Pacific Asia Journal of the Association for Information Systems
Research Paper doi: 10.17705/1pais.xxxxx Volume xx, Issue x (Year)
ISSN 1943-7544

Big data reference architectures: state of the art

Pouya Ataei, Alan Litchfield

Auckland University of Technology, New Zealand, pouya.ataei@aut.ac.nz
Auckland University of Technology, New Zealand, alan.litchfield@aut.ac.nz

Abstract

Background: Big Data (BD) is a nascent term emerged to describe large amount of data that comes in different forms from various channels. In modern world, users are the ceaseless generators of structured, semi-structured, and unstructured data that if gleaned and crunched precisely, will reveal game-changing patterns. While the opportunities exist with big data, the unprecedented amount of data can bring traditional approaches to a bottleneck, and the growth of data is outpacing technological and scientific advances in data analytics. It is estimated that approximately 75% of the big data projects have failed within the last decade according to multiple sources. Among the challenges, system development and software architecture are prominent.

Method: This paper aims to facilitate big data system development and architecture by conducting a systematic literature review (SLR) on big data reference architectures. The primary goal is to point out major concepts of reference architectures and if they can be helpful for big data system development. The secondary goal is to find all big data reference architecture, describe the challenges of creating these reference architectures, discuss the common architectural components of these reference architectures and the limitations of these reference architectures.

Results: As a result of this work, firstly major concepts about reference architectures are discussed and their applicability to big data system development was depicted. Secondly, 23 big data reference architecture was assessed from academia and practice and their commonalities, challenges, and limitations are identified.

Conclusions: The findings gained emerges the understanding that RAs can be an effective artefact to tackle complex BD system development.

Keywords: Big Data, Big Data Reference Architecture, Big Data Architecture, Reference Architecture, Data analytics, Big Data for business, Data-intensive applications,

1 Introduction

The rapid development of software technologies, the proliferation of digital devices and networking infrastructure of today, have by and large, augmented user's capability to generate data (Rada, Ataeib, Khakbizc, & Akbarzadehd, 2017). In the age of information, users are unceasing generators of structured, semi-structured, and unstructured data that if collected and crunched correctly, may reveal game-changing patterns (Ataei & Litchfield, 2020).

The unprecedented proliferation of data have emerged a new ecosystem of technologies; one of these ecosystems is big data (Mannering, Bhat, Shankar, & Abdel-Aty, 2020; Rad & Ataei, 2017). Big data is a term emerged to describe large amount of data that comes in various forms from different channels. Within the years, big data has attained a lot of attention from academia and industry, and many strive to benefit from this new material (Erevelles, Fukawa, & Swayne, 2016). Howbeit, adopting big data requires the absorption of great deal of complexity and many traditional systems cannot cope with characteristics of this domain.

Based on various reports and surveys published within the last decade, approximately 75% of big data projects have failed (AI, 2019; Gartner, 2014; Manyika et al., 2011; Nash, 2015; Partners, 2019; White, 2019). Among the challenges of adopting big data, the most frequently mentioned are 1) Architectural and system development challenges 2) Rapid technology change challenges and 3) Organizational challenges (Bashari Rad, Akbarzadeh, Ataei, & Khakbiz, 2016; Chen, Kazman, Garbajosa, & Gonzalez, 2017; Singh, Lai, Vejvar, & Cheng, 2019).

Today, majority of big data systems are designed underlying ad-hoc and complicated architectural solutions (Gorton & Klein, 2015; Hummel, Eichelberger, Giloj, Werle, & Schmid, 2018; Nadal et al., 2017). As more companies try to develop their own BD systems, new ad-hoc solutions are created, and new technologies are introduced. This will challenge software architects to design a suitable solution for any given context, creates a foundation for an immature architectural decision, and does not promote the growth and development of BD systems.

Therefore, since the approach of ad-hoc design to big data systems is undesirable and leaves many engineers in the dark, there is a need for more software engineering research for big data systems. To this end, this study presents a systematic literature review (SLR) on big data (BD) reference architectures (RAs). This study is extension of our previous work (Ataei & Litchfield, 2020), which has been limited to a conference paper. Here, we aimed at extending our previous work by including the year 2021 in the SLR, adding two more research question and expanding the architectural components of BD systems further. We have also added a lot of discussion around various reference architectures and went a bit deep on technology selection and challenges that an architect might face.

Conceptualization of the system as a reference architecture, helps with understanding of the system's key components, behavior, composition and evolution of it, which in turn affect quality attributes such as maintainability, scalability and performance (Hilliard). Therefore RAs can be a good standardization artefact and a communication medium that not only results in concrete architectures for big data systems, but also provide stakeholders with unified elements and symbols to discuss and progress big data projects (Galster & Avgeriou, 2011) (Angelov, Grefen, & Greefhorst, 2009).

2 Review Methodology

This research has been designed following the guidelines demonstrated by B. Kitchenham et al. (2009) and Shamseer et al. (2015). B. A. Kitchenham, Dyba, and Jorgensen (2004) framework is used because of its clear instructions on critically appraising evidence for impact, validity and applicability. In addition, to further increase systematicity, transparency and to prevent bias, we used the guidelines provided by Shamseer et al. (2015) on Preferred Reporting Items for Systematic Reviews and Meta-Analysis (PRISMA).

SLR has been chosen because it is a qualitative research methodology that is aimed at driving knowledge and understanding about the subject matter and the elements around it. Besides, SLR provides a transparent and reproducible procedure that elicits patterns, relationships, trends, and delineates the overall picture of the subject (Borrego, Foster, & Froyd, 2014).

The main objective of this study is to assess the current state of BD RAs, identify their major architectural components, point out fundamental concepts and discuss their limitations. This objective is achieved in four phases (figure 1). In first phase, research questions are stated, literature are identified and pooled, and exclusion and inclusion criteria are defined. In second phase, literatures are assessed for their quality based on inclusion/exclusion criteria and relevance to research questions. Thirdly, selected pool of literature is coded based on research questions. Lastly, findings are synthesized, trends and patterns realized and depicted.

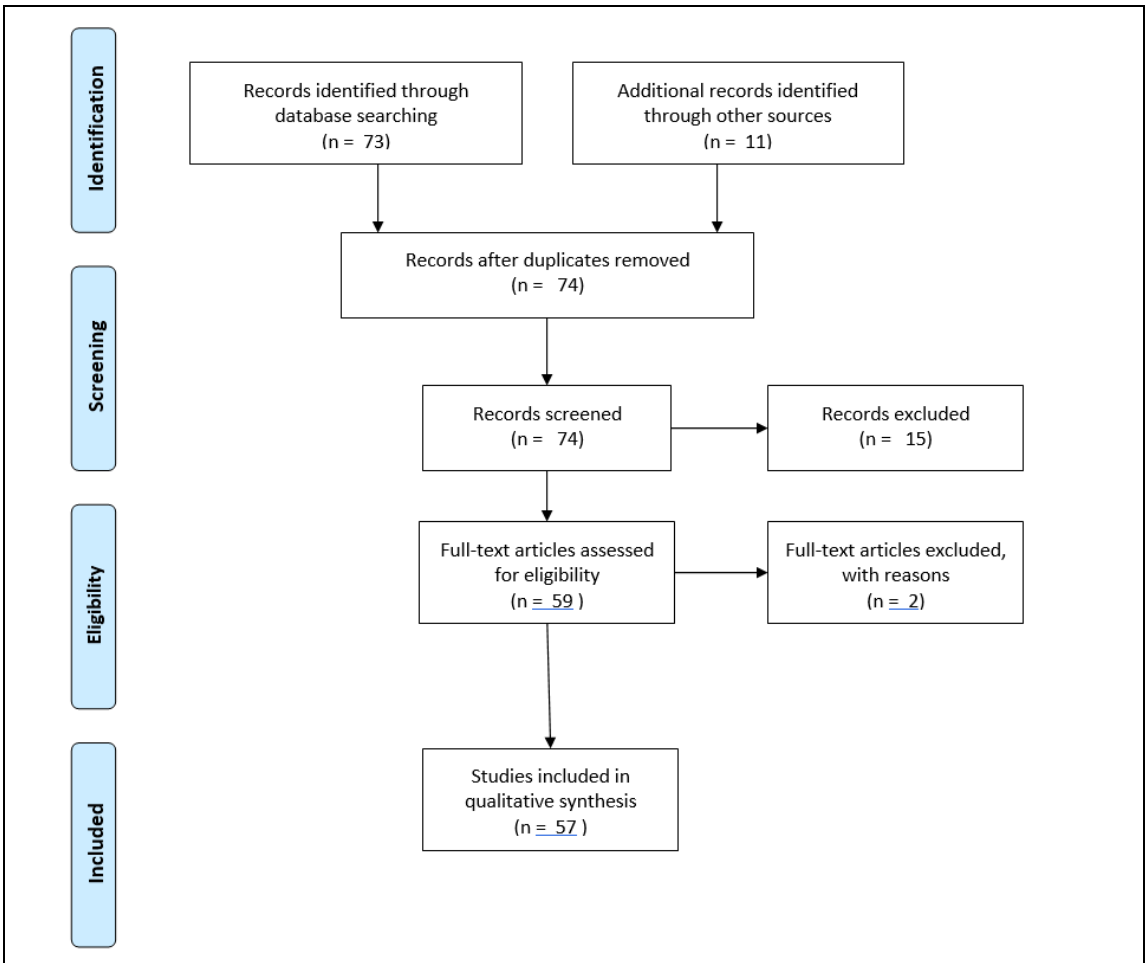

Figure 1 – PRISMA flowchart of the SLR (Moher, Liberati, Tetzlaff, Altman, &

This SLR is based on the following seven research questions:

RQ1. What are the fundamental concepts of RAs?
RQ2. How can RAs help BD system development?
RQ3. What are current BD RAs?
RQ4. What are some common approaches for creating BD RAs?
RQ5. What are the challenges of creating BD RAs?
RQ6. What are major architectural components of BD RAs?
RQ7. What are the limitations of current BD RAs?

2.1. *Identification*

In this phase, we first limited the literature search to the last 11 years, that is 2010-2021. Most literature selected were within the years 2016-2021 as they provided with most recent and relevant information. Howbeit, some studies dating back to 2010 have been included to provide fundamental knowledge regarding big data systems, reference architectures and architecture evaluation.

Databases searched were ScienceDirect, IEEE Explore, SpringerLink, AISeL, Elsevier, MIS Quarterly and ACM library. To pursue to goal of finding all literature available on the topic, and to avoid overlooking valuable research, abstract and citation databases and search engines such as Scopus, Google Scholar, Web of Science, and Research Gate was used.

After the initial search, it becomes apparent that AISeL and Elsevier are good sources for good quality big data literature, whereas MIS Quarterly provided with the highest quality of Information Systems (IS) research.

A combination of short-tail keywords and long-tail keywords based on research question was used. These keywords are as followings:

- *Big Data Reference Architectures*
- *Reference Architectures in the domain of Big Data*
- *Reference Architectures and Big Data*
- *Big data and Reference Architectures*
- *Reference Architectures concepts*
- *The concept of Reference Architectures*
- *Reference Architectures in the domain of Big Data*
- *Big Data specific Reference architectures*

2.2. *Screening and Eligibility*

After initial collection of literature, first we removed duplicates and then screened records for relevancy to our research questions. As a result, 15 studies excluded.

From there on, a full-text assessment took place based on our inclusion and exclusion criteria. These criteria are as following:

Inclusion Criteria:

- The study Includes detailed analysis, preferably in practice
- The study showed considerable case studies that aimed to explore data-business context
- Quantitative or qualitative research that points out industry gaps

- Studies that depict RA concepts
- Research that Indicates the current state of RAs in the field of BD and demonstrates possible outcomes
- Explore BD RAs extensively, and discusses the ecosystem, drivers, and challenges
- Studies that are within the specified time range
- Studies that are scholarly publications, book, book chapter, thesis, dissertation, or conference proceedings
- Grey literature such as white paper that includes extensive information on BD RAs

Exclusion Criteria:

- Studies that are not English
- Very short studies (less than 8 pages)
- Studies that do not aim to explore practice, or discuss practice related concepts
- Studies that provided with low quality information
- Studies that do not directly address the research questions
- Duplicated studies
- Studies that did not explore BD RAs

After excluding papers based on inclusion and exclusion criteria, we assessed studies based on their quality. The quality assessment took place based on the following three factors.

- Is the study rich in terms of its relevant to practice and case studies?
- Does the study provide with ample information/data?
- Is the study discussing contemporary trends in BD RAs?

Regarding the first quality factor, richness is defined as quality and volume of information provided. Primary studies with international focus have been considered to be a good source of information. For example, studies that followed rigorous research methodologies with good pedigrees and aimed to solve a complex problem in an actual industrial setup with a prototype that is extensively evaluated has been considered rich in terms of relevance to practice and cast studies.

In regard to the second quality factor, studies that revolved around either creation of a novel BD RA or exploration and examination of current RAs have been perceive as quality research and have been included in the pool.

Lastly, any study that discussed the recent trends in BD RAs have been added to the pool. A lot of attention has been paid to research methodology and evaluation.

2.3. *Data Collection and Synthesis*

In the first phase (identification) of this SLR, a total of 84 literature has been pooled. Some of this literature has been added to the pool by the process of forward and backward searching. For instance, by reading NIST RA, we found out about Oracle, Facebook, and Amazon RAs and included those in the pool of literature as well.

In the screening phase, the literature that were not in-line with our inclusion and exclusion criteria have been eliminated. For example, if the paper did not either discuss a BD RA, or its ecosystem or limitations, it was excluded. As a result of this phase, 15 papers excluded.

In the next phase, we've applied the quality framework against the remaining literature. The clear criteria set in the framework has helped eliminating bias. In this phase, 2 further study excluded with reason.

By this stage, research questions have been set, inclusion and exclusion criteria are defined and applied, the quality assessment framework is developed and applied to the pool of studies, and the research embarked on actual synthesis of data. The software Nvivo, being primarily developed for qualitative research, was used to label, code, and classify studies.

In Nvivo, we defined 8 nodes for this SLR. These nodes are as followings;

1) big data reference architecture
2) big data reference architecture limitations
3) reference architecture concepts
4) big data challenges
5) big data reference architecture gaps
6) big data RA development
7) big data RA development challenges
8) big data architectural component

Consequently, all the studies were coded and classified based on the defined nodes. After coding all studies, we synthesized this coded information to induce findings and point out patterns. The synthesis took place in-line with the research questions and objectives.

2.4. *SLR Statistics*

By the result of this work, 63 articles have been selected comprising of proceedings, journal articles, book chapters, and white papers. Out of the pool of articles, 33.3% are from IEEE Explore, 5.2% from ScienceDirect, 24.5% from SpringerLink, 15.7% from ACM, and 21% from other sources such as Google Scholar and Research Gate. 30 journal articles, 13 conference proceedings, 12 book chapters, 6 white papers, 1 Master's Thesis and 1 PhD thesis were selected. 51% of the articles were selected from the years 2016- 2020, 33% belonged to years 2013-2016, and the rest to years 2010-2013.

These stats are portrayed inf Figure 2.

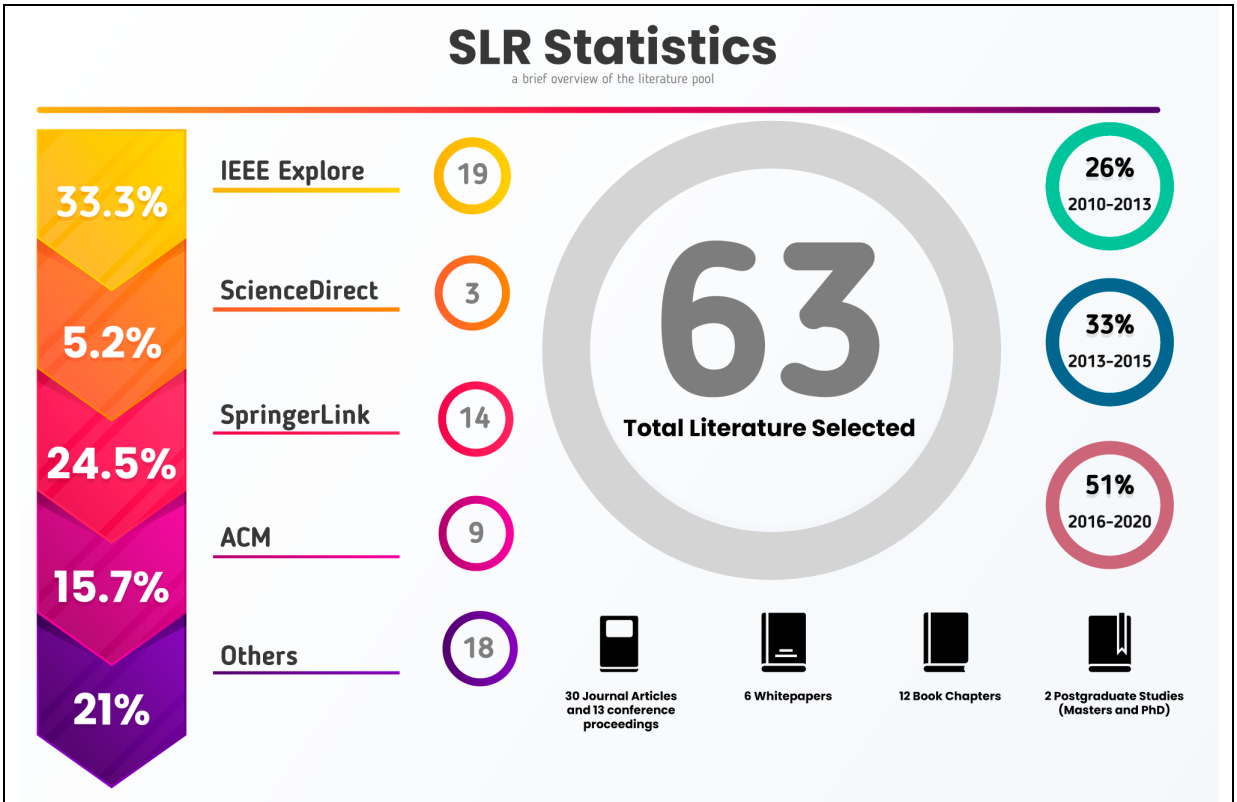

Figure 2 – SLR Statistics

3 Findings

In this section, we map our findings against the research questions in a series of sub-sections. For increased clarity, these sub sections are exact research questions.

3.1. *What are the fundamental concepts of RAs?*

As the complexity of man-made systems grow, procedures, principles, and concepts of software architecture are increasingly applied to address those complexity faced by practitioners (Hilliard). A system abstracted and expressed in terms of architectural concepts, facilitates the understanding of system's essence, properties revolving around it, and evolution of it, which in turn affects quality attributes such as performance, maintainability, and scalability.

In recent years, IT architectures played a pivotal role in the progress and evolution of system development and gained acceptance in maintenance, planning, development, and cost reduction of complex systems (Martínez-Fernández et al. 2014; van Engelenburg et al. 2019).

To address ambiguity about what should be developed to address what needs, an architecture can play an overarching role by portraying the fundamental components of the system and the means and ways in which these components communicate to achieve the overall goal of the system (Sievi-Korte et al. 2019). This in turn creates manageable components that can be used to address different aspect of the problem and provides stakeholders with an abstract artefact to observe, reflect upon, contribute to, and communicate with (Kohler and Specht 2019).

Many successful IT artefacts today stemmed from an effective RA. A few good examples are the Open Systems Interconnection model or OSI (Zimmerman, 1980), Open Authentication

or OATH (OATH), Common Object Request Broker Architecture or CORBA (Pope, 1998),and WMS or workflow management systems (Grefen & de Vries, 1998). In fact, every system goes with an architecture, either known or unknown, and it is in the architecture that the overall qualities of the system are defined.

Whereas there are various definitions to what constitutes an RA, they all share the same principle that the concept of patterns plays a significant role (Cloutier et al., 2010). Reed (2002) defines RA as “a predefined architectural pattern, or set of patterns, possible, partially or completely instantiated, designed, and proven for use in particular business and technical contexts, together with supporting artifacts to enable their use”. In Software Product Line (SPL) development, RAs are defined as generic schema that can be instantiated and configured for a particular class of systems (Derras et al., 2018b).

In software engineering, RAs can be defined as an artefact that transfers software engineering knowledge as a family of solutions to a problem domain (Klein, Buglak, Blockow, Wuttke, & Cooper, 2016). In another terms, RAs are artefacts that embody domain relevant concepts and qualities, break down solutions and a create a ubiquitous language to facilitate effective communication, and inform various stakeholders.

Taking all into consideration, and to answer RQ1, five concepts of RAs is identified; these concepts are as the following;

1. **RAs are at the highest level of abstraction:** RAs aim to capture the essence of the practice as an abstraction that portrays elements necessary for communication, standardization, implementation and maintenance of certain class of systems (Cloutier et al., 2010). Hence, RAs aim to inject software engineering knowledge as a set of high-level architectural patterns and do not provide implementation details such as specific frameworks, vendors or environments. RAs are at higher level of abstraction that concrete architectures.

2. **RAs emphasize heavily on architectural qualities:** RAs, sitting at a higher level of abstractions are artifacts created for a wider audience and a bigger context, and are usually used by solution architects to deduce a concrete architecture in a specific environment (Angelov, Trienekens, & Grefen, 2008; Stricker et al., 2010). As a result, RAs pay more attention to architectural qualities.

3. **In RAs, stakeholders are not clearly defined:** Stakeholders are usually people of the same company involved in the actual design and implementation of the system and do get involved in the product creation in various phases. Different stakeholders have different concerns and are crucial to the creation of the overall product (Geerdink, 2013). A stakeholder can be a developer, a designer, a product owner, a data scientist or a business analyst. Notwithstanding, due to the generic nature of the RAs, it is not feasible to indicate all stakeholders a priori. RAs are at a higher level of abstraction and tend to provide a generic solution for a class of problems, not a specific context. Therefore, defining and introducing stakeholders into RAs can potentially decrease their effectiveness (Ataei & Litchfield, 2020; Chang & Boyd, 2018) .

4. **RAs promote adherence to common standards:** The design of an RA is usually guided by existing architectural patterns based on common pitfalls in practice, the body of literature and various models. For this reason, RAs convey standard approaches and patterns that avoid known pitfall, facilitate reuse, and decrease complexity (Ataei & Litchfield, 2020).

5. **RAs are effective artefacts for system development and communication:** RAs are powerful artefacts that can be used by architects that design, manage, and utilize complex system (Hilliard). Because RAs are created as assets that codify the best practice and conventions of the industry and often include architectural descriptions and standards, they can be deemed effective artefacts for system development and communication.

3.2. *How can RAs help BD system development?*

Despite the high failure rate of BD projects, IT giants such as Google, Facebook or Amazon have developed exclusive BD systems with complicated data pipelines, data management, procurement and batch and real-time analysis capabilities (Kohler & Specht, 2019). Having the resources required, these companies attract the best of talent from around the globe to manage the complexity involved in development of big data systems. Notwithstanding, that's not the reality of majority of organizations that are trying to benefit from big data analytics.

Big data systems sail away from traditional small data analytics paradigms and bring various challenges including rapid technology change challenges (Chen, Kazman, & Haziyev, 2016), system development and architecture challenges (Jagadish et al., 2014) and organizational challenges (Rada et al., 2017; Vassakis, Petrakis, & Kopanakis, 2018). BD does not only mean 'big' amount of data, or just volume; other characteristics of BD such as velocity, variety, veracity and variability bring significant challenges to the practice. Although these challenges do not only belong to domain of BD systems, BD exacerbates these challenges because of the following reasons;

1) Distributed scaling is required to address batch and stream processing demands
2) There is a need for real near-time performance (stream processing)
3) Complex technology orchestration is required to create effective communication channels between components and data flow
4) Continuous delivery is required to continually disseminate patterns and insights into various business domains
5) Two different approaches are required for data processing, stream and batch processing; or fast and delayed processing (Jagadish et al., 2014)

To provide a solution to these challenges, one has to realize the core fundamentals of BD systems. Academic and practitioners of BD, describe BD as an interplay of methodology (workflow, organization), software engineering (data engineering, storage, etc.), and analysis (math, statistics) (Akhtar, Frynas, Mellahi, & Ullah, 2019; Rad & Ataei, 2017).Therefore, one can deduce that technology orchestration is a focal matter in BD system development and maintenance.

Positioned on top of this rationale, and to address RQ2, RAs can be considered an effective artefact that help with component delineation, interface definition, technology orchestration, variability management, scalability, and maintenance of BD systems (Chang & Boyd, 2018; Nadal et al., 2017). The purpose of RAs is to create an integrated environment in which fragmented processes around the system are optimized, responsiveness to change is assured, and delivery of architectural strategies is supported.

Most authors and practitioners agree that issues around BD software engineering and system development are severe and that this justifies the use of RAs for BD systems (Chang & Boyd, 2018). Starting with a grounded RA means that the software architect can refer to an already designed orchestration of components, interfaces, inter-communications, and variability points and map them against the organization's capability framework, desired quality attributes, and business drivers and vision. This also means that the software architecture or the software architecture group is no longer challenged to model a new

architecture from an array of independent components that needs to be assembled through effective interfaces, cache mechanisms, storage, etc.

Taking all into consideration, one can deduce that RAs are artefacts that facilitates development and homogenization of BD systems. Using RA to address complex problems have been successfully applied for Database Management Systems (DBMS) (Piñeiro et al., 2019) and Distributed Database Management Systems (DDBMS) (Rahimi & Haug, 2010).

3.3. *What are current BD RAs?*

As a result of this SLR and to answer RQ3, 23 BD RA has been found, among which 18 RAS are from academia, 4 from practice, and one through the collaboration of academia and practice. These are described further in Table 1.

Table 1 – BD RAs

No.	Reference Architecture	Domain	Year
1	Towards a big Data reference architecture (Maier, Serebrenik, & Vanderfeesten, 2013)	Academia	2013
2	A reference architecture for Big Data solutions introducing a model to perform predictive analytics using Big Data technology (Geerdink 2013)	Academia	2013
3	A proposal for a reference architecture for long-term archiving, preservation, and retrieval of Big Data (Viana & Sato, 2014)	Academia	2014
4	Questioning the Lambda architecture; Kappa Architecture (Kreps, 2014)	Academia	2014
5	Defining architecture components of the Big Data Ecosystem (Demchenko, De Laat, & Membrey, 2014)	Academia	2014
6	Big Data driven e-commerce architecture (Ghandour, 2015) - journal	Academia	2015
7	The solid architecture for real-time management of big semantic data; Solid architecture (Martínez-Prieto, Cuesta, Arias, & Fernández, 2015)	Academia	2015
8	Reference architecture and classification of technologies, products and services for big data systems (Pääkkönen & Pakkala, 2015)	Academia	2015
9	A Reference Architecture for Big Data Systems (Sang, Xu, & De Vrieze, 2016)	Academia	2016
10	A reference architecture for Big Data systems in the national security domain (Klein et al., 2016)	Academia	2016
11	A Reference Architecture for Supporting Secure Big Data Analytics over Cloud-Enabled Relational Databases (Cuzzocrea, 2016)	Academia	2016
12	Managing Cloud-Based Big Data Platforms: A Reference Architecture and Cost Perspective (Heilig & Voß, 2017)	Academia	2017
13	Scalable data store and analytic platform for real-time monitoring of data-intensive scientific infrastructure (Suthakar, 2017)	Academia	2017
14	A software reference architecture for semantic-aware Big Data systems; Bolster Architecture (Nadal et al., 2017)	Academia	2017

15	Towards a secure, distributed, and reliable cloud-based reference architecture for Big Data in smart cities (Kohler & Specht, 2019)	Academia	2019
16	Reference Architectures and Standards for the Internet of Things and Big Data in Smart Manufacturing (Ünal, 2019)	Academia	2019
17	Lambda architecture (Kiran, Murphy, Monga, Dugan, & Baveja, 2015)	Practice	2011
18	IBM - Reference architecture for high performance analytics in healthcare and life science (Quintero & Lee, 2019)	Practice	2013
20	Microsoft - Big Data ecosystem reference architecture (Levin, 2013)	Practice	2013
21	Oracle - Information Management and Big Data: A Reference Architecture (Cackett, 2013)	Practice	2014
22	SAP - NEC Reference Architecture for SAP HANA & Hadoop (2016)	Practice	2016
23	NIST Big Data interoperability framework (Chang & Boyd, 2018)	Hybrid	2018

Within the past years, there has been a considerable attention to the BD domain, and in specific BD system development (Li et al., 2018). For instance, in March 2012, White House announced an initiative for BD research and development (House, 2012). The goal of this initiative was to accelerate the speed of science and engineering discovery, to improve national security, and to improve the knowledge extraction from large and complicated sets of data (Chang & Grady, 2015). This project has been supported by six federal departments and has been given more than $200 million USD with the goal of substantial progress in the tools and techniques to handle big data (Chang & Grady, 2015).

A year later, in June 2013, National Institute of Standards and Technology (NIST) Big Data Public Working Group (NBD-PWG) was launched with considerable participation from across the nation. Practitioners, researchers, agents, government representatives, and none-profit organizations joined in this momentum (Chang & Grady, 2015). One of the results of this project was NIST Big Data Reference Architecture (NBDRA). According to US Department of Defense, one of the main objectives of NBDRA was to provide with an authoritative source of information on big data that restraint and guides the overall practice (Chang & Boyd, 2018). This is arguably one of the most comprehensive and recent RAs available on the fields of big data. NBDRA is made up of two fabrics encompassing five functional logical components connected by various interfaces, representing intertwined nature of security and privacy and management.

Along the lines, other giant IT vendors published their own RAs for big data. In this SLR, 5 BD RA has been collected from the practice, and mostly through white papers. These white papers are from IBM, Microsoft, Oracle, SAP, and a conference in which Lambda was discussed. Among these RAs, arguably Lambda architecture is the most commonly discussed and studied. It is also worth mentioning that there has been other BD RAs found in practice, but they were rather too short or did not reflect the contemporary state of BD analytics and has been eliminated as described in the research methodology section.

In the realm of academia, there has been numerous efforts including a postgraduate master's dissertation (Maier et al., 2013) and PhD thesis (Suthakar, 2017) for creating big data RAs. In addition, few universities have published their own RA. For instance, university of Amsterdam published the BD architecture framework (Chang & Mishra, 2015).

Last but not least, there has been numerous reference architectures developed recently for specific domains. These studies have been usually published as short journal papers, and many have promised future publication of the full reference architecture as a book. For instance, Klein et al. (2016) developed a BD Ra in the national security domain, and Weyrich and Ebert (2015) worked on a BD RA in the domain of internet of things (IOT).

Through the process of literature review for this SLR, scarcity of big data reference architectures has been witnessed. The studies listed above are prominent research, with great potential to induce concrete architectures. But with all, they are mostly published as short journals and provide with little information about architectural qualities, metadata management, and security, privacy concerns. In another terms, they are notion or brief discussions on reference architectures in very particular domains.

3.4. *What are some common approaches to creating BD RAs?*

The findings gained from this study led to the understanding that there are not many frameworks available for design and development of RAs. Nevertheless, to address RQ4, we sought to find the research methodology and approaches chosen to develop RAs. One of the most commonly used approaches for developing RAs is 'Empirically grounded Reference Architectures' by Galster and Avgeriou (2011).The research methodology is well-received because of its emphasis on empirical validity and empirical foundation. This methodology is comprising of 6 step process which are respectively 1) Selecting the type of the RA, 2) Selection of the design strategy, 3) Empirical acquisition of data, 4) Construction of the RA, 5) Enabling RA with variability, 6) Evaluation of the RA.

Another seminal work in this area is a framework for analysis and design of software RAs created by Angelov, Grefen, and Greefhorst (2012). The framework utilizes a multi-dimensional classification space to classify RAs and as a result presents 5 major types. It is developed with the objective of supporting analysis of RAs with regards to their architectural specification/design, goal, and context. This is achieved through three major dimensions, each having their own corresponding subdimensions of design, goal, and context. These dimensions and sub-dimensions are derived by interrogatives of 'why', 'where', 'who', 'when', 'what', and 'how', which is a well-established practice for problem analysis. The interrogative why addresses the goal of the RA, who, when, where address the context, and how and what address the design dimensions. This framework categorizes RAs in two major groups: facilitation RAs and standardization RAs.

Volk, Bosse, Bischoff, and Turowski (2019) utilized Software Architecture Comparison Analysis Method (SCAM) to compare and examine RAs based on their applicability. This result of this work was a decision-support process for selection of BD RAs.

Two standards that have been observed the most were ISO/IEC 25010 for choosing quality software products for RAs (Estdale & Georgiadou, 2018), and ISO/IEC 42010 for architecture description (Emery & Hilliard, 2009).

Surprisingly, based on the evidence gained from this SLR, most researchers and practitioners use informal architectural description methods like boxes and lines, except for the works of Geerdink (2013). In this study, the author used ArchiMate (Josey, Lankhorst, Band, Jonkers, & Quartel, 2016) as the modeling language which is a formal and standard modeling language that is accepted and recommended in ISO/IEC 42010 as well. Informal methods of modeling can introduce inconsistency issues between system design and implementation of the system (Zhu, 2005), do not adhere to a well-established standard and do not promote the development of modeling approaches. Therefore, one can argue that there is a need for more emphasis on the modeling language with which different researchers and practitioners describe ontologies.

Lastly, Hevner's information systems research framework (Hevner, March, Park, & Ram, 2004) has been used for the development of RA presented by Geerdink (2013), which is a suitable research design, since a BD RA is an information system artefact based on existing literature and business needs.

3.5. *What are the challenges of creating BD RAs?*

Among the challenges of developing RAs, perhaps evaluation is the most significant (Maier et al., 2013). According to Galster and Avgeriou (2011), two fundamental pillars of the evaluation is the correctness and the utility of the RA and how efficiently it can be adapted and instantiated.

RAs and concrete architectures come with a different level of abstraction and have divergent qualities. Whereas there are many well-established evaluation methods for concrete architectures such as Architecture Level Modifiability Analysis (Bengtsson, Lassing, Bosch, & van Vliet, 2004), Scenario-based Architecture Analysis Method (SAAM) (Kazman, Abowd, Bass, & Clements, 1996), Architecture Trade-off Analysis Method (ATAM) (Kazman et al., 1998), and Performance Assessment of Software Architecture (Williams & Smith, 2002), none of the these can really be directly applied to RAs.

For instance, ATAM is reliant on participation of stakeholders in early stages for creation of utility tree, and RAs, being highly abstract, do not have a clear group of stakeholders at that stage. In addition, many of evaluation methodologies listed make use of scenarios, whereas RAs are highly abstract and are potentially adopted for various contexts, therefore making scenario creation difficult and sometimes invalid. Either a few general scenarios are developed to cover all aspects, or a large number of specific scenarios are developed to cover various aspects of the RA. Each of which can pose threats to validity.

Based on three problems discussed above, available methods of architecture analysis are not sufficient for evaluating RAs. Various researched tried to address this problem. In one study Angelov et al. (2008), modified ATAM and extended it to resonate well with RAs. This process took place by invitation of representatives from leading industries for the evaluation process, and the selection of various contexts and defined scenarios for these contexts. ATAM was extended to evaluate completeness, buildability and applicability. Howbeit the selection of the right candidate and involving them in the process is a daunting task and unfeasible at times (Angelov et al., 2008).

In Another study by Maier et al. (2013) as a postgraduate thesis in Eindhoven University of Technology, the evaluation of the RA has been conducted by mapping it against existing reference and concrete architectures described in industrial whitepapers and reports. Along the lines, Galster and Avgeriou (2011) suggested reference implementations, prototyping and incremental approach for the validation of the RA.

By the virtue of the findings from this SLR, and by studying the approaches from Bosch (2000), Avgeriou (2003), and Derras et al. (2018a), an evaluation framework for a RA can be done through architectural prototype evaluation, which means a concrete architecture of the RA is generated and then evaluated through a well-grounded method such as ATAM.

3.6. *What are major architectural components of BD RAs?*

To address RQ5, RAs listed in table 1 was reviewed and compared to deduce common architectural components of BD RAs. Some of the RAs collected were in in the form of a short paper and provided with not much detail, whereas some of the other such as NIST were quite comprehensive.

Majority of RAs have been inspired or based on other RAs, and this signified the notion that "RAs can be perceived more effective when they are created out of available knowledge, studied domain, and existing RAs rather than from scratch"

We describe these architectural components as three major categories, namely 'BD management and storage', 'Data Processing and application interfaces', and 'BD infrastructure'.

BD Management and Storage

One of the prominent characteristics of big data is 'variety', which rises the need for distinct storage solutions. This is sometimes referred to as 'polyglot persistence' (Khine & Wang, 2019). For instance, when it comes to dynamic data, NoSQL databases such as MongoDB is a suitable choice because of their non-tabular nature (Banker, Garrett, Bakkum, & Verch, 2016), and when there is a need for complex relationship between entities, graph databases such as Neo4J are more suitable because of their tree traversal performance (Van Bruggen, 2014).

In the same vein, when there is a requirement for a large dataset that can be distributed across multiple database nodes, column wide databases such as Cassandra can be a good architectural component (Carpenter & Hewitt, 2020), and when there is a need for text analysis, one can choose Elastic Search (Tsaousi, 2021).

Choosing the right database or databases, is an important architectural decision that can also include patterns for data access, storage and caching. For example, the practitioners of distributed system that are specialized in micro-services architecture may opt to use Command Query Responsibility Segregation (CQRS) pattern for high performance applications (Márquez & Astudillo, 2018). Therefore, the type of storage and the access pattern are two major architectural components of big data systems.

Another architectural component that is popular in BD systems is data lake. Data lake can be perceived as an ingestion framework that can be given various types of data including internal and external data. The data stored in the data lake can then be stored for cleansing, preparation and modeling through data pipelines (Sawadogo & Darmont, 2021).

Similar to the way that Business Intelligence (BI) and BD differ in their source data types both in terms of granularity and data structure of it, a data lake and data warehouse share some of those same differences. In the case of a data warehouse, usually a relational database is used which decreases flexibility when it comes to analysis. In the case of data lake, data of different kind can be stored without the engineer needing to define the schema in advance. This increases the flexibility.

Howbeit, this flexibility itself has its own downside and can be abused by data engineers. One can throw different data sets without much regard at all for how they're structured, which leads to what people refer to as data swamp (Khine & Wang, 2018). Data governance can alleviate some of these issues (Josey, 2016).

Based on this, another major architectural component of BD RAs is a data lake.

Data Processing and Application Interfaces

The result of this SLR shed lights on two major data processing activities that a BD system encompasses. These processes generally fall into stream processing and batch processing. Stream processing or fast processing is required for sensitive operations and time critical

processes such as checking a fraudulent credit card, and batch processing required for a long-running continuum of data analysis such as regression analysis.

The decision on required type of processing for a context-specific architecture is determined by the characteristics of the data being analyzed, that is primarily variety, volume and velocity (Chang & Boyd, 2018).

For instance, most algorithms for stream processing are using in-memory stateful data structures such Hyperloglog to compute values in real-time (Sahal, Breslin, & Ali, 2020). A streaming component can be tailored to adopt specific windowing approaches such as tuple-at-a-time and a micro-batch processing. When in fact, these techniques are not required for batch processing. An architect may opt for MapReduce and Bulk Synchronous Parallel (BSM) processing for batch-oriented requirements or go for a streaming processing based on a specific performance requirement set to handle velocity and volume of data.

Moreover, every BD RA analyzed provided the notion of data pipelines, even when the exact phrase wasn't used. This is due to the fact that in BD systems, data usually goes through several phases before it's ready for statistical analysis (Geerdink, 2013). This is quite evident when it comes to Lambda (Hausenblas & Bijnens, 2015) and Bolster architectures (Nadal et al., 2017), where data goes through either batch ingestion or stream ingestion components, gets dispatched to the data lake, gets into batch or streaming processing component, gets into batch or real-time views and from there on into the query engine and finally the data analyst. In fact, Bolster is an extension of Lambda, with further improvement on metadata management.

Therefore, batch processing, stream processing and data pipelines are considered major architectural components.

BD Infrastructure

BD infrastructure provides BD systems with services or resources required to ingest, store, and analyze data. There are many technologies that can be utilized for BD infrastructures, and many harnesses to power of cloud services such as AWS EC2. For instance, one major component of a NIST BD RA is called big data framework provider which includes 'computing and analytics', 'data organization and distribution', and 'infrastructures such as networking, computing and storage'.

Despite the former two major components, BD infrastructure is more of a layer than a component. A layer in which the RA lays out a possible computing and networking design of a BD system. This is crucial, as practitioners of BD have been commonly architecting underlying distributed paradigms and horizontal scaling. In recent years, the advent of web 3.0 and micro-services architecture, have shifted the overall paradigm of software engineering and data engineering towards decentralization and distribution (Gan et al., 2019).

Therefore, CAP theorem, ACID and BASE transactions, data consistency, service discovery, and tail latency are potential architectural challenges one should consider. Should a BD system adopt an event-driven approach through an event backbone such as Kafka? Or should it stick to REST based communication. What is the overhead of context switch and networking in the case of RPCs among services?

All and all, as a result of this SLR, a component of a BD infrastructure has been witnessed as a common pattern, in various forms and approaches. We decided to present this as layer as it's play an important role in creation and design of an RA. Therefore, we consider platform layer of BD as a major architectural component of these systems. Whereas one

might argue that infrastructure is an architectural component of any system, the design challenge is more significant in the case of BD systems as these systems are usually distributed in nature.

Lastly, our findings depicts the fact that many of RA presented are not designed underlying completely distributed architecture while BD systems can benefit from this paradigm (Klimentov et al., 2015; Mazumder, Bhadoria, & Deka, 2017).

3.7. *What are the limitations of current BD RAs?*

To answer the RQ7, RAS collected for this SLR have been appraised to point out limitations. One of the limitations that we came across was that the concept of metadata has been poorly discussed. For instance, Maier et al. (2013) discussed the limitation of metadata management systems, stating that most metadata solutions are ad-hoc. The researcher then went ahead and introduced a layer for metadata management in the RA, but as a non-integrated component. For instance, the author did not discuss how data provenance can be achieved through the RA and underlying which logical flow one can do linear analysis.

In another case, NIST BD RA only discusses metadata in a sentence, and in sub-activity named ‘metadata management’. The RA only states what are essential metadata information and how they are used. Except for one BD RA (Bolster), metadata has not been accentuated enough and metadata layer is not thoroughly discussed. This is a noticeable limitation in current BD RAs, as metadata plays an important role in BD systems, addressing wide range of challenges such as privacy, security, data provenance, and linear analysis (Eichler 2019).

Based on that, one can argue that any BD system can benefit from a well-defined metadata layer as a means for bridging data stored in different platforms such as on-premises or on cloud, reducing complexity, facilitating access management, facilitating data governance, and potentially the creation of data mesh (Chang and Boyd 2018).

Furthermore, white papers collected from IT giants tend to pivot the RA around their services, which can potentially reduce its applicability, hinder RAs openness, and even affect architectural qualities. In these white papers, alternative technologies or vendors are typically not discussed which leaves the architect with a small pool of options.

Another limitation with current BD RAs is that they have not fully absorbed the advancements of software architecture and software engineering. While this is a factor of age in some of the RAs, some other for instance do not fully benefit from micro-services patterns (Márquez & Astudillo, 2018), and container technologies such as Docker (Wang, Kadiyala, & Rubin, 2021). Majority of RAs tend to be portrayed underlying old ‘monolithic’ structure which has been proven to be troublesome to maintain and scale in the long run.

Lastly, privacy and security does not seem to have been discussed enough, or it has been mostly marginalized. For instance, we have not found an architectural component that allows for data scrubbing, or we did not understand how one can achieve security in-between data pipelines. Specially, in regard to privacy and with recent global movements towards increased privacy, BD architects are now increasingly challenged to design underlying the shadow of regional data privacy policies (Bashari Rad et al., 2016). Placing this challenge next to security challenges of package management, endpoint proliferation and DDOS handling can further signify more research on BD RAs.

Many of core architectural decisions revolving around security and privacy, if not addressed in an initial phase, can result in massive losses and potential bottlenecks.

4 Conclusion

This study sought to find all BD RAs available in practice and academia. The findings gained emerges the understanding that RAs can be an effective artefact to tackle complex BD system development. RAs at their core bring software engineering knowledge as a collection of patterns designed to address a class of problems with attention to specific requirements and context and do solve many of the prevalent architectural challenges that an architect might face.

As data proliferates further, there will be more BD systems created which in turn means more technology orchestration is required around data that can be effectively done through a well-established RA. RAs guide the evolution of the system both in terms of functional and non-functional requirements, and pinpoint variability points that can result in more successful BD projects and avoidance of common pitfalls.

Withal, BD RAs have yet to mature and become ubiquitous in industry and there is further research required in this area. These researches can be done in the area of micro-services RA for BD systems, event-driven paradigms for BD systems, security and privacy issues in BD systems, and metadata management.

5 References

AI, V. (2019). Why do 87% of data science projects never make it into production? Retrieved from https://venturebeat.com/2019/07/19/why-do-87-of-data-science-projects-never-make-it-into-production/

Akhtar, P., Frynas, J. G., Mellahi, K., & Ullah, S. (2019). Big data-savvy teams' skills, big data-driven actions and business performance. *British Journal of Management, 30*(2), 252-271.

Angelov, S., Grefen, P., & Greefhorst, D. (2009). *A classification of software reference architectures: Analyzing their success and effectiveness.* Paper presented at the 2009 Joint Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture.

Angelov, S., Grefen, P., & Greefhorst, D. (2012). A framework for analysis and design of software reference architectures. *Information and Software Technology, 54*(4), 417-431.

Angelov, S., Trienekens, J. J., & Grefen, P. (2008). *Towards a method for the evaluation of reference architectures: Experiences from a case.* Paper presented at the European Conference on Software Architecture.

Ataei, P., & Litchfield, A. T. (2020). Big Data Reference Architectures, a systematic literature review.

Avgeriou, P. (2003). Describing, instantiating and evaluating a reference architecture: A case study. *Enterprise Architecture Journal, 342*, 1-24.

Banker, K., Garrett, D., Bakkum, P., & Verch, S. (2016). *MongoDB in Action: Covers MongoDB version 3.0*: Simon and Schuster.

Bashari Rad, B., Akbarzadeh, N., Ataei, P., & Khakbiz, Y. (2016). Security and Privacy Challenges in Big Data Era. *International Journal of Control Theory and Applications, 9*(43), 437-448.

Bengtsson, P., Lassing, N., Bosch, J., & van Vliet, H. (2004). Architecture-level modifiability analysis (ALMA). *Journal of Systems and Software, 69*(1-2), 129-147.

Borrego, M., Foster, M. J., & Froyd, J. E. (2014). Systematic literature reviews in engineering education and other developing interdisciplinary fields. *Journal of Engineering Education, 103*(1), 45-76.

Bosch, J. (2000). *Design and use of software architectures: adopting and evolving a product-line approach*: Pearson Education.

Cackett, D. (2013). Information Management and Big data: A Reference Architecture. *Oracle: Redwood City, CA, USA*.

Carpenter, J., & Hewitt, E. (2020). *Cassandra: the definitive guide: distributed data at web scale*: O'Reilly Media.

Chang, W. L., & Boyd, D. (2018). *NIST Big Data Interoperability Framework: Volume 6, Big Data Reference Architecture*. Retrieved from

Chang, W. L., & Grady, N. (2015). *NIST Big Data Interoperability Framework: Volume 1, Big Data Definitions*. Retrieved from

Chang, W. L., & Mishra, S. (2015). *NIST Big Data Interoperability Framework: Volume 5, Architectures White Paper Survey*. Retrieved from

Chen, H.-M., Kazman, R., Garbajosa, J., & Gonzalez, E. (2017). *Big Data Value Engineering for Business Model Innovation.* Paper presented at the Proceedings of the 50th Hawaii International Conference on System Sciences.

Chen, H.-M., Kazman, R., & Haziyev, S. (2016). Agile big data analytics for web-based systems: An architecture-centric approach. *IEEE Transactions on Big Data, 2*(3), 234-248.

Cloutier, R., Muller, G., Verma, D., Nilchiani, R., Hole, E., & Bone, M. (2010). The concept of reference architectures. *Systems Engineering, 13*(1), 14-27.

Cuzzocrea, A. (2016). *A reference architecture for supporting secure big data analytics over cloud-enabled relational databases.* Paper presented at the 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC).

Demchenko, Y., De Laat, C., & Membrey, P. (2014). *Defining architecture components of the Big Data Ecosystem.* Paper presented at the 2014 International conference on collaboration technologies and systems (CTS).

Derras, M., Deruelle, L., Douin, J.-M., Levy, N., Losavio, F., Pollet, Y., & Reiner, V. (2018a). *Reference Architecture Design: A Practical Approach.* Paper presented at the ICSOFT.

Derras, M., Deruelle, L., Douin, J. M., Levy, N., Losavio, F., Pollet, Y., & Reiner, V. (2018b). *Reference Architecture Design: a practical approach.* Paper presented at the 13th International Conference on Software Technologies (ICSOFT).

Emery, D., & Hilliard, R. (2009). *Every architecture description needs a framework: Expressing architecture frameworks using ISO/IEC 42010.* Paper presented at the 2009 Joint Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture.

Erevelles, S., Fukawa, N., & Swayne, L. (2016). Big Data consumer analytics and the transformation of marketing. *Journal of Business Research, 69*(2), 897-904.

Estdale, J., & Georgiadou, E. (2018). *Applying the ISO/IEC 25010 quality models to software product.* Paper presented at the European Conference on Software Process Improvement.

Galster, M., & Avgeriou, P. (2011). *Empirically-grounded reference architectures: a proposal.* Paper presented at the Proceedings of the joint ACM SIGSOFT conference--QoSA and ACM SIGSOFT symposium--ISARCS on Quality of software architectures--QoSA and architecting critical systems--ISARCS.

Gan, Y., Zhang, Y., Cheng, D., Shetty, A., Rathi, P., Katarki, N., . . . Jackson, B. (2019). *An open-source benchmark suite for microservices and their hardware-software implications for cloud & edge systems.* Paper presented at the Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems.

Gartner. (2014). Survey Analysis: Big Data Investment Grows but Deployments Remain Scarce in 2014. Retrieved from http://www.gartner.com/document/2841519

Geerdink, B. (2013). *A reference architecture for big data solutions introducing a model to perform predictive analytics using big data technology.* Paper presented at the 8th International Conference for Internet Technology and Secured Transactions (ICITST-2013).

Ghandour, A. (2015). Big Data driven e-commerce architecture. *International Journal of Economics, Commerce and Management, 3*(5), 940-947.

Gorton, I., & Klein, J. (2015). Distribution, Data, Deployment. *STC 2015*, 78.

Grefen, P., & de Vries, R. R. (1998). A reference architecture for workflow management systems. *Data & Knowledge Engineering, 27*(1), 31-57.

Hausenblas, M., & Bijnens, N. (2015). Lambda architecture. *URL: http://lambda-architecture.net/. Luettu, 6*, 2014.

Heilig, L., & Voß, S. (2017). Managing cloud-based Big Data platforms: A reference architecture and cost perspective. In *Big Data Management* (pp. 29-45): Springer.

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

Hilliard, R. ISO/IEC/IEEE 42010. In.

House, W. (2012). Big Data is a Big Deal. Retrieved from https://obamawhitehouse.archives.gov/blog/2012/03/29/big-data-big-deal

Hummel, O., Eichelberger, H., Giloj, A., Werle, D., & Schmid, K. (2018). *A collection of software engineering challenges for big data system development.* Paper presented at the 2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA).

Jagadish, H. V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J. M., Ramakrishnan, R., & Shahabi, C. (2014). Big data and its technical challenges. *Communications of the ACM, 57*(7), 86-94.

Josey, A. (2016). *TOGAF® Version 9.1-A Pocket Guide*: Van Haren.

Josey, A., Lankhorst, M., Band, I., Jonkers, H., & Quartel, D. (2016). An introduction to the ArchiMate® 3.0 specification. *White Paper from The Open Group*.

Kazman, R., Abowd, G., Bass, L., & Clements, P. (1996). Scenario-based analysis of software architecture. *IEEE software, 13*(6), 47-55.

Kazman, R., Klein, M., Barbacci, M., Longstaff, T., Lipson, H., & Carriere, J. (1998). *The architecture tradeoff analysis method.* Paper presented at the Proceedings. Fourth IEEE International Conference on Engineering of Complex Computer Systems (Cat. No. 98EX193).

Khine, P. P., & Wang, Z. (2019). A review of polyglot persistence in the big data world. *Information, 10*(4), 141.

Khine, P. P., & Wang, Z. S. (2018). *Data lake: a new ideology in big data era.* Paper presented at the ITM web of conferences.

Kiran, M., Murphy, P., Monga, I., Dugan, J., & Baveja, S. S. (2015). *Lambda architecture for cost-effective batch and speed big data processing.* Paper presented at the 2015 IEEE International Conference on Big Data (Big Data).

Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering–a systematic literature review. *Information and software technology, 51*(1), 7-15.

Kitchenham, B. A., Dyba, T., & Jorgensen, M. (2004). *Evidence-based software engineering.* Paper presented at the Proceedings of the 26th international conference on software engineering.

Klein, J., Buglak, R., Blockow, D., Wuttke, T., & Cooper, B. (2016). *A reference architecture for big data systems in the national security domain.* Paper presented at the 2016 IEEE/ACM 2nd International Workshop on Big Data Software Engineering (BIGDSE).

Klimentov, A., Buncic, P., De, K., Jha, S., Maeno, T., Mount, R., . . . Petrosyan, A. (2015). *Next generation workload management system for big data on heterogeneous distributed computing.* Paper presented at the Journal of physics: conference series.

Kohler, J., & Specht, T. (2019). Towards a Secure, Distributed, and Reliable Cloud-Based Reference Architecture for Big Data in Smart Cities. In *Big Data Analytics for Smart and Connected Cities* (pp. 38-70): IGI Global.

Kreps, J. (2014). Questioning the Lambda Architecture. The Lambda Architecture has its merits, but alternatives are worth exploring. *O'Reilly Media. https://www. oreilly. com/ideas/questioning-the-lambda-architecture. Zugegriffen am, 21*, 2020.

Levin, B. (2013). Big data ecosystem reference architecture. *Microsoft Corporation*.

Li, Q., Xu, Z., Chan, I., Yang, S., Pu, Y., Wei, H., & Yu, C. (2018). *Big data architecture and reference models.* Paper presented at the OTM Confederated International Conferences" On the Move to Meaningful Internet Systems".

Maier, M., Serebrenik, A., & Vanderfeesten, I. (2013). Towards a big data reference architecture. *University of Eindhoven*.

Mannering, F., Bhat, C. R., Shankar, V., & Abdel-Aty, M. (2020). Big data, traditional data and the tradeoffs between prediction and causality in highway-safety analysis. *Analytic methods in accident research, 25*, 100113.

Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., & Byers, A. H. (2011). Big data: The next frontier for innovation, competition, and productivity.

Márquez, G., & Astudillo, H. (2018). *Actual use of architectural patterns in microservices-based open source projects.* Paper presented at the 2018 25th Asia-Pacific Software Engineering Conference (APSEC).

Martínez-Prieto, M. A., Cuesta, C. E., Arias, M., & Fernández, J. D. (2015). The solid architecture for real-time management of big semantic data. *Future Generation Computer Systems, 47*, 62-79.

Mazumder, S., Bhadoria, R. S., & Deka, G. C. (2017). Distributed computing in big data analytics. In *InCon-cepts, Technologies and Applications 2017*: Springer.
Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., & The, P. G. (2009). Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The PRISMA Statement. *PLOS Medicine, 6*(7), e1000097. doi:10.1371/journal.pmed.1000097
Nadal, S., Herrero, V., Romero, O., Abelló, A., Franch, X., Vansummeren, S., & Valerio, D. (2017). A software reference architecture for semantic-aware Big Data systems. *Information and software technology, 90*, 75-92.
Nash, H. (2015). CIO SURVEY 2015. *Association with KPMG*.
OATH, O. Reference Architecture. *Release, 2*, 2004-2007.
Pääkkönen, P., & Pakkala, D. (2015). Reference architecture and classification of technologies, products and services for big data systems. *Big data research, 2*(4), 166-186.
Partners, N. (2019). Big Data and AI Executive Survey 2019. *Data and Innovation*.
Piñeiro, C., Morales, J., Rodríguez, M., Aparicio, M., Manzanilla, E. G., & Koketsu, Y. (2019). Big (pig) data and the internet of the swine things: a new paradigm in the industry. *Animal frontiers, 9*(2), 6-15.
Pope, A. L. (1998). *The CORBA reference guide: understanding the common object request broker architecture*: Addison-Wesley Longman Publishing Co., Inc.
Quintero, D., & Lee, F. N. (2019). *IBM reference architecture for high performance data and AI in healthcare and life sciences*: IBM Corporation, International Technical Support Organization.
Rad, B. B., & Ataei, P. (2017). The big data Ecosystem and its Environs. *International Journal of Computer Science and Network Security (IJCSNS), 17*(3), 38.
Rada, B. B., Ataeib, P., Khakbizc, Y., & Akbarzadehd, N. (2017). The Hype of Emerging Technologies: Big Data as a Service.
Rahimi, S. K., & Haug, F. S. (2010). *Distributed database management systems: A Practical Approach*: John Wiley & Sons.
Reed. (2002). Reference Architecture: The Best of Best Practices. Retrieved from http://www.ibm.com/developerworks/rational/library/2774.html
Sahal, R., Breslin, J. G., & Ali, M. I. (2020). Big data and stream processing platforms for Industry 4.0 requirements mapping for a predictive maintenance use case. *Journal of Manufacturing Systems, 54*, 138-151.
Sang, G. M., Xu, L., & De Vrieze, P. (2016). *A reference architecture for big data systems.* Paper presented at the 2016 10th International Conference on Software, Knowledge, Information Management & Applications (SKIMA).
SAP. (2016). Whitepaper NEC SAPHANA Hadoop. Retrieved from https://www.scribd.com/document/418835912/Whitepaper-NEC-SAPHANA-Hadoop
Sawadogo, P., & Darmont, J. (2021). On data lake architectures and metadata management. *Journal of Intelligent Information Systems, 56*(1), 97-120.
Shamseer, L., Moher, D., Clarke, M., Ghersi, D., Liberati, A., Petticrew, M., . . . Stewart, L. A. (2015). Preferred reporting items for systematic review and meta-analysis protocols (PRISMA-P) 2015: elaboration and explanation. *Bmj, 349*.
Singh, N., Lai, K.-H., Vejvar, M., & Cheng, T. (2019). Big data technology: Challenges, prospects and realities. *IEEE Engineering Management Review*.
Stricker, V., Lauenroth, K., Corte, P., Gittler, F., De Panfilis, S., & Pohl, K. (2010). Creating a reference architecture for service-based systems–a pattern-based approach. In *Towards the Future Internet* (pp. 149-160): IOS Press.
Suthakar, U. (2017). *A scalable data store and analytic platform for real-time monitoring of data-intensive scientific infrastructure.* Brunel University London,
Tsaousi, K. D. (2021). Elasticity of Elasticsearch. In.
Ünal, P. (2019). *Reference Architectures and Standards for the Internet of Things and Big Data in Smart Manufacturing.* Paper presented at the 2019 7th International Conference on Future Internet of Things and Cloud (FiCloud).
Van Bruggen, R. (2014). *Learning Neo4j*: Packt Publishing Ltd.

Vassakis, K., Petrakis, E., & Kopanakis, I. (2018). Big data analytics: applications, prospects and challenges. In *Mobile big data* (pp. 3-20): Springer.

Viana, P., & Sato, L. (2014). *A proposal for a reference architecture for long-term archiving, preservation, and retrieval of big data.* Paper presented at the 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications.

Volk, M., Bosse, S., Bischoff, D., & Turowski, K. (2019). *Decision-support for selecting big data reference architectures.* Paper presented at the International Conference on Business Information Systems.

Wang, Y., Kadiyala, H., & Rubin, J. (2021). Promises and challenges of microservices: an exploratory study. *Empirical Software Engineering, 26*(4), 1-44.

Weyrich, M., & Ebert, C. (2015). Reference architectures for the internet of things. *IEEE software, 33*(1), 112-116.

White, A. (2019). Top Data and Analytics Predicts for 2019.

Williams, L. G., & Smith, C. U. (2002). *PASASM: a method for the performance assessment of software architectures.* Paper presented at the Proceedings of the 3rd International Workshop on Software and Performance.

Zhu, H. (2005). *Software design methodology: From principles to architectural styles*: Elsevier.

Zimmerman, H. (1980). reference model-the OSI model of archictecture for open system interconnection'IEEE Trans. In: Commun.