englewood Cliffs, N.J.: Prentice-Hall, 1977.

[SVP18] Singh, P. K., Verma, R. K., and Prasad, P. E. S. N. Krishna. "IoT-Based Smartbots for Smart City Using MCC and Big Data." In: *SIST (Smart Intelligent Computing and Applications)* (2018), pp. 525–534. DOI: 10.1007/978-981-13-1921-1_52.

[SBF15] Sir, M., Bradac, Z., and Fiedler, P. "Ontology versus Database." In: *IFAC-PapersOnLine* 48.4 (2015), pp. 220–225. DOI: 10.1016/j.ifacol.2015.07.036.

[SPK+15] Siriweera, T., Paik, I., Kumara, B. T., and Koswatta, K. "Intelligent Big Data Analysis Architecture Based on Automatic Service Composition." In: *IEEE International Congress on Big Data* (2015), pp. 276–280. DOI: 10.1109/BigDataCongress.2015.46.

[SPT+17] Smirnov, A., Ponomarev, A., Teslya, N., and Shilov, N. "Human-Computer Cloud for Smart Cities: Tourist Itinerary Planning Case Study." In: *Business Information Systems Workshops*. Ed. by Abramowicz, W. Vol. 303. Lecture Notes in Business Information Processing. Cham: Springer, 2017, pp. 179–190. DOI: 10.1007/978-3-319-69023-0_16.

[Som16] Sommerville, I. *Software engineering*. 10. ed. 2016.

[SGW+15] Song, J., Guo, C., Wang, Z., Zhang, Y., Yu, G., and Pierson, J.-M. "HaoLap: A Hadoop based OLAP system for big data." In: *Journal of Systems and Software* 102 (2015), pp. 167–181. DOI: 10.1016/j.jss.2014.09.024.

[SV12a] Sonnenberg, C. and Vom Brocke, J. "Evaluation Patterns for Design Science Research Artefacts." In: *Practical aspects of design science*. Ed. by Helfert, M. and Donnellan, B. Vol. 286. Communications in computer and information science. Berlin and Heidelberg: Springer, 2012, pp. 71–83. DOI: 10.1007/978-3-642-33681-2_7.

[SV12b] Sonnenberg, C. and Vom Brocke, J. "Evaluations in the Science of the Artificial – Reconsidering the Build-Evaluate Pattern in Design Science Research." In: Springer, Berlin, Heidelberg, 2012, pp. 381–397. DOI: 10.1007/978-3-642-29863-9_28.

[SWF17] Spangenberg, N., Wilke, M., and Franczyk, B. "A Big Data architecture for intra-surgical remaining time predictions." In: *Procedia Computer Science* 113 (2017), pp. 310–317. DOI: 10.1016/j.procs.2017.08.332.

[SPM17] Sriramulu, B., Patil, M., and McGregor, C. "A Cloud Based Big Data Based Online Health Analytics for Rural NICUs and PICUs in India: Opportunities and Challenges." In: *2017 IEEE 30th International Symposium on Computer-Based Medical Systems*. Ed. by Bamidis, P. D., Konstantinidis, S. T., and Pereira Rodrigues, P. Piscataway, NJ: IEEE, 2017, pp. 385–390. DOI: 10.1109/CBMS.2017.112.

[SSS+01] Staab, S., Studer, R., Schnurr, H.-P., and Sure, Y. "Knowledge processes and ontologies." In: *IEEE Intelligent Systems* 16.1 (2001), pp. 26–34. DOI: 10.1109/5254.912382.

[SS09] Staab, S. and Studer, R., eds. *Handbook on Ontologies*. Springer, Berlin, Heidelberg, 2009.

[SHT19] Staegemann, D., Hintsch, J., and Turowski, K. "Testing in Big Data: An Architecture Pattern for a Development Environment for Innovative, Integrated and Robust Applications." In: *Wirtschaftsinformatik 2019 Proceedings* (2019). URL: https://aisel.aisnet.org/wi2019/track03/papers/12.

[SVD+20] Staegemann, D., Volk, M., Daase, C., and Turowski, K. "Discussing Relations Between Dynamic Business Environments and Big Data Analytics." In: *Complex Systems Informatics and Modeling Quarterly* 23 (2020), pp. 58–82. DOI: 10.7250/csimq.2020-23.05.

[SVG+20] Staegemann, D., Volk, M., Grube, A., Hintsch, J., Bosse, S., Häusler, R., Nahhas, A., Pohl, M., and Turowski, K. "Classifying Big Data Taxonomies: A Systematic Literature Review." In: *Proceedings of the 5th International Conference on Internet of Things, Big Data and Security*. SCITEPRESS

- Science and Technology Publications, 2020, pp. 267–278. DOI: 10.5220/0009390102670278.

[SVJ+19] Staegemann, D., Volk, M., Jamous, N., and Turowski, K. "Understanding Issues in Big Data Applications - A Multidimensional Endeavor." In: *Twenty-fifth Americas Conference on Information Systems*. 2019.

[SVJ+20] Staegemann, D., Volk, M., Jamous, N., and Turowski, K. "Exploring the Applicability of Test Driven Development in the Big Data Domain." In: *ACIS 2020 Proceedings* (2020). URL: https://aisel.aisnet.org/acis2020/37.

[SVL+21] Staegemann, D., Volk, M., Lautenschläger, E., Pohl, M., Abdallah, M., and Turowski, K. "Applying Test Driven Development in the Big Data Domain – Lessons From the Literature." In: *Advanced machine learning and deep learning*. Ed. by Jaber, K. M. Piscataway, NJ: IEEE, 2021, pp. 511–516. DOI: 10.1109/ICIT52682.2021.9491728.

[SVN+19a] Staegemann, D., Volk, M., Nahhas, A., Abdallah, M., and Turowski, K. "Exploring the Specificities and Challenges of Testing Big Data Systems." In: *15th International Conference on Signal Image Technology & Internet based Systems, SITIS*. Italy, 2019.

[SVN+19b] Staegemann, D., Volk, M., Nahhas, A., Abdallah, M., and Turowski, K. "Exploring the Specificities and Challenges of Testing Big Data Systems." In: *2019 15th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS)*. IEEE, 11/26/2019 - 11/29/2019, pp. 289–295. DOI: 10.1109/SITIS.2019.00055.

[SVP+21] Staegemann, D., Volk, M., Pohl, M., Häusler, R., Nahhas, A., Abdallah, M., and Turowski, K. "A Preliminary Overview of the Situation in Big Data Testing." In: *Proceedings of the 6th International Conference on Internet of Things, Big Data and Security*. SCITEPRESS - Science and Technology Publications, 4/23/2021 - 4/25/2021, pp. 296–302. DOI: 10.5220/0010475002960302.

[SVS+21] Staegemann, D., Volk, M., Shakir, A., Lautenschläger, E., and Turowski, K. "Examining the Interplay Between Big Data and Microservices – A Bibliometric Review." In: *Complex Systems Informatics and Modeling Quarterly* 27 (2021), pp. 87–118. DOI: 10.7250/csimq.2021-27.04.

[SVT21] Staegemann, D., Volk, M., and Turowski, K. "Quality Assurance in Big Data Engineering - A Metareview." In: *Complex Systems Informatics and Modeling Quarterly* 28 (2021), pp. 1–14. DOI: 10.7250/csimq.2021-28.01.

[Sta19] Stary, C. "Developing Scientific Work Practice in Business Informatics - Digital Support for a Design Science-based Research Project." In: *e-mentor* 82.5 (2019), pp. 4–17. DOI: 10.15219/em82.1439.

[SKK00] Steinbach, M., Karypis, G., and Kumar, V. *A comparison of document clustering techniques*. 2000.

[SZG+15] Strohbach, M., Ziekow, H., Gazis, V., and Akiva, N. "Towards a Big Data Analytics Framework for IoT and Smart City Applications." In: *Modeling and processing for next-generation big-data technologies*. Ed. by Xhafa, F. Vol. 4. Modeling and Optimization in Science and Technologies. Cham: Springer, 2015, pp. 257–282. DOI: 10.1007/978-3-319-09177-8_11.

[SSJ+16] Sun, Y., Song, H., Jara, A. J., and Bie, R. "Internet of Things and Big Data Analytics for Smart and Connected Communities." In: *IEEE Access* 4 (2016), pp. 766–773. DOI: 10.1109/ACCESS.2016.2529723.

[Sut03] Sutcliffe, A. "Scenario-based requirements engineering." In: *Proceedings of 11th IEEE International Requirements Engineering Conference*. Los Alamitos, Calif.: IEEE Computer Society Press, 2003, pp. 320–329. DOI: 10.1109/ICRE.2003.1232776.

[Szc20] Szcepanski, M. "Is data the new oil?" In: (2020). URL: https://www.europarl.europa.eu/RegData/etudes/BRIE/2020/646117/EPRS_BRI(2020)646117_EN.pdf (visited on 03/05/2022).

[SPB16] Szyperski, C., Petitclerc, M., and Barga, R. "Three Experts on Big Data Engineering." In: *IEEE Software* 33.2 (2016), pp. 68–72. DOI: 10.1109/MS.2016.58.

[TZ13] Tamošaitienė, J. and Zavadskas, E. K. "The Multi-stage Decision Making System for Complicated Problems." In: *Procedia - Social and Behavioral Sciences* 82 (2013), pp. 215–219. DOI: 10.1016/j.sbspro.2013.06.248.

[TSL15] Tan, C., Sun, L., and Liu, K. "Big Data Architecture for Pervasive Healthcare: A Literature Review." In: *ECIS (European Conference on Information Systems)* 23 (2015).

[TSK+19] Tan, P.-N., Steinbach, M., Karpatne, A., and Kumar, V. *Introduction to data mining*. Second edition. NY NY: Pearson, 2019.

[TWW+16] Tan, Y., Wang, W., Wu, Q., and Lin, J. "An implementation of heterogeneous architecture based MapReduce in the clouds." In: *Proceedings of 2016 2nd International Conference on Cloud Computing and Internet of Things*. Piscataway, NJ: IEEE, 2016, pp. 16–20. DOI: 10.1109/CCIOT.2016.7868295.

[TG16] Tao, C. and Gao. Jerry. "Quality Assurance for Big Data Application - Issuses, Challenges, and Needs." In: *The 28th International Conference on Software Engineering and Knowledge Engineering, SEKE 2016, Redwood City, San Francisco Bay, USA, July 1-3, 2016*. Ed. by Jerry Gou. KSI Research Inc. and Knowledge Systems Institute Graduate School, 2016, pp. 375–381. DOI: 10.18293/SEKE2016-166.

[Taw20] Tawalbeh, L. "System Architecture." In: *The NICE Cyber Security Framework*. Ed. by Alsmadi. Cham: Springer International Publishing, 2020, pp. 195–206. DOI: 10.1007/978-3-030-41987-5_9.

[TMK12] Thanos, C., Manegold, S., and Kersten, M. "Big Data-introduction to the special theme." In: *ERCIM News* (2012), pp. 10–11. URL: https://pure.uva.nl/ws/files/1617538/159349_380004.pdf.

[TS19] Thippanna, S. and SowmyaNag, K. "A study on Modern Messaging Systems-Kafka, RabbitMQ and NATS Streaming." In: *arXiv* (2019).

[TPK+16] Tihfon, G. M., Park, S., Kim, J., and Kim, Y.-M. "An efficient multi-task PaaS cloud infrastructure based on docker and AWS ECS for application deployment." In: *Cluster Computing* 19.3 (2016), pp. 1585–1597. DOI: 10.1007/s10586-016-0599-0.

[TAL05] Turban, E., Aronson, J. E., and Liang, T.-P. *Decision Support Systems and Intelligen Systems*. 7th ed. 2005.

[Tur21] Turck, M. *Red Hot: The 2021 Machine Learning, AI and Data (MAD) Landscape*. 2021. URL: https://mattturck.com/data2021/ (visited on 03/20/2022).

[Tur22] Turck, M. *The AI and Data Landscape*. 20.03.2022. URL: http://dfkoz.com/ai-data-landscape/ (visited on 03/20/2022).

[Tur03] Turowski, K. *Fachkomponenten: Komponentenbasierte betriebliche Anwendungssysteme*. Magdeburger Schriften zur Wirtschaftsinformatik. Aachen: Shaker, 2003.

[UPA+15] Ullah Rathore, M. M., Paul, A., Ahmad, A., Chen, B.-W., Huang, B., and Ji, W. "Real-Time Big Data Analytical Architecture for Remote Sensing Application." In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 8.10 (2015), pp. 4610–4621. DOI: 10.1109/JSTARS.2015.2424683.

[UG96] Uschold, M. and Gruninger, M. "Ontologies: principles, methods and applications." In: *The Knowledge Engineering Review* 11.2 (1996), pp. 93–136. DOI: 10.1017/S0269888900007797.

[Vaa22] Vaadin. *Vaadin - An open platform for building web apps in Java*. 2022. URL: https://vaadin.com/ (visited on 05/21/2022).

[VK15] Vaishnavi, V. K. and Kuechler, W. *Design science research methods and patterns: Innovating information and communication technology*. Second edition. Boca Raton, FL, London, and New York: CRC Press Taylor & Francis Group, 2015. DOI: 10.1201/b18448.

[VCC+15] Vaquero, L. M., Celorio, A., Cuadrado, F., and Cuevas, R. "Deploying Large-Scale Datasets on-Demand in the Cloud: Treats and Tricks on Data Distribution." In: *IEEE Transactions on Cloud Computing* 3.2 (2015), pp. 132–144. DOI: 10.1109/TCC.2014.2360376.

[VPK+19] Vassakis, K., Petrakis, E., Kopanakis, I., Makridis, J., and Mastorakis, G. "Location-Based Social Network Data for Tourism Destinations." In: *Big Data and Innovation in Tourism, Travel, and Hospitality: Managerial Approaches, Techniques, and Applications*. Ed. by Sigala, M., Rahimi, R., and Thelwall, M. Singapore: Springer Singapore, 2019, pp. 105–114. DOI: 10.1007/978-981-13-6339-9_7.

[VPB16] Venable, J., Pries-Heje, J., and Baskerville, R. "FEDS: a Framework for Evaluation in Design Science Research." In: *European Journal of Information Systems* 25.1 (2016), pp. 77–89. DOI: 10.1057/ejis.2014.36. URL: https://link.springer.com/article/10.1057/ejis.2014.36.

[VS14] Viana, P. and Sato, L. "A Proposal for a Reference Architecture for Long-Term Archiving, Preservation, and Retrieval of Big Data." In: *International Conference on Trust, Security and Privacy in Computing and Communications)* 13 (2014), pp. 622–629. DOI: 10.1109/TrustCom.2014.80.

[VF20] Vitor Afonso Pinto and Fernando Silva Parreiras. "Towards a Taxonomy for Big Data Technological Ecosystem." In: *undefined* (2020). URL: https://www.semanticscholar.org/paper/Towards-a-Taxonomy-for-Big-Data-Technological-Pinto-Parreiras/737265170de8fc279ab622331a786e6d067e85c4.

[VAC+09] Vogel, O., Arnold, I., Chughtai, A., Ihler, E., Kehrer, T., Mehlig, U., and Zdun, U. *Software-Architektur*. Heidelberg: Spektrum Akademischer Verlag, 2009. DOI: 10.1007/978-3-8274-2267-5.

[VBB+19a] Volk, M., Bosse, S., Bischoff, D., and Turowski, K. "Decision-Support for Selecting Big Data Reference Architectures." In: *22nd International Conference, BIS (Business Information Systems)*. Ed. by Abramowicz, W. 2019, pp. 3–17. DOI: 10.1007/978-3-030-20485-3_1.

[VBB+19b] Volk, M., Bosse, S., Bischoff, D., and Turowski, K. "Decision-Support for Selecting Big Data Reference Architectures." In: *BUSINESS INFORMATION SYSTEMS*. Ed. by Abramowicz, W. and Corchuelo, R. Vol. 353. Lecture Notes in Business Information Processing. [S.l.]: Springer, 2019, pp. 3–17. DOI: 10.1007/978-3-030-20485-3_1.

[VBT17] Volk, M., Bosse, S., and Turowski, K. "Providing Clarity on Big Data Technologies: A Structured Literature Review." In: *2017 IEEE 19th Conference on Business Informatics*. Piscataway, NJ: IEEE, 2017, pp. 388–397. DOI: 10.1109/CBI.2017.26.

[VHB+16] Volk, M., Hart, S. W., Bosse, S., and Turowski, K. "How much is Big Data? A Classification Framework for IT Projects and Technologies Diego, CA, USA, August 11-14, 2016." In: *22nd Americas Conference on Information Systems, AMCIS 2016*. AIS, 2016.

[VJT17] Volk, M., Jamous, N., and Turowski, K. "Ask the Right Questions: Requirements Engineering for the Execution of Big Data Projects." In: *Proceedings AMCIS 2017* (2017). URL: https://aisel.aisnet.org/amcis2017/ITProjMgmt/Presentations/4.

[VPT18] Volk, M., Pohl, M., and Turowski, K. "Classifying Big Data Technologies - An Ontology-based Approach." In: *Proceedings AMCIS 2018* (2018). URL: https://aisel.aisnet.org/amcis2018/Semantics/Presentations/4.

[VSJ+17] Volk, M., Shareef, A. E., Jamous, N., and Turowski, K. "New E-Commerce User Interest Patterns." In: *2017 IEEE International Congress on Big Data - BigData Congress 2017*. Ed. by Karypis, G. and Zhang, J. Piscataway, NJ: IEEE, 2017, pp. 406–413. DOI: 10.1109/BigDataCongress.2017.60.

[VSB+20a] Volk, M., Staegemann, D., Bosse, S., Häusler, R., and Turowski, K. "Approaching the (Big) Data Science Engineering Process." In: *Proceedings of the 5th International Conference on Internet of Things, Big Data and Security*. SCITEPRESS - Science and Technology Publications, 2020, pp. 428–435. DOI: 10.5220/0009569804280435.

[VSB+20b] Volk, M., Staegemann, D., Bosse, S., Nahhas, A., and Turowski, K. "Towards a Decision Support System for Big Data Projects." In: *WI2020 Zentrale Tracks*. Ed. by Gronau, N., Heine, M., Poustcchi, K., and Krasnova, H. GITO Verlag, 2020, pp. 357–368. DOI: 10.30844/wi_2020_c10-behnen.

[VSI+22] Volk, M., Staegemann, D., Islam, A., and Turowski, K. "Facing Big Data System Architecture Deployments: Towards an Automated Approach Using Container Technologies for Rapid Prototyping." In: *HICSS2022*. 2022.

[VSJ+20] Volk, M., Staegemann, D., Jamous, N., Pohl, M., and Turowski, K. "Providing Clarity on Big Data Technologies: The BDTOnto Ontology." In: *International Journal of Intelligent Information Technologies* 16.2 (2020), pp. 49–73. DOI: 10.4018/IJIIT.2020040103.

[VSP+19] Volk, M., Staegemann, D., Pohl, M., and Turowski, K. "Challenging Big Data Engineering: Positioning of Current and Future Development." In: *Proceedings of the 4th International Conference on Internet of Things, Big Data and Security*. SCITEPRESS - Science and Technology Publications, 2019, pp. 351–358. DOI: 10.5220/0007748803510358.

[VSP+20] Volk, M., Staegemann, D., Prothmann, F., and Turowski, K. "Towards an Automatized Way for Modeling Big Data System Architectures." In: *Business Information Systems*. Ed. by Abramowicz, W. and Klein, G. Lecture Notes in Business Information Processing. Cham: Springer International Publishing, 2020, pp. 46–60.

[VSS+22] Volk, M., Staegemann, D., Saxena, A., Hintsch, J., and Turowski, K. "Lowering Big Data Project Barriers – Identifying System Architecture Templates for Big Data Standard Use Cases." In: *Proceedings of the 19th International Conference on Smart Business Technologies (ICSBT2022)*. SCITEPRESS, 2022.

[VST+20] Volk, M., Staegemann, D., Trifonova, I., Bosse, S., and Turowski, K. "Identifying Similarities of Big Data Projects – A Use Case Driven Approach." In: *IEEE Access* 8 (2020), p. 1. DOI: 10.1109/ACCESS.2020.3028127.

[VST20] Volk, M., Staegemann, D., and Turowski, K. "Big Data." In: *Handbuch Digitale Wirtschaft*. Ed. by Kollmann, T. Wiesbaden: Springer Fachmedien Wiesbaden, 2020, pp. 1–18. DOI: 10.1007/978-3-658-17345-6_71-1.

[VST21] Volk, M., Staegemann, D., and Turowski, K. "Applying Multi Criteria Decision-Making for the Selection of Big Data Technologies." In: *27th annual Americas Conference on Information Systems (AMCIS)*. 2021.

[VST22] Volk, M., Staegemann, D., and Turowski, K. "Providing Clarity on Big Data – Discussing its Definition and the Most Relevant Data Characteristics (Under Review)." In: *Proceedings of International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*. 2022.

[VM19] Vom Brocke, J. and Maedche, A. "The DSR grid: six core dimensions for effectively planning and communicating design science research projects." In: *Electronic Markets* 29.3 (2019), pp. 379–385. DOI: 10.1007/s12525-019-00358-7. URL: https://link.springer.com/article/10.1007/s12525-019-00358-7.

[W H08] W. Holsapple, C. "DSS Architecture and Types." In: *Handbook on Decision Support Systems 1*. Ed. by Holsapple, C. W. SpringerLink Bücher. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 163–189. DOI: 10.1007/978-3-540-48713-5_9.

[W317] W3. *OWL 2 Web Ontology Language Quick Reference Guide (Second Edition)*. 2.10.2017. URL: https://www.w3.org/TR/2012/REC-owl2-quick-reference-20121211/ (visited on 05/02/2022).

[WWL+17] Wang, H., Wang, Q., Liu, P., and Sun, L. "Big Data and Intelligent Agent based Smart Grid Architecture." In: *ICA (IEEE International Conference on Agents)* (2017), pp. 106–107.

[WKC+14] Wang, Y., Kung, L., Chuang Wang, William Yu, and Cegielski, C. G. "Developing a Big Data-Enabled Transformation Model in Healthcare: A Practice Based View." In: *Proceedings of the 35th International Conference on Information Systems (ICIS)*. 2014.

[WE06] Wang, Y.-M. and Elhag, T. M. "An approach to avoiding rank reversal in AHP." In: *Decision Support Systems* 42.3 (2006), pp. 1474–1480. DOI: 10.1016/j.dss.2005.12.002.

[Wan15] Wang, Y. *Big data algebra: a rigorous approach to big data analytics and engineering*. 2015.

[WB13] Ward, J. S. and Barker, A. "Undefined by data: a survey of big data definitions." In: *arXiv preprint arXiv:1309.5821* (2013).

[WAS+20] Wassouf, W. N., Alkhatib, R., Salloum, K., and Balloul, S. "Predictive analytics using big data for increased customer loyalty: Syriatel Telecom Company case study." In: *Journal of Big Data* 7.1 (2020). DOI: 10.1186/s40537-020-00290-0.

[WR02] Webster, J. and R. Watson. "Analyzing the Past to Prepare for the Future: Writing a Literature Review." In: *undefined* (2002).

[WW02] Webster, J. and Watson, R. T. "Guest Editorial: Analyzing the Past to Prepare for the Future: Writing a literature Review." In: *MIS Quarterly* 26.2 (2002).

[Wei95] Weick, K. E. *Sensemaking in organizations*. [Nachdr.] Foundations for organizational science. Thousand Oaks, Calif.: Sage Publ, 1995.

[WSO14] Weick, K. E., Sutcliffe, K. M., and Obstfeld, D. "Organizing and the Process of Sensemaking." In: *Driving desired futures*. Ed. by Shamiyeh, M. and Ahrweiler, P. Basel: Birkhäuser, 2014, pp. 216–235. DOI: 10.1515/9783038212843.216.

[WB15] Werner Ceusters and Barry Smith. *Aboutness: Towards Foundations for the Information Artifact Ontology*. 2015.

[Wir00] Wirth, R. *CRISP-DM: Towards a standard process model for data mining*. 2000.

[WSS+18] Woo, J., Shin, S.-J., Seo, W., and Meilanitasari, P. "Developing a big data analytics platform for manufacturing systems: architecture, method, and implementation." In: *The International Journal of Advanced Manufacturing Technology* 99.9-12 (2018), pp. 2193–2217. DOI: 10.1007/s00170-018-2416-9.

[Woo15] Woodie, A. *Why Gartner Dropped Big Data Off the Hype Curve*. Ed. by Datanami. 2015. URL: https://www.datanami.com/2015/08/26/why-gartner-dropped-big-data-off-the-hype-curve/ (visited on 11/25/2020).

[WCC+14] Wu, S., Chen, C., Chen, G., Chen, K., Shou, L., Cao, H., and Bai, H. "YZS-tack." In: *Proceedings of the VLDB Endowment* 7.13 (2014), pp. 1778–1783. DOI: 10.14778/2733004.2733083. URL: https://link.springer.com/article/10.1007/s10586-016-0599-0.

[WCK+17] Wu, P.-Y., Cheng, C.-W., Kaddi, C. D., Venugopalan, J., Hoffman, R., and Wang, M. D. "-Omic and Electronic Health Record Big Data Analytics for Precision Medicine." In: *IEEE transactions on bio-medical engineering* 64.2 (2017), pp. 263–273. DOI: 10.1109/TBME.2016.2573285.

[XSQ15] Xu, M., Siraj, S., and Qi, L. "A Hadoop-Based Data Processing Platform for Fresh Agro Products Traceability." In: *Proceedings of the European Conference on Data Mining 2015*. Ed. by Abraham, A. P., Reis, A. P. d., and Roth, J. Computer science and information systems series. Lissabon: IADIS Press, 2015, pp. 37–44.

[YHL+17] Yang, C., Huang, Q., Li, Z., Liu, K., and Hu, F. "Big Data and cloud computing: innovation opportunities and challenges." In: *International Journal of Digital Earth* 10.1 (2017), pp. 13–53. DOI: 10.1080/17538947.2016.1239771.

[YAB+18] Yang, M., Adomavicius, G., Burtch, G., and Ren, Y. "Mind the Gap: Accounting for Measurement Error and Misclassification in Variables Generated via Data Mining." In: *Information Systems Research* 29.1 (2018), pp. 4–24. DOI: 10.1287/isre.2017.0727.

[YMR+17] Yasir Arfat, Muhammad Aqib, Rashid Mehmood, Aiiad Albeshri, Iyad Katib, Nasser Albogami, and Ahmed Alzahrani. "Enabling Smarter Societies through Mobile Big Data Fogs and Clouds." In: *Procedia Computer Science* 109 (2017), pp. 1128–1133. DOI: 10.1016/j.procs.2017.05.439. URL: http://www.sciencedirect.com/science/article/pii/S1877050917311213.

[YSH+18] Yassine, A., Singh, S., Hossain, M. S., and Muhammad, G. "IoT Big Data Analytics for Smart Homes with Fog and Cloud Computing." In: *Future Generation Computer Systems* 91 (2018). DOI: 10.1016/j.future.2018.08.040.

[Yin09] Yin, R. K. *Case study research: Design and methods*. 4th ed. Vol. 5. Applied social research methods series. Los Angeles, Calif: Sage Publications, 2009.

[YP16] Ylijoki, O. and Porras, J. "Perspectives to Definition of Big Data: A Mapping Study and Discussion." In: *Journal of Innovation Management* 4.1 (2016), pp. 69–91. DOI: 10.24840/2183-0606_004.001_0006.

[YCJ+17] Yuan, J., Chen, M., Jiang, T., and Li, T. "Complete tolerance relation based parallel filling for incomplete energy big data." In: *Knowledge-Based Systems* 132 (2017), pp. 215–225. DOI: 10.1016/j.knosys.2017.06.027.

[ZAR+17] Zafar, M. N., Azam, F., Rehman, S., and Anwar, M. W. "A Systematic Review of Big Data Analytics Using Model Driven Engineering." In: *ICCBDC (International Conference on Cloud and Big Data Computing)* (2017), pp. 1–5. DOI: 10.1145/3141128.3141138.

[Zah03] Zaha, J. M. *Komponentenfindung in monolithischen betrieblichen Anwendungssystemen.* 2003.

[ZZW+18] Zhang, Y., Zhang, M., Wo, T., Lin, X., Yang, R., and Xu, J. "A Scalable lnternet-of-Vehicles Service over Joint Clouds." In: *Proceedings of 12th IEEE International Symposium on Service-Oriented System Engineering (SOSE 2018)/9th.* Piscataway, NJ: IEEE, 2018, pp. 210–215. DOI: 10.1109/SOSE.2018.00035.

[ZKF05] Zhao, Y., Karypis, G., and Fayyad, U. "Hierarchical Clustering Algorithms for Document Datasets." In: *Data Mining and Knowledge Discovery* 10.2 (2005), pp. 141–168. DOI: 10.1007/s10618-005-0361-3.

[ZWS+16] Zhuang, Y., Wang, Y., Shao, J., Chen, L., Lu, W., Sun, J., Wei, B., and Wu, J. "D-Ocean: an unstructured data management system for data ocean environment." In: *Frontiers of Computer Science* 10.2 (2016), pp. 353–369. DOI: 10.1007/s11704-015-5045-6.

[ZRI+16] Zicari, R. V., Rosselli, M., Ivanov, T., Korfiatis, N., Tolle, K., Niemann, R., and Reichenbach, C. "Setting Up a Big Data Project: Challenges, Opportunities, Technologies and Optimization." In: *Big Data optimization.* Ed. by Emrouznejad, A. Vol. 18. Studies in Big Data. Cham: Springer, 2016, pp. 17–47. DOI: 10.1007/978-3-319-30265-2_2.

Ehrenerklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; verwendete fremde und eigene Quellen sind als solche kenntlich gemacht. Insbesondere habe ich nicht die Hilfe eines kommerziellen Promotionsberaters in Anspruch genommen. Dritte haben von mir weder unmittelbar noch mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Ich habe insbesondere nicht wissentlich:

- Ergebnisse erfunden oder widersprüchliche Ergebnisse verschwiegen,
- statistische Verfahren absichtlich missbraucht, um Daten in ungerechtfertigter Weise zu interpretieren,
- fremde Ergebnisse oder Veröffentlichungen plagiiert,
- fremde Forschungsergebnisse verzerrt wiedergegeben.

Mir ist bekannt, dass Verstöße gegen das Urheberrecht Unterlassungs- und Schadensersatzansprüche des Urhebers sowie eine strafrechtliche Ahndung durch die Strafverfolgungsbehörden begründen kann. Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht.

Magdeburg, den November 4, 2022

.....
Matthias Volk, M. Sc.