Software Engineering for Big Data Systems

by

Vijay Dipti Kumar

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science
Waterloo, Ontario, Canada, 2017

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Software engineering is the application of a systematic approach to designing, operating and maintaining software systems and the study of all the activities involved in achieving the same. The software engineering discipline and research into software systems flourished with the advent of computers and the technological revolution ushered in by the World Wide Web and the Internet. Software systems have grown dramatically to the point of becoming ubiquitous. They have a significant impact on the global economy and on how we interact and communicate with each other and with computers using software in our daily lives.

However, there have been major changes in the type of software systems developed over the years. In the past decade owing to breakthrough advancements in cloud and mobile computing technologies, unprecedented volumes of hitherto inaccessible data, referred to as big data, has become available to technology companies and business organizations farsighted and discerning enough to use it to create new products, and services generating astounding profits. The advent of big data and software systems utilizing big data has presented a new sphere of growth for the software engineering discipline. Researchers, entrepreneurs and major corporations are all looking into big data systems to extract the maximum value from data available to them. Software engineering for big data systems is an emergent field that is starting to witness a lot of important research activity.

This thesis investigates the application of software engineering knowledge areas and standard practices, established over the years by the software engineering research community, into developing big data systems by:

- surveying the existing software engineering literature on applying software engineer-

ing principles into developing and supporting big data systems;

- identifying the fields of application for big data systems;
- investigating the software engineering knowledge areas that have seen research related to big data systems;
- revealing the gaps in the knowledge areas that require more focus for big data systems development; and
- determining the open research challenges in each software engineering knowledge area that need to be met.

The analysis and results obtained from this thesis reveal that recent advances made in distributed computing, non-relational databases, and machine learning applications have lured the software engineering research and business communities primarily into focusing on system design and architecture of big data systems. Despite the instrumental role played by big data systems in the success of several businesses organizations and technology companies by transforming them into market leaders, developing and maintaining stable, robust, and scalable big data systems is still a distant milestone. This can be attributed to the paucity of much deserved research attention into more fundamental and equally important software engineering activities like requirements engineering, testing, and creating good quality assurance practices for big data systems.

Acknowledgements

First and foremost I would like to thank Professor Paulo Alencar, for his guidance, support, and understanding. It has been a honor working under his tutelage. Also thanks to Professor Daniel Berry for serving as my co-supervisor; and Professors Donald Cowan and Gladimir Baranoski for agreeing to read this thesis and for their valuable feedback. I would also like to thank Ivens Portugal for his help, feedback, and ideas.

Special thanks to my parents for their unconditional love and support and to my close friends Khin Phyo Hlaing and Philip McCarthy for their love, support, and patience in seeing me through the difficult times during my Master's program.

Dedication

This is dedicated to Neelanjana.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

If a Google search is performed to uncover the provenance of big data, a few interesting articles turn up like "A Very Short History of Big Data" [1] and "A Brief History of Big Data Everyone Should Read" [2]. These articles reveal concerns about the growing amounts of information being produced dating back as early as 1800 when the US Census Bureau estimated that it would take them 8 years to process all the data they had collected in the 1800 census and 10 years to process all the information collected in the 1890 census which would by then be outdated because it would be time for the 1900 census. It was a Bureau employee, Herman Hollerith's punchcard invention that decreased this processing time from ten years to three months and earned him the title of "Father of modern automated computation". Hollerith went on to establish a company that would later metamorphose into IBM.[3]

[1] A Very Short History of Big Data - http://www.forbes.com/sites/gilpress/2013/05/09/a-very-short-history-of-big-data/#3fc4fe9055da

[2] A Brief History of Big Data Everyone Should Read - https://www.linkedin.com/pulse/brief-history-big-data-everyone-should-read-bernard-marr

[3] Herman Hollerith - http://www.columbia.edu/cu/computinghistory/hollerith.html

As early as 1967, concerns about being able to store all the data being produced were being raised. A paper titled "Automatic Data Compression", published by ACM discussed the "information explosion" would make it essential that storage requirements for all information be kept to a minimum. It proposed the idea of a new type of information compressor which could be used with "any" body of information to greatly reduce slow external storage requirements and to increase the rate of information transmission through a computer [102].

Access to a large volume of data, gave birth to terms like "Business Intelligence" that sparked ideas about using commercial customer data by businesses to increase sales and improve profit margins. In fact, the first time this term was used was in *"Cyclopaedia of Commercial and Business Anecdotes..."* in 1865 describing how banker Henry Furnese was able to get ahead of his competition by analyzing information collected about his business activities and contacts[41].

The New York Times Magazine ran an article[4] on how Target was using predictive analytics on big data to attract new customers at a vulnerable point of time in their lives when they were more likely to switch brand loyalties. The vulnerable point was when women/couples were about to have a baby and Target wanted to know exactly when the customer was about to start shopping for baby products. According to the article, "new parents are a retailer's holy grail... There are, however, some brief periods in a person's life when old routines fall apart and buying habits are suddenly in flux. One of those moments, the moment, really is right around the birth of a child, when parents are exhausted and overwhelmed and their shopping patterns and brand loyalties are up for grabs... We knew that if we could identify them in their second trimester, there is a good chance we could

[4]NYTM-Predictive Analytics and Shopping Habits-http://www.nytimes.com/2012/02/19/magazine/shopping-habits.html

capture them for years... As soon as we get them buying diapers from us, they are going to start buying everything else too."

This article even prompted popular late night comedy show host Stephen Colbert to make a video titled "Surrender to a Buyer Power". The New York Times Magazine article and Colbert's video brought to the attention of the mainstream, the power and reach of big data and predictive analytics methods that retail giants like Target and Walmart were using to sell more products, capitalize new markets and increase profits.

1.1 Background

Big Data has been on the forefront of the technological revolution in the world of computing for the latter half of the past decade. The use of big data has seen unprecedented success in areas including but not limited to search engines, social networking, e-commerce, and audio and video streaming services. A few of the popular players and leaders in these fields leveraging big data are Google, Facebook, LinkedIn, Twitter, Amazon, eBay, Spotify, Apple and Netflix. Owing of the popularity of big data and the enormous success of Internet born companies like Google, Amazon, and Facebook attributed to their manipulation of big data, there has been tremendous academic and commercial interest in exploring, researching, and exploiting the data available in the world today.

In addition to the examples described until now in which big data is influencing technology and retail giants, it is already being utilized or showing signs of incipient growth in healthcare, politics, aviation, banking and finance, telecommunications, manufacturing, meteorology, environmental conservation and many, many more fields, henceforth collectively referred to as "application domains" in this thesis.

According to a Gartner survey[5], 64% out of 720 respondents (part of the Gartner Research Circle members worldwide) had invested or planned to invest in big data applications in 2013. However, less than 8% had actually deployed at the time of the survey. Most of the business cases identified for future application of big data technology were related to improving process efficiency and to enrich customer experience. As shown in a case study of optimizing the manufacturing process of digital displays [116], it is possible to enhance process efficiency using big data. Enriching customer experience using big data has been exemplified in the form of the "People You May Know" option offered by Facebook and LinkedIn or the movie recommendations in Netflix and even the "Customers Who Bought This Item Also Bought" service provided by Amazon.

In order to understand the software engineering challenges specific to big data systems, background information about big data and software engineering needs to be discussed.

Characterizing Big Data

It is clear that there is a huge demand for big data systems. However, one of the most important questions about big data is "What qualifies large volumes of data to be considered as big data"?

The first characterization of big data came in the form of describing its dimensions as the 3Vs : Volume, Velocity and Variety in 2001 [89]. This characterization was later expanded to include two more dimensions - Veracity and Validity making it 5Vs. In addition to the popular 5Vs, more dimensions namely, Volatility and Value were added [78].

[5]Big Data Gartner survey-http://www.gartner.com/newsroom/id/2593815

1. **Volume** - Volume implies the data explosion that has come to mark the last decade in the world of computing. The size of the data that is being created by the World Wide Web, smartphones, smart appliances, social media, electronic medical record systems, wearable technology monitoring health metrics (e.g. Fitbit, Apple Watch), space exploration, geographic information systems, commercial flights, military aircrafts and tanks, surveillance cameras in cities, transaction data from credit cards and point of sale machines in retail and commercial establishments, weather monitoring stations all over the world and many, many more sources are overwhelming our current data processing and storage systems like relational databases and SQL servers.

2. **Velocity** - Velocity is the constraint that demands real time processing of the data available, the failure of which would result in its loss or obsolescence. This attribute of big data is very closely related to the previous attribute - Volume. It is not just that petabytes of data are generated and made available, all this is occurring in real time and applications that are consuming this big data have to process all this information instantly or stand to lose it. One aspect that could be considered is that this incoming data may be stored for later processing and analysis. However, the conditions when said data was obtained and curated for later analysis also needs to be recorded to retain proper context. This indirectly means that the meta data of the data also would need to be stored to keep the data at hand relevant.

3. **Variety** - Variety means that the format of the data can be structured, semi-structured, unstructured, streaming or come from multiple sources. Data can be available in the form of plain text, structured text, audio, video, images and a lot more. Structured data is the kind of data we store in relational databases, that have a definite base type like characters, numbers, e.g., account numbers, account balances,

withdrawal limits that show up in a regular bank statement. Semi-structured data is the type of data that has a partial structure and meta-model (a well defined concept to express and process the information) to it. Example of semi-structured data would be information defined in XML. The XML tags give the data structure and make it easier to process. Unstructured data is the type of data that have no rules or conventions for explaining its format or the type of information it contains. Information can be available in any format — Twitter feeds, Facebook posts, Snapchat videos and much more. These kind of data do not have a meta-model that explains the contents and its parts which makes it harder to program a system that is expected to handle this kind of data. Streaming data is any kind of data that is generated continuously and can contain a mixture of structured, unstructured or semi-structured data. Data generated from the logs of mobile applications and phones, data from server logs, e-commerce purchases through a third party authorization body like Visa, etc. Applications processing big data must be capable of handling this variety in input data format and the multiple sources from which they originate. Owing to this variety in data, the possibility of the data containing errors also increases mainly because a lot of data is the output of human interaction with machines and software.

4. **Veracity** - Veracity refers to the truthfulness of the available data - historical or even real-time streaming data. Each and every data point in a massive data collection may not be accurate. There may be abnormalities or noise in the data at hand which needs to be cleaned prior to processing to ensure its usefulness to the application. All data coming into a system may not be useful or trustworthy. e.g., if a political party's public relations team were performing data analytics to understand the opinion and voting inclinations of the residents of a particular state towards their candidate for the state Senate, data mined from social media or news websites like Facebook or

Breitbart may not be accurate because there is no verification mechanism in place to check the authenticity of Facebook posts or articles appearing in controversial news website like Breitbart. Recently, Google and Facebook were reported to be implementing measures to crack down on fake news websites. Such websites were blamed for spreading misinformation and creating confusion among voters during the 2016 US Presidential Election[6]. Any data mining system trying to process data from news websites for even a simple social analytics project would get false information from such fake websites if the correct veracity checks are not put in place.

5. **Validity** - Validity is the constraint that questions if a particular data set, despite being truthful, is still relevant to the problem being analyzed. There may be a large data set available for processing but every data point in this collection need not be processed to extract information. If the data point falls outside the time frame for which the data needs to be analyzed, then such data would not be valid to the problem. For example, if meteorologists were looking into the causes for cloud bursts in arid regions, the year round data collected from a weather station in such a geographical region of interest may be irrelevant except for the time periods before, after and during the actual occurrence of a cloud burst. A big portion of the data recovered from such weather stations would be irrelevant to the problem being analyzed. The collection and storage of such data is futile and only involves consumption of resources and irrelevant data like this may need to be eliminated beforehand and only useful data must be stored and processed for extracting information.

[6]Crackdown on fake news websites - http://www.nytimes.com/2016/11/15/technology/google-will-ban-websites-that-host-fake-news-from-using-its-ad-service.html?_r=0

6. **Volatility** - Volatility signifies one of the most difficult aspects of working with big data. This is due to the fact that in big data environments it is quite common for data to change constantly from one form to another or to become irrelevant within a matter of a few hours. If big data systems do not account for this constant change that input data is capable of going through, any analytical results derived may be inaccurate. This potential inaccuracy is most important when real-time processing is crucial such as in stock markets or telecommunications. Volatility also is a factor when deciding the storage duration of the data collected. In the past, structured data was stored in relational database systems and historical data spanning multiple decades were stored in data warehouses. However, the unstructured and streaming data produced in today's world make it difficult to decide how long the data would remain valid and needs to be stored because security and storage of unstructured and streaming data has become a resource intensive and expensive affair.

7. **Value** - Value is the most important dimension of big data — the end game. Big data in itself may be of no particular value, the amount of information extracted from big data, the analysis performed on this information and the conclusions derived and the measures put into effect based on these conclusions make the value of big data its most important dimension. The value of big data is in how organizations will put this data to use to make their products more effective, far-reaching and ubiquitous. For example, Google is the most used among search engines which resulted in the word "google" making an official entry in the English dictionary in 2006. On the other hand, Facebook continues to be the most important social media website since its launch in 2004 and despite facing stiff competition from Twitter, Instagram and Snapchat. Despite Google's being a leader in the big data and analytics field, it still wasn't able to usurp Facebook's position as the most popular social media website

with Google+, Google's social networking website. There are several reasons behind this interesting occurrence but one of them is that Facebook is more adept at utilizing big data to fit its social media platform than Google has been with Google+.

Software Engineering and Its Knowledge Areas

Mainstream big data software companies like Facebook, Netflix and Amazon have proved that it is possible to create successful big data systems. However, there is much ground yet to be covered, and the software engineering community has a big role to play in making it possible for big data system developers to master the techniques and processes necessary to build fully functional, robust and scalable big data systems.

In 1968, at the first world conference on software engineering, sponsored by the North Atlantic Treaty Organization (NATO), there was general consensus about the repeated occurrences of failure in creating software that satisfied all the stakeholders involved. Terms like "software crisis" and "software failure" used to emphasize the problem, motivated the need for developing best practices for the tasks involved in developing software [123]. Fortunately for the world of computing, software engineering evolved over the years to have multiple disciplines with rich literature, extensive research, best practices and international standards to guide the design, development and testing of software. Fast forward to 2005, The Software Engineering Body of Knowledge - an international standard ISO/IEC TR 19759:2005 was created by the IEEE Computer Society to summarize the generally accepted knowledge about software engineering. The latest standard released was ISO/IEC TR 19759:2015[7], in which multiple software engineering disciplines are categorized into specific knowledge areas.

[7] https://www.iso.org/standard/67604.html

Notwithstanding the creation of standard practices, the whole process of software development continues to be susceptible to errors and problems due to changes in requirements, the environment or communication issues between the stakeholders involved. And this is true even today despite the fact that we have been developing software for more than 40 years. When factors such as the 7Vs of big data are taken into account, the complexities involved in developing software utilizing big data, hereon referred to as big data systems, only increase. The standard practices and guidelines laid down by the software engineering community was developed for developing regular software — software that did not deal with big data. This thesis looks into research done till date in applying the standard practices and guidelines developed by the software engineering community for developing big data systems. Owing to the complexities involved in manipulating big data due to the 7Vs, it is important that the standards and guidelines be leveraged to build robust and scalable big data systems. It would be to the advantage of the stakeholders and developers involved in building a big data system that the best practices and methodologies laid down by the software engineering research community be applied to build systems that are fault tolerant, and scalable - capable of handling even more data than envisioned at the time of their creation. To this end, in this thesis, the latest Guide to the Software Engineering Body Of Knowledge (SWEBOK Guide) Version 3 [17], hereafter referred to as SWEBOK, is used as a guideline to identify the Knowledge Areas (KA) addressed by the research studies included in this thesis.

1.2 Motivation

Although there have been some literature reviews addressing big data, no review has been provided that focuses specifically on the intersection of software engineering and big data

systems. One research publication discusses the state of the art in architecture and large scale data analysis platforms [10], one is a comprehensive big data survey [33], another describes the related technologies and acquisition and applications of big data [27], one provides an overview of big data applications, tools and opportunities [13], one even lists the different definitions of big data [155].

The guest editor's introduction to one of the issues of the IEEE Software magazine [62] discussed the software engineering challenges in building data-intensive, or big data software systems. Application of software engineering research standards and best practices for the development, maintenance and support of big data systems is a new and emergent field which is only recently starting to see significant and promising research efforts. No comprehensive study reviewing existing software engineering standard practices for enabling development of big data systems was found. There were no research publications found addressing the methods and techniques of important software engineering knowledge areas specifically for big data system development which provided the opportunity for publishing a conference paper [84] and the motivation for this thesis. This thesis tries to bring together the research studies that have applied the knowledge, methods, practices and tools from the software engineering knowledge areas into developing big data systems.

1.3 Contributions

The main goal of this thesis was to look into the existing research on applying software engineering knowledge and guidelines to big data systems. This thesis surveys the state of the art of software engineering specific to big data systems surveying principles and practices for the creation, operation and maintenance of big data systems in order to

uncover insights related to:

1. Software Engineering **K**nowledge **A**reas (KA);
2. Application Domains;
3. Types of big data; and
4. Big data technology trends.

Because of the approach of this thesis to look into big data system development in the context of software engineering knowledge areas, this investigation was able to uncover open research areas and challenges that need to be addressed in the future. To the best of my knowledge, this is the first comprehensive research study that attempts to assess the application of software engineering for big data systems.

1.4 Thesis Outline

Chapter 2 presents the Research Method utilized in this thesis and includes five sections. The first section lists the research questions posed in this thesis, the second section mentions the search strategy employed and the sources used for obtaining the research studies. The third section lists the inclusion and exclusion criteria to separate the relevant research studies from the search results and the fourth, the classification criteria for the identified research studies. Finally, the fifth section explains the data collection process. Chapter 3 provides the demographic data of the studies covered in this thesis, namely the conferences and journals in which the identified studies were published. It shows the trend in the

publications of research studies about software engineering of big data system and lists all the research studies that helped answer the research questions. Chapter 4 discusses the results of this thesis — namely identifying the application domains, the SWEBOK KA each study belongs to, the big data data type referenced and the big data technology used or discussed in the research studies. It further elaborates on the insights obtained from the studies in each KA of the SWEBOK and goes on to list the open research challenges in each KA specific to development of big data systems. The thesis is concluded and future work suggested in Chapter 4.

Chapter 2

Research Method

The literature review performed to obtain research studies relevant to this thesis was conducted based on a hybrid approach that combines automated and manual searches and follows the popular and well established guidelines proposed by Kitchenham et al., for performing a systematic literature review [79]. The systematic literature review process is an excellent method in guiding a review by defining the research questions first, identifying and confirming which of the obtained results are relevant to the research questions and then analyzing the results to obtain important information and observations. The most common reasons for performing a systematic literature review according to [79] thoroughly sync with the motivation for taking the same approach in this thesis — **to summarize the existing research** studies about software engineering knowledge being applied to developing big data systems and **to identify the gaps in the current research** in order to uncover deeper insights and reveal the open research challenges in developing big data systems.

For channeling the search process in order to receive the best results, research questions

were first identified as listed in section "Research Questions". In the section "Search Strategy and Source Materials", the hybrid approach inspired by the guidelines of Kitchenham et al. [79], combining automated and manual searches is discussed. Limiting the search process to get the most accurate results was the next step. In order to select the best and most relevant research studies for this literature review, inclusion and exclusion criteria were defined and are discussed in section "Inclusion and Exclusion Criteria". After the results based on the inclusion and exclusion criteria was obtained, the information to be extracted from each research study were identified on the basis of the classification criteria listed in Section "Classification Criteria".

2.1 Research Questions

The research questions that guided this thesis are as follows:

RQ1. Which application domains have received attention for the development of big data systems?

RQ2. Which SWEBOK KA was studied for development of big data systems?

The above research questions when applied to each research study part of the literature review helped uncover the results that are provided in this thesis. In order to answer the above research questions accurately, classification criteria listed in a subsequent section were developed.

2.2 Search Strategy and Source Materials

The main strategy used in the literature review performed for this thesis was combining automated with manual searches. The main sources for conducting the automated search for research studies relevant to the topic of this thesis were digital academic databases of peer reviewed research literature and repositories of scientific and technical content. For manual searches, research outlets specific to the topic of this thesis, namely, journals and conferences were targeted. Each type of search has its limitations. In case of automated searches, digital databases and repositories get regularly updated with newer and more current research studies so there is always the possibility of recent publications being missed due to the time factor. In addition to this, the results obtained from the searches are dependent on the search strings used and the efficiency and accuracy of their underlying search engines. For manual searches, efficiency of searches performed in research outlets depends on the type, quality and number of journals and conferences covered and may not always be exhaustive.

For the literature review, the following popular digital sources were targeted:

1. Scopus
2. Web of Science
3. IEEE Xplore Digital Library

The search was performed using the "Command" option under the "Advanced Search" section of these databases and repositories. A combination of keywords related to software engineering were selected from the software engineering standard textbook *Software*

Engineering: A Practitioner's Approach [120] to understand which knowledge areas were popular among researchers.

In order to narrow the search results obtained, keywords most suitable to the research questions were selected. In the context of software engineering, the search terms used were architecture, evolution, process, quality, reuse, specification, requirement(s) engineering, design, "domain modeling", testing, verification, validation, maintenance, quality, analysis, framework, process, and patterns.

An example of the pattern of queries used for the literature review is as follows:

"big data" AND (engineering OR requirement OR specification OR design OR architecture OR analysis OR testing OR verification OR validation OR maintenance OR framework OR quality OR design OR evolution OR patterns OR process OR reuse OR "domain modeling")

The idea behind formulating such queries was to search for papers that combined topics about software engineering and big data. An extensive search and selection process to identify a complete set of studies was performed. The search process involved a combination of automatic and manual searching. The title, abstract, introduction and conclusion sections of each research study returned by the search commands were read. Results obtained from the search engines and repositories were manually selected or discarded based on the the inclusion and exclusion criteria listed in the following section.

Popular peer reviewed journals and conferences related to software engineering were not the only ones selected for manual searches. Even journals and conferences that were not directly related to software engineering were covered in order to widen the scope of the

search as much as possible. A list of the journals and conferences covered by the automated and manual search can be found in the next chapter, "Demographic Data".

2.3 Inclusion and Exclusion Criteria

Inclusion and exclusion criteria helped identify the most relevant publications. These criteria helped in eliminating numerous big data studies that did not address the research questions of this review despite being published in high quality conference and journal venues. These criteria were applied for selecting relevant research studies from the results provided by the search queries. A study was selected if it met all the inclusion (IC) criteria and did not fall under any of the exclusion (EC) criteria laid down as follows -

Inclusion criteria:

IC1: Only those research studies relevant to software engineering were included. Studies related to hardware or other type of systems were included only if they also had a software engineering KA as part of their research study.

IC2: Only those research studies which originated from a reputed/cited source were included. Relevant information gathering was the main objective due to which only high quality research was targeted.

IC3: The research studies that discussed at least one software engineering KA in context of big data systems were included. Numerous big data studies from high quality venues were not included despite the fact that they proposed new or improved approaches to big data technology and methods or even positioned themselves in an interesting application domain because they did not reference any KA.

Exclusion criteria:

EC1: Short papers, poster papers, tutorial papers, summary papers, introduction to conference proceedings, workshop summaries, or panel summaries were excluded owing to the fact that they could not contain the necessary information in detail to address the research questions of this review.

EC2: Research studies that did not propose any new, improved or updated techniques/challenges related to software engineering KA phase for big data systems were excluded.

EC3: Working papers and PhD or Masters Thesis were excluded.

EC4: Research studies published in any language other than English were excluded from this study. Multiple papers in Chinese, Turkish, Dutch and other international languages were returned in the manual search but were excluded from this review because of lack of verified English translations.

EC5: Research studies in the field of big data but having no approaches of software engineering being utilized for big data systems were excluded.

2.4 Classification Criteria

The following were the main criteria taken into account when analyzing and categorizing each research study. Under each criteria, multiple categories were identified.

C1. Which application domain does the research study belong to?

This classification criteria is used to identify the application domains of the research studies identified in the literature review. The authors self-identify the application domains in a lot of the research studies.

C2. Which SWEBOK KA is studied in the paper?

The KA of software engineering, according to the SWEBOK Guide Version 3, in which each research study and its results and methods are positioned was discovered from this classification criteria.

C3. What type of data is being used by the authors of the paper?

The type of big data being discussed or used in the research study namely - structured, unstructured, semi-structured or streaming data.

C4. Which big data technology can be identified from the paper?

Some technologies have come to be associated with big data more than others such as data mining, MapReduce, cloud computing, machine learning, and clustering. Some of these technologies and methods already existed before the concept of big data became popular. However, these technologies have seen widespread use in big data applications that has given rise to the idea of viewing them as big data technologies. This classification criteria helped understand the trend of the different big data technologies being used in the research studies identified in the review.

2.5 Data Collection

Approximately 2,000 papers were returned by the digital databases and repositories. Many of these papers were duplicates where the same paper was returned by more that one search

source. Numerous papers related to big data but concerned with hardware and having no details about software engineering KAs were discarded in accordance to the exclusion criteria. Similarly, numerous studies that were related to software for big data systems were discarded due to a lack of software engineering KAs being addressed. Ultimately 152 papers were shortlisted for the literature review. Data from the research studies was analyzed and collected in spreadsheets for easy classification and future referencing. The spreadsheets were updated periodically as analysis continued on the 152 papers covered by this review. This data was again manually reviewed after analysis was completed to eliminate errors that may have cropped up because of the volume of publications and venues covered.

Chapter 3

Demographic Data

The demographic data of the literature review performed in this thesis is described here. It has been divided into three sections - Publication Venues, Publication Trends and Publication Lists.

3.1 Publication Venues

Out of the 152 research studies selected to be part of this review, 95 were conference publications, 52 were journal papers and 5 were chapters from books on conference proceedings and collection of research studies related to big data and software engineering.

Table 3.1 lists all the journals covered as part of this review, the publisher and the Google Scholar h-index to indicate the quality of the journal from which the publication was taken. The journals whose h-index values were not available have been left blank.

Table 3.1: Journals

	Journal	Publisher	h-index
1	ACM SIGMOD Record	ACM	22
2	ACM Transactions on Intelligent Systems and Technology	ACM	37
3	Annals of GIS	Taylor&Francis	11
4	Applied Energy	Elsevier	101
5	Big Data Research	Elsevier	–
6	Cluster Computing	Springer	20
7	Computers, Environment and Urban Systems	Elsevier	29
8	Computers & Graphics	Elsevier	28
9	Data & Knowledge Engineering	Elsevier	25
10	Decision Support Systems	Elsevier	63
11	Energy and Buildings	Elsevier	70
12	Environmental Modelling & Software	Elsevier	57
13	Future Generation Computer Systems	Elsevier	54
14	Geoforum	Elsevier	41
15	IBM Journal of Research and Development	IBM	25
16	IEEE Access	IEEE	22
17	IEEE Computer Graphics and Applications	IEEE	21
18	IEEE Intelligent Systems	IEEE	35
19	IEEE Network	IEEE	–
20	IEEE Software	IEEE	34
21	IEEE Transactions on Big Data	IEEE	–
22	IEEE Transactions on Cloud Computing	IEEE	22
23	IEEE Transactions on Industrial Informatics	IEEE	68
24	IEEE Transactions on Multimedia	IEEE	51
25	IEEE Transactions on Parallel and Distributed Systems	IEEE	76

Continued on next page

Table 3.1 – continued from previous page

	Journal	Publisher	h-index
26	Information Sciences	Elsevier	81
27	International Journal of Digital Earth	Taylor&Francis	22
28	International Journal of Production Economics	Elsevier	76
29	ISPRS Journal of Photogrammetry and Remote Sensing	Elsevier	47
30	IT Professional	IEEE	23
31	Journal of Medical Systems	Springer	43
32	Journal of Selected Topics in Applied Earth Observations and Remote Sensing	IEEE	42
33	Journal of Urban Technology	Taylor&Francis	18
34	Knowledge-Based Systems	Elsevier	60
35	Mobile Networks and Applications	Springer	27
36	Procedia Computer Science	Elsevier	–
37	Procedia CIRP	Elsevier	26
38	Procedia Engineering	Elsevier	36
39	Procedia Technology	Elsevier	24
40	Requirements Engineering	Springer	22
41	Science of Computer Programming	Elsevier	27
42	SIGKDD Explorations Newsletter	ACM	–
43	Simulation Modelling Practice and Theory	Elsevier	32
44	Software: Practice and Experience	Wiley	28
45	Transportation Research Part C: Emerging Technologies	Elsevier	52
46	Tsinghua Science and Technology	Tsinghua University Press	18
47	Utilities Policy	Elsevier	18
48	Wireless Personal Communications	Springer	30

Table 3.2 lists all the conferences covered as part of the literature review in this thesis,

the publisher and the Google Scholar h-index to indicate the quality of the conference from which the publication was taken. The conferences whose h-index values were not available have been left blank.

Table 3.2: Conferences

	Conference	**Publisher**	**h-index**
1	ACM Symposium on Cloud Computing	ACM	38
2	Annual Computer Software and Applications Conference (COMPSACW)	IEEE	19
3	Asian Conference on Intelligent Information and Database Systems (ACIIDS)	Springer	13
4	Asia Pacific Software Engineering Conference	IEEE	13
5	Computer Society Annual Symposium on VLSI	IEEE	14
6	Conference on IT in Business, Industry and Government	IEEE	–
7	Consumer Communication and Networking Conference	IEEE	24
8	Design, Automation and Test in Europe Conference and Exhibition (DATE)	EDA Consortium	39
9	Euromicro Conference on Software Engineering and Advanced Applications	Conference Publishing Services (CPS)	15
10	European Conference on Software Architecture (ECSA)	IEEE	15
11	Hawaii International Conference on System Sciences	IEEE Computer Society Press	39
12	IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing	IEEE/ACM	–
13	IEEE/IFIP Conference on Software Architecture (WICSA)	IEEE	17
14	IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)	IEEE	10

Continued on next page

Table 3.2 – continued from previous page

	Conference	Publisher	h-index
15	IEEE International Conference on Communication Technology (ICCT)	IEEE	11
16	IEEE International Conference on Fuzzy Systems (FUZZ)	IEEE	18
17	IEEE International Conference on Systems, Man and Cybernetics	IEEE	23
18	IEEE Wireless Communications and Networking Conference Workshops (WCNCW)	IEEE	16
19	International Advance Computing Conference	IEEE	17
20	International Black Sea Conference on Communications and Networking (BlackSeaCom)	IEEE	9
21	International Conference on Advanced Cloud and Big Data	IEEE	4
22	International Conference on Automation and Computing	IEEE	9
23	International Conference on Autonomic and Trusted Computing	IEEE	–
24	International Conference on Big Data	IEEE	18
25	International Conference on Big Data Analysis	IEEE	–
26	International Conference on Big Data and Cloud Computing (BdCloud)	IEEE	6
27	International Conference on Big Data Computing Service and Applications	IEEE	–
28	International Conference on Cloud Computing (CLOUD)	IEEE	43
29	International Conference on Cloud Computing and Big Data	IEEE	6
30	International Conference on Cloud Engineering	IEEE	14
31	International Conference on Communication Problem Solving	IEEE	–
32	International Conference on Complex, Intelligent and Software Intensive Systems	IEEE	16

Continued on next page

Table 3.2 – continued from previous page

	Conference	Publisher	h-index
33	International Conference on Computer and Communication Engineering	IEEE	10
34	International Conference of Computer Systems and Applications	IEEE/ACS	–
35	International Conference on Computing and Communication Technologies, Research, Innovation, and Vision for the Future (RIVF)	IEEE	9
36	International Conference on Data and Software Engineering	IEEE	2
37	International Conference on Data Science and Data Intensive Systems	IEEE	2
38	International Conference on High Performance Computing and Communications	IEEE	21
39	International Conference on High Performance Computing and Simulation (HPCS)	IEEE	18
40	International Conference on Information and Automation	IEEE	12
41	International Conference on Information and Communication Systems	IEEE	–
42	International Conference on Innovative Computing Technology	IEEE	7
43	International Conference on Mobile Services	IEEE	9
44	International Conference on P2P, Parallel, Grid, Cloud and Internet Computing	IEEE	–
45	International Conference on Parallel and Distributed Systems	IEEE	19
46	International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)	IEEE	29
47	International Conference on Reliability, Infocom Technologies and Optimization	IEEE	–

Continued on next page

Table 3.2 – continued from previous page

	Conference	Publisher	h-index
48	International Conference on Scalable Computing and Communications	IEEE	–
49	International Conference on Services Computing	IEEE	21
50	International Conference on Smart Computing Workshops	IEEE	–
51	International Conference on Software Engineering	IEEE/ACM	63
52	International Conference on Software Testing, Verification and Validation Workshops (ICSTW)	IEEE	26
53	International Conference on Tools with Artificial Intelligence	IEEE	17
54	International Conference on Transparent Optical Networks	IEEE	18
55	International Conference on Ubiquitous and Future Networks (ICUFN)	IEEE Communications Society	14
56	International Conference on Ubiquitous Intelligence and Computing	IEEE	–
57	International Congress on Big Data	IEEE	14
58	International Enterprise Distributed Object Computing Conference Workshops and Demonstrations	IEEE	–
59	International Symposium on Low Power Electronics and Design	IEEE/ACM	24
60	International Symposium on Software Reliability Engineering	IEEE	19
61	International Workshop on Big Data Software Engineering - Co-located with International Conference of Software Engineering	IEEE	–
62	International Workshop on Data Mining with Industrial Applications	IEEE	–
63	International Workshop on Emerging software as a Service and Analytics	SciTePress	–

Continued on next page

Table 3.2 – continued from previous page

	Conference	Publisher	h-index
64	International Workshop on Modeling in Software Engineering	IEEE Press	–
65	Mediterranean Conference on Embedded Computing	IEEE	7
66	World Congress on Information and Communication Technologies (WICT)	IEEE	15
67	World Congress on Services	IEEE Computer Society Press	19

3.2 Publication Trends

The number of research studies which applied software engineering KAs related to development of big data systems were plotted into a graph to show the trend of publications over the years in Figure 3.1. The first study relevant to the research questions posed in this thesis dates back to 2007. The number of studies from 2016 are fewer than expected most likely due to the fact that the search for research studies had been terminated by the end of September 2016 and there may have been more research studies which only became available in the digital databases/repositories after that. Another possible reason could be the time it takes from research studies being accepted for publishing in journals or at conferences to them actually becoming available online in the databases/repositories.

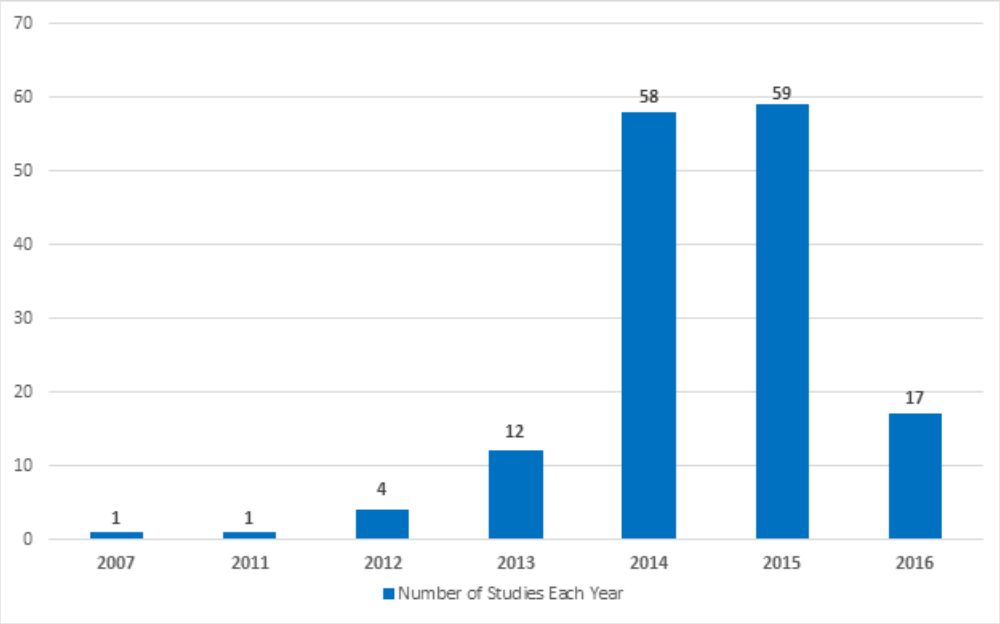

Figure 3.1: Big Data Research Studies Over The Years

3.3 Publication List

Table 3.3 lists the 152 research studies identified by the literature review conducted as part of this thesis.

Table 3.3: Complete List of Research Studies

Author(s)	Title	Year	Reference
Addo et al.	A Reference Architecture for Social Media Intelligence Applications in the Cloud	2015	[2]
Agrawal et al.	A Layer Based Architecture for Provenance in Big Data	2014	[3]
Goya et al.	The Use of Distributed Processing and Cloud Computing in Agricultural Decision-Making Support Systems	2014	[67]
Akmal, Allison, and González-Vélez	Assembling Cloud-based Geographic Information Systems: A Pragmatic Approach using Off-the-Shelf Components	2015	[4]
Al Zamil and Samarah	The Application of Semantic-based Classification on Big Data	2014	[5]
Alder and Hostetler	Web based Visualization of Large Climate Data Sets	2015	[6]
Anderson et al.	EPIC-OSM: A Software Framework for OpenStreetMap Data Analytics	2016	[7]
Anderson	Embrace the Challenges: Software Engineering in a Big Data World	2015	[8]
Bazargani, Brinkley, and Tabrizi	Implementing Conceptual Search Capability in a Cloud-Based Feed Aggregator	2013	[9]
Begoli and Horey	Design Principles for Effective Knowledge Discovery from Big Data	2012	[11]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Belo, Cuzzocrea, and Oliveira	Modeling and Supporting ETL Processes via a Pattern-Oriented, Task-Reusable Framework	2014	[12]
Bersani et al.	Continuous Architecting of Stream-Based Systems	2016	[14]
Bonomi et al.	Fog Computing: A Platform for Internet of Things and Analytics	2014	[16]
Breuker	Towards Model-Driven Engineering for Big Data Analytics–An Exploratory Analysis of Domain-Specific Languages for Machine Learning	2014	[18]
Camilli	Formal Verification Problems in a Big Data World: Towards a Mighty Synergy	2014	[19]
Cao et al.	A Scalable Framework for Spatiotemporal Analysis of Location-based Social Media Data	2015	[20]
Casale et al.	DICE: Quality-Driven Development of Data-intensive Cloud Applications	2015	[21]
Cecchinel et al.	An Architecture to Support the Collection of Big Data in the Internet of Things	2014	[23]
Cecchinel, Mosser, and Collet	Software Development Support for Shared Sensing Infrastructures: A Generative and Dynamic Approach	2014	[22]
Chang and Ramachandran	A Proposed Case for the Cloud Software Engineering in Security	2014	[25]
Chebotko, Kashlev, and Lu	A Big Data Modeling Methodology for Apache Cassandra	2015	[26]
Chen, Wu, and Wang	The Evolvement of Big Data Systems: From the Perspective of an Information Security Application	2015	[28]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Chen, Kazman, and Haziyev	Agile Big Data Analytics for Web-based Systems: An Architecture-centric Approach	2016	[29]
Chen, Kazman, and Haziyev	Strategic Prototyping for Developing Big Data Systems	2016	[30]
Chen et al.	Big Data System Development: An Embedded Case Study with a Global Outsourcing Firm	2015	[31]
Chen et al.	WaaS: Wisdom as a Service	2014	[32]
Cheng et al.	Building a Big Data Platform for Smart Cities: Experience and Lessons from Santander	2015	[34]
Cuesta, Martínez-Prieto, and Fernández	Towards an Architecture for Managing Big Semantic Data in Real-Time	2013	[35]
Dajda and Dobrowolski	Architecture Dedicated to Data Integration	2015	[36]
Demirkan and Delen	Leveraging the Capabilities of Service-oriented Decision Support Systems: Putting Analytics and Big Data in Cloud	2013	[37]
Deng, Gao, and Vuppalapati	Building a Big Data Analytics Service Framework for Mobile Advertising and Marketing	2015	[38]
Deng and Di	Building Open Environments to Meet Big Data Challenges in Earth Sciences	2014	[39]
Desai and Nagegowda	Advanced Control Distributed Processing Architecture (ACDPA) using SDN and Hadoop for Identifying the Flow Characteristics and Setting the Quality of Service (QoS) in the Network	2015	[40]
Ding, Zhang, and Hu	A Framework for Ensuring the Quality of a Big Data Service	2016	[42]
Divn et al.	Towards a Data Processing Architecture for the Weather Radar of the INTA Anguil	2015	[43]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Dobre and Xhafa	Intelligent Services for Big Data Science	2014	[44]
D'Oca and Hong	Occupancy Schedules Learning Process through a Data Mining Framework	2015	[45]
Douglas	An Open Framework for Dynamic Big-Data-Driven Application Systems (DBDDAS) Development	2014	[46]
Doyle et al.	The EMBERS Architecture for Streaming Predictive Analytics	2014	[47]
Durham, Rosen, and Harrison	A Model Architecture for Big Data Applications using Relational Databases	2014	[48]
Dutta and Bose	Managing a Big Data project: The case of Ramco Cements Limited	2015	[49]
Dutta et al.	Development of an Intelligent Environmental Knowledge System for Sustainable Agricultural Decision Support	2014	[50]
Elagib et al.	Big Data Analysis Solutions Using MapReduce Framework	2014	[51]
Eridaputra, Hendradjaya, and Sunindyo	Modeling the Requirements for Big Data Application using Goal Oriented Approach	2014	[52]
Esposito et al.	A Knowledge-based Platform for Big Data Analytics Based on Publish/Subscribe Services and Stream Processing	2015	[53]
Fadiya, Saydam, and Zira	Advancing Big Data for Humanitarian Needs	2014	[54]
Fang et al.	An Integrated System for Regional Environmental Monitoring and Management Based on Internet of Things	2014	[55]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Forkan et al.	BDCaM: Big Data for Context-aware Monitoring - A Personalized Knowledge Discovery Framework for Assisted Healthcare	2015	[56]
Gai et al.	Electronic Health Record Error Prevention Approach Using Ontology in Big Data	2015	[58]
Giachetta	A Framework for Processing Large Scale Geospatial and Remote Sensing Data in MapReduce Environment	2015	[59]
Girardi and Marinho	A Domain Model of Web Recommender Systems based on Usage Mining and Collaborative Filtering	2007	[60]
Gökalp, Koçyigit, and Eren	A Cloud Based Architecture for Distributed Real Time Processing of Continuous Queries	2015	[61]
Gorton and Klein	Distribution, Data, Deployment: Software Architecture Convergence in Big Data Systems	2015	[63]
Gorton, Klein, and Nurgaliev	Architecture Knowledge for Evaluating Scalable Databases	2015	[64]
Gousios, Safaric, and Visser	Streaming Software Analytics	2016	[66]
Guerriero et al.	Towards a Model-driven Design Tool for Big Data Architectures	2016	[68]
Rehman, Liew, and Wah	UniMiner: Towards a Unified Framework for Data Mining	2014	[124]
Holley, Sivakumar, and Kannan	Enrichment Patterns for Big Data	2014	[69]
Huai et al.	DOT: A Matrix Model for Analyzing, Optimizing and Deploying Software for Big Data Analytics in Distributed Systems	2011	[70]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Huang and Xu	A Data-Driven Framework for Archiving and Exploring Social Media Data	2014	[71]
Immonen, Pääkkönen, and Ovaska	Evaluating the Quality of Social Media Data in Big Data Architecture	2015	[72]
Jararweh et al.	CloudExp: A Comprehensive Cloud Computing Experimental Framework	2014	[74]
Jutla, Bodorik, and Ali	Engineering Privacy for Big Data Apps with the Unified Modeling Language	2013	[75]
Kanoun et al.	Low Power and Scalable Many-Core Architecture for Big-Data Stream Computing	2014	[76]
Kaur and Rani	A Smart Polyglot Solution for Big Data in Healthcare	2015	[77]
Klein et al.	Model-Driven Observability for Big Data Storage	2016	[81]
Klein et al.	A Reference Architecture for Big Data Systems in the National Security Domain	2016	[80]
Kousiouris, Vafiadis, and Varvarigou	Enabling Proactive Data Management in Virtualized Hadoop Clusters based on Predicted Data Activity Patterns	2013	[82]
Krämer and Senner	A Modular Software Architecture for Processing of Big Geospatial Data in the Cloud	2015	[83]
Kumara et al.	Ontology-Based Workflow Generation for Intelligent Big Data Analytics	2015	[85]
Kushiro, Matsuda, and Takahara	Model Oriented System Design on Big-Data	2014	[86]
Kwoczek et al.	An Architecture to Process Massive Vehicular Traffic Data	2015	[87]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Lamba and Dubey	Analysis of Requirements for Big Data Adoption to Maximize IT Business Value	2015	[88]
Ledur et al.	Towards a Domain-Specific Language for Geospatial Data Visualization Maps with Big Data Sets	2015	[90]
Li, Grechanik, and Poshyvanyk	Sanitizing and Minimizing Databases for Software Application Test Outsourcing	2014	[91]
Li et al.	Research and Application of One-Key Publishing Technologies for Meteorological Service Products	2016	[93]
Li, Huang, and Chen	Breeze Graph Grammar: A Graph Grammar Approach for Modeling the Software Architecture of Big Data-oriented Software Systems	2014	[92]
Li et al.	Improving Rail Network Velocity: A Machine Learning Approach to Predictive Maintenance	2014	[94]
Li et al.	A Scalable Big Data Test Framework	2015	[95]
Li et al.	Breaking the Boundary for Whole-System Performance Optimization of Big Data	2013	[96]
Liu	Research of Performance Test Technology for Big Data Applications	2014	[98]
Madhavji, Miranskyy, and Kontogiannis	Big Picture of Big Data Software Engineering: With Example Research Challenges	2015	[99]
Marchal et al.	A Big Data Architecture for Large Scale Security Monitoring	2014	[100]
Marín-Ortega et al.	ELTA: New Approach in Designing Business Intelligence Solutions in Era of Big Data	2014	[101]
Martínez-Prieto et al.	The SOLID Architecture for Real-Time Management of Big Semantic Data	2015	[103]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Meng et al.	KASR: A Keyword-Aware Service Recommendation Method on MapReduce for Big Data Applications	2014	[104]
Mishra, Lin, and Chang	A Cognitive Oriented Framework for IoT Big-Data Management Prospective	2014	[105]
Moguel et al.	Multilayer Big Data Architecture for Remote Sensing in Eolic Parks	2015	[106]
Müller, Bernard, and Kadner	Moving Code – Sharing Geoprocessing Logic on the Web	2013	[107]
Mytilinis et al.	I/O Performance Modeling for Big Data Applications over Cloud Infrastructures	2015	[108]
Naseer, Alkazemi, and Waraich	A Big Data Approach for Proactive Healthcare Monitoring of Chronic Patients	2016	[109]
Ning et al.	Research on Warship Communication Operation and Maintenance Management Based on Big Data	2014	[110]
Noorwali, Arruda, and Madhavji	Understanding Quality Requirements in the Context of Big Data Systems	2016	[111]
Nowling and Vyas	A Domain-Driven, Generative Data Model for Big Pet Store	2014	[112]
Ochian et al.	Big Data Search for Environmental Telemetry	2014	[113]
Ordonez et al.	Extending ER Models to Capture Database Transformations to Build Data Sets for Data Mining	2014	[114]
Pääkkönen and Pakkala	Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems	2015	[116]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Patton et al.	SemantEco: A Semantically Powered Modular Architecture for Integrating Distributed Environmental and Ecological Data	2014	[117]
Pelekis, Theodoridis, and Janssens	On the Management and Analysis of Our LifeSteps	2014	[118]
Pham et al.	An Adaptable Framework to Deploy Complex Applications onto Multi-Cloud Platforms	2015	[119]
Rahmes et al.	Multi-Disciplinary Ontological Geo-Analytical Incident Modeling	2015	[121]
Rajbhoj, Kulkarni, and Bellarykar	Early Experience with Model-Driven Development of MapReduce based Big Data Application	2014	[122]
Restrepo-Arango, Henao-Chaparro, and Jiménez-Guarín	Using the Web to Monitor a Customized Unified Financial Portfolio	2012	[125]
R.Šendelj et al.	Towards Semantically Enabled Development of Service-Oriented Architectures for Integration of Socio-Medical Data	2016	[127]
Rysavy, Bromley, and Daggett	DIVE: A Graph-based Visual-Analytics Framework for Big Data	2014	[128]
S., Lee, and Lee	Trajectory Patterns Mining Towards Lifecare Provisioning	2014	[129]
Samuel et al.	A Framework for Composition and Enforcement of Privacy-Aware and Context-Driven Authorization Mechanism for Multimedia Big Data	2015	[130]
Sciacca et al.	Towards a Big Data Exploration Framework for Astronomical Archives	2014	[131]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Shah, Rabhi, and Ray	Investigating an Ontology-based Approach for Big Data Analysis of Inter-dependent Medical and Oral Health Conditions	2015	[132]
Shen	A Pervasive Framework for Real-Time Activity Patterns of Mobile Users	2015	[133]
Shi et al.	Context-based Ontology-driven Recommendation Strategies for Tourism in Ubiquitous Computing	2014	[134]
Shi et al.	Improving Power Grid Monitoring Data Quality: An Efficient Machine Learning Framework for Missing Data Prediction	2015	[135]
Shukla and Sadashivappa	A Distributed Randomization Framework for Privacy Preservation in Big Data	2014	[136]
Singh and Liu	A Cloud Service Architecture for Analyzing Big Monitoring Data	2016	[137]
Sinnott, Morandini, and Wu	SMASH: A Cloud-Based Architecture for Big Data Processing and Visualization of Traffic Data	2015	[138]
Sneed and Erdoes	Testing Big Data (Assuring the Quality of Large Databases)	2015	[139]
Suciu et al.	Cloud Computing for Extracting Price Knowledge from Big Data	2015	[140]
Sun et al.	iCARE: A Framework for Big Data-based Banking Customer Analytics	2014	[141]
Sun et al.	Store, Schedule and Switch - A New Data Delivery Model in the Big Data Era	2013	[142]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Taneja et al.	Linked Enterprise Data Model and Its Use in Real Time Analytics and Context-Driven Data Discovery	2015	[144]
Tao et al.	Facilitating Twitter Data Analytics: Platform, Language and Functionality	2014	[145]
Tesfagiorgish and JunYi	Big Data Transformation Testing Based on Data Reverse Engineering	2015	[146]
Thangaraj and Anuradha	State of Art in Testing for Big Data	2015	[147]
Tracey and Sreenan	A Holistic Architecture for the Internet of Things, Sensing Services and Big Data	2013	[148]
Truong and Dustdar	Sustainability Data and Analytics in Cloud-Based M2M Systems	2014	[149]
Vanauer, Bhle, and Hellingrath	Guiding the Introduction of Big Data in Organizations: A Methodology with Business- and Data-Driven Ideation and Enterprise Architecture Management-Based Implementation	2015	[150]
Villari et al.	AllJoyn Lambda: An Architecture for the Management of Smart Environments in IoT	2014	[151]
Vinay et al.	Cloud Based Big Data Analytics Framework for Face Recognition in Social Networks Using Machine Learning	2015	[152]
Wang, Li, and Zhou	SODA: Software Defined FPGA Based Accelerators for Big Data	2015	[153]
Wang et al.	Smart Traffic Cloud: An Infrastructure for Traffic Applications	2012	[154]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Westerlund et al.	A Generalized Scalable Software Architecture for Analyzing Temporally Structured Big Data in the Cloud	2014	[156]
Wilder, Smith, and Mockus	Exploring a Framework for Identity and Attribute Linking Across Heterogeneous Data Systems	2016	[157]
Wu et al.	Nonparametric Discovery of Contexts and Preferences in Smart Home Environments	2015	[159]
Wu et al.	Building Pipelines for Heterogeneous Execution Environments for Big Data Processing	2016	[160]
Wu, Zhang, and Lim	A Cooperative Sensing and Mining System for Transportation Activity Survey	2014	[161]
Xinhua et al.	Big Data-as-a-Service: Definition and architecture	2013	[162]
Xu, Li, and Butt	GERBIL: MPI + YARN	2015	[163]
Xu et al.	Knowle: A Semantic Link Network Based System for Organizing Large Scale Online News Events	2015	[164]
Yang and M.	A Big-Data Processing Framework for Uncertainties in Transportation Data	2015	[165]
Yang et al.	An Automatic Discovery Framework of Cross-Source Data Inconsistency for Web Big Data	2015	[166]
Yang-Turner, Lau, and Dimitrova	A Model-Driven Prototype Evaluation to Elicit Requirements for a Sensemaking Support Tool	2012	[167]
Yao et al.	Design and Development of a Medical Big Data Processing System based on Hadoop	2015	[168]
Yim	Norming to Performing: Failure Analysis and Deployment Automation of Big Data Software Developed by Highly Iterative Models	2014	[169]

Continued on next page

Table 3.3 – continued from previous page

Author(s)	Title	Year	Reference
Yongpisanpop, Hata, and Matsumoto	Bugarium: 3D Interaction for Supporting Large-Scale Bug Repositories Analysis	2014	[170]
Zhang et al.	A Task-Level Adaptive MapReduce Framework for Real-Time Streaming Data in Healthcare Applications	2015	[171]
Zhang	Designing Big Data Driven Cyber Physical Systems based on AADL	2014	[173]
Zhang	A Framework to Model Big Data Driven Complex Cyber Physical Control Systems	2014	[172]
Zhang et al.	TerraFly GeoCloud: An Online Spatial Data Analysis and Visualization System	2015	[174]
Zhang et al.	A Deep-Intelligence Framework for Online Video Processing	2016	[175]
Zhou et al.	An Empirical Study on Quality Issues of Production Big Data Platform	2015	[176]
Zimmermann et al.	Adaptable Enterprise Architectures for Software Evolution of SmartLife Ecosystems	2014	[178]
Zimmermann et al.	Towards Service-Oriented Enterprise Architectures for Big Data Applications in the Cloud	2013	[177]

Chapter 4

Results

The results of this thesis are divided into three sections. The first section of the results, obtained upon applying classification criterion C1 mentioned in the Classification Criteria section of the Chapter Research Method, details the applications domains that were observed from all the research studies covered by the literature review and lists the potential application domains that could benefit from big data systems. The second section, from applying classification criterion C2, is a detailed analysis of each software engineering KA from SWEBOK addressed in the research studies covered by the literature review and observations about the open research challenges in each KA with respect to big data systems. The third section contains the information gathered from classification criteria C3 and C4 - the big data types and the big data technologies referenced or used in each research study to direct attention towards the trends and popularity of each technology/method.

Each section contains some tables and graphs to illustrate the number of research studies that were addressed in each section.

4.1 Application Domains

Big data has the potential to transform numerous fields that can deploy technology. The fields in which big data systems are being used are referred to as "Application Domains" in this thesis. The most important benefit of using big data systems in this age of "information explosion" is to realize the impact of large volumes of recurring data and even analyze the same in conjunction with previously available or historical smaller data points. Any application domain and not just businesses, that recurrently generate data can deploy big data systems to discover patterns of activity and mine these patterns to glean relevant and sometimes even heretofore unknown information about the application domain.

The first research question **RQ1** and the research studies obtained from the classification criterion **C1** from the literature review in this thesis provides the information that is discussed and analyzed in this section.

Figure 4.1 provides a pie chart that shows the sectors for the different application domains discovered from the literature review.

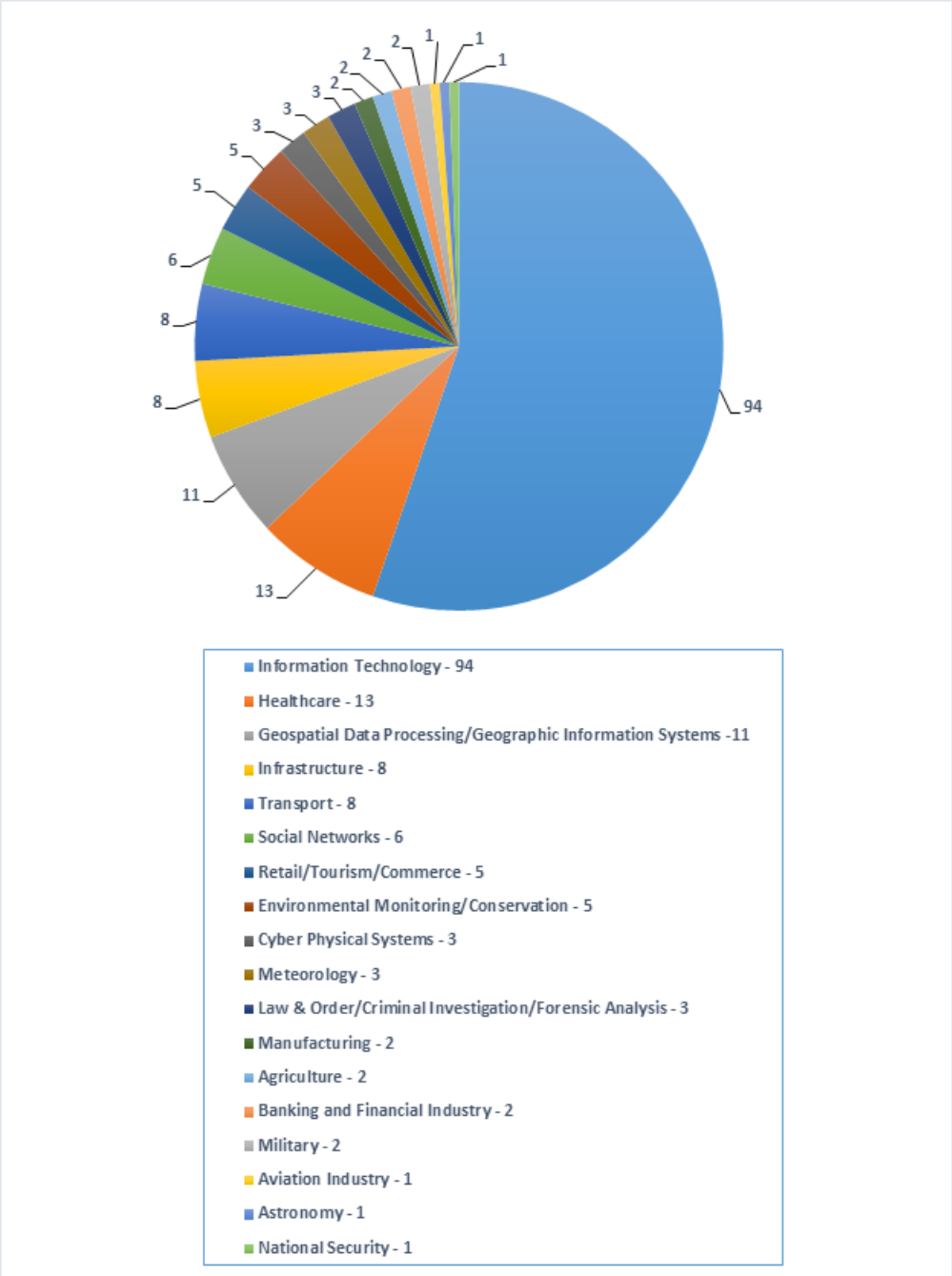

Figure 4.1: Application Domains

4.1.1 Analysis

Analysis of the application domains discovered in this thesis starts with a discussion on the first research question posed in the literature review.

[**RQ1**] Which application domains have received attention for the development of big data systems?

Figure 4.1 and Table 4.1 show the distribution of the research studies part of this thesis and the application domains that were identified from each. Table 4.1 lists all the research studies that were classified under the identified application domains.

Research studies or "Papers", as they will be called when used in the tables of this thesis, proposing new methods or customized versions of existing technology which do not specify any particular application domain for deployment, such as healthcare, military, or infrastructure, were classified into the "Information Technology" category. Out of the 152 research studies selected through the literature review, the number of research studies dealing with this category was 94, shown in the column "Count" of Table 4.1. Nearly 61% of the analyzed papers focused on the Information Technology application domain, which goes to show that most of the research studies focused on topics that would directly affect the world of computing. The primary observation was that researchers focused on improving existing technologies to suit current requirements better.

The Healthcare application domain was the second runner-up with 13 research studies. Some of the research studies in this application domain dealt with Healthcare Information Systems (HIS), Electronic Medical Record (EMR) systems or Electronic Health Record (EHR) systems [58] [77]. This does not come as a surprise because the healthcare industry is a source of huge amounts of data sourced from the electronic medical records of patients.

Data from hospitals, clinics, medical governing bodies and even insurance providers can be mined to study disease occurrence rates, patterns and susceptibility trends. Analyzing medical data can also help in developing innovative treatment methods, customize more effective and economical treatment plans, or even help healthcare professionals in dispensing medication to groups of patients suffering from similar afflictions that have identical medical history of symptoms and reactions.

Interestingly, Geospatial Data Processing/Geographic Information Systems application domain was a close runner-up to Healthcare with 12 research studies. Geographic Information Systems (GIS) have attracted the attention of big data researchers because of the widespread use of smartphones and the breakthroughs made in digital cartography, also called digital mapping - the process of collecting geographic or location data and formatting it into a virtual image. This can be seen in the rise in popularity and subsequent ubiquity of applications like Google Maps and Apple Maps and the businesses that leverage these into producing a completely new product like transportation network companies and local businesses or restaurant recommendation websites.

4.1.2 Open Research Areas

Application domains like Banking and Financial industry are commonly believed to be data rich. The banking industry has access to the monetary blueprint of the world. The tremendous amount of transaction data that commercial banks handle on a daily basis can be used to understand the spending patterns of its customers better. Credit card usage, mobile banking application usage patterns, mortgage and credit history of customers can provide greater insights into the needs of and risks to its customers, and help tailor their products accordingly enhancing customer experience and satisfaction. One example

Table 4.1: Application Domain Categories

Application Domain	Papers	Count
Information Technology	[3], [5], [8], [9], [11], [12], [14], [16, 18, 19], [21], [23], [24, 25, 26], [28, 29, 30, 31, 32], [36], [37], [40], [46, 47, 48], [51, 52, 53, 54], [60], [61], [63, 64, 66, 68, 69, 70, 124], [74, 75, 76], [81], [82, 83, 85], [88], [91], [92], [95], [96], [98, 99, 100, 101], [103], [104], [105], [108], [111, 112, 113, 114, 116], [119], [122], [130], [133], [136], [137], [139], [142], [144], [146], [147, 148, 149], [150, 151, 152, 153], [156], [157, 159, 160], [162], [163], [166], [167], [170], [175], [176, 177, 178]	94
Healthcare	[42], [56], [58], [77], [109], [127, 128, 129], [132], [164], [168], [171], [174]	13
Geospatial Data Processing/Geographic Information Systems	[4], [6], [7], [20], [39], [55], [59], [83], [90], [107], [174]	11
Transport	[44], [87], [94], [118], [138], [161], [165], [175]	8
Infrastructure	[23], [34], [44], [45], [118], [135], [154], [161]	8
Social Networks	[2], [20], [71], [72], [145], [152]	6
Retail/Tourism/Commerce	[38], [134], [140], [174], [175]	5
Environmental Monitoring/ Conservation	[55], [83], [106], [117], [144]	5
Meteorology	[6], [43], [93]	3
Cyber Physical Systems	[22], [173], [172]	3
Law and Order/Criminal Investigation/ Forensic Analysis	[36], [80], [121]	3
Agriculture	[67], [50]	2
Banking and Financial Industry	[125], [141]	2
Manufacturing	[49], [86]	2
Military	[80], [110]	2
Aviation Industry	[173]	1
Astronomy	[131]	1
National Security	[80]	1

is detecting fraudulent transactions on credit cards where big data could avoid a lot of problems. Banking software would need to perform extremely fast processing of the bank's customer data, history of fraudulent transactions archived by banks, patterns to detect unusual credit card activity based on factors like irregular transaction amounts, unlikely locations and unregistered retailers. All this data would have to be mined and business information generated to alert customers in real time to minimize losses faced by the customer as well as the bank itself. This would require speedy in-database analytical processes. Such business cases are tailor made for deployment of big data systems in order to ensure customer satisfaction.

However, development of such elaborate software systems using big data which source from different types of datasets mostly housed in structurally different database systems would not be an easy task. Another concern that slows the adoption of big data in the banking sector is privacy concern for customer data. The development of big data systems that would not infringe or violate the privacy of customers would only be successful if all the requirements are thoroughly elicited, analyzed, and verified before system design and architectures are created. In such situations, the software engineering standard practices would be excellent guides for creating robust and scalable software systems. Because big data systems will potentially have a huge role to play in the operations of banks, there should be more research in the software engineering KAs for building better big data systems for the banking industry. A potential approach would be applying software engineering KAs to develop big data systems with special attention to privacy requirements [75] that could be customized specifically for development of banking software.

The financial industry including the stock market, equities, bonds and investment banking has also witnessed proliferation of big data technology [125]. Another promising ap-

proach would be deploying big data systems in stock exchanges around the world in order to detect illegal or insider trading trends to prevent stock market crashes which lead to huge losses for businesses, trading entities and individual traders.

Data rich domains like infrastructure, transport and manufacturing have only seen scant research in applying software engineering KAs in developing big data systems. Metropolitan authorities responsible for traffic management and building infrastructural facilities can use current as well as historical data to build smart cities and green buildings that use minimum amounts of energy and water for heating and maintenance.

The aviation industry and the military also have huge potential for using big data for better performance and maintenance of equipment, aircrafts and vehicles. The data generated by aircrafts and helicopters when mined can provide inputs for better aerodynamics, optimal fuel consumption and current functioning status and performance of older machine parts and aircraft carriers.

Global warming is causing serious damage to our environment and wildlife and meteorologists can use big data from the global weather sensors to make more accurate weather predictions and generate timely natural disaster alerts. Environmental Conservation and Monitoring can also be a big beneficiary of big data systems by generating data from tracking of migratory animals and birds, air quality and pollution measurements, population metric and analysis of flora and fauna species in ecological spheres, and for studying biodiversity in wildlife endangered zones.

The telecommunications industry is an extremely competitive industry and one that has a huge potential for big data systems. It unfortunately did not show up in any research study covered in this thesis. Cable and telecommunication service providers can use big

data from their existing customers to create lucrative offers to attract customers from their rival operators. Analytics of data about telephone traffic and services frequently used by customers can help in improving current services and creating new sources of revenue all the while continuing to keep costs under control.

4.2 Software Engineering KA

The Guide to the Software Engineering Body Of Knowledge Version 3 or SWEBOK [17], was used as a guideline to identify important software engineering Knowledge Areas (KA). These KAs were then employed to classify the research studies identified in this thesis. The second research question **RQ2** and the data obtained from the classification criterion **C2** for the review in this thesis provides the information that is discussed and analyzed in this section. Each KA is discussed under separate subsections. The KAs from SWEBOK could be considered as macro categories for classification of the research studies. These macro categories were further broken down into specific topics as they appear in SWEBOK to provide more granular observations in each KA. These specific topics under each KA are referred to as micro categories. Each micro category used to categorize the research studies are listed in the KA subsections. The second research question drives the information presented in this section and the subsections that follow. Each subsection analyses some of the representational research studies covered under it and the importance of the micro categories that did not have any research studies classified under them.

In SWEBOK, there are a total of 15 KAs. Only 12 of these KAs are used in this thesis because the remaining three were foundational subjects on computing, mathematics and engineering and not within the scope of the research questions and goal of this thesis. The

KAs that were identified from SWEBOK to categorize the research studies are:

1. Software Requirements
2. Software Design
3. Software Construction
4. Software Testing
5. Software Maintenance
6. Software Configuration Management
7. Software Engineering Management
8. Software Engineering Process
9. Software Engineering Models and Methods
10. Software Quality
11. Software Engineering Professional Practice
12. Software Engineering Economics

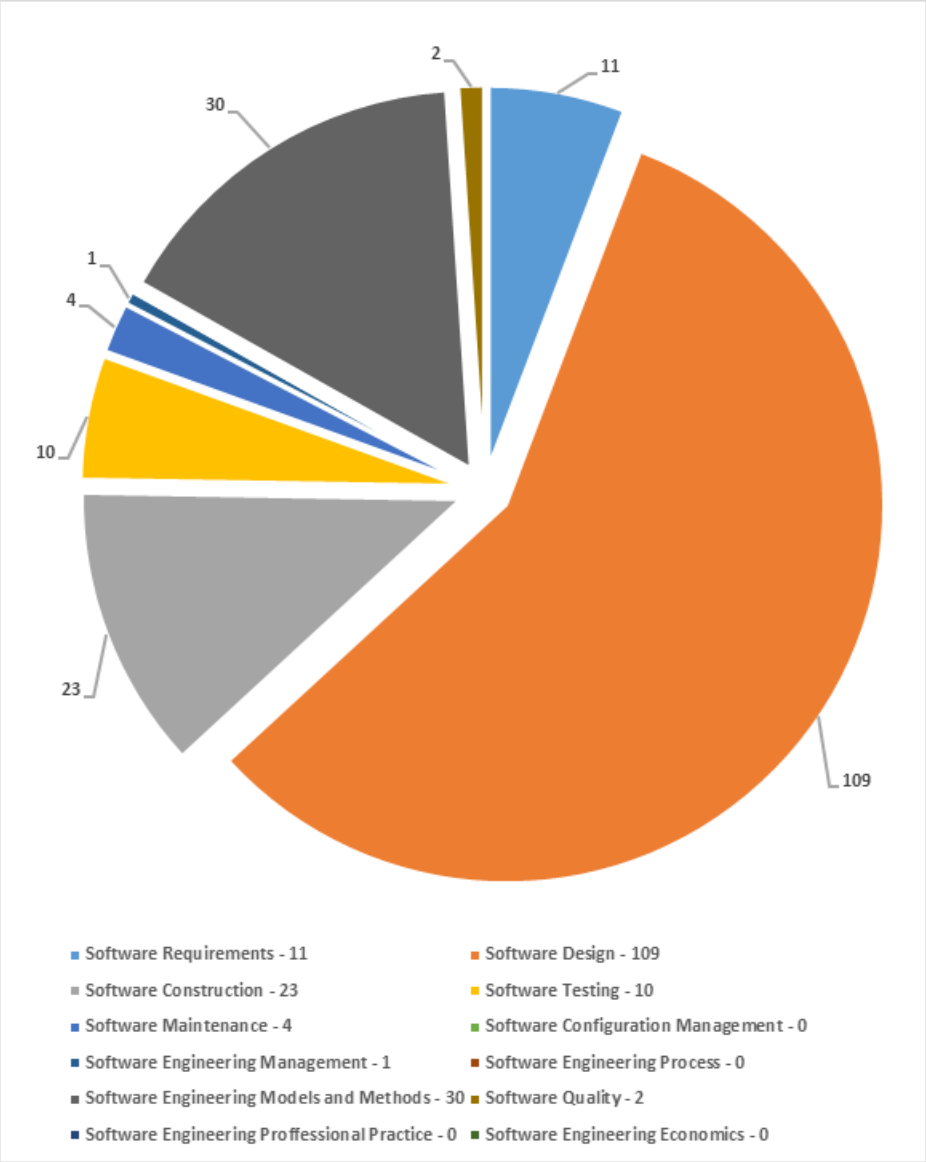

Figure 4.2: SWEBOK KAs

[**RQ2**] Which SWEBOK KA was studied for development of big data systems?

Figure 4.2 illustrates the breakdown of the research studies according to the KA ad-

Table 4.2: SWEBOK KAs

SWEBOK Categories	Papers	Count
Software Requirements	[23], [37], [52], [60], [75], [88], [99], [111], [148], [167], [173]	11
Software Design	[2], [3], [67], [4], [6], [7], [8], [9], [11], [14], [16], [20], [23], [25], [26], [28], [29], [31], [32], [34], [35], [36], [37], [38], [40], [43], [44], [45],[46], [47], [48], [50], [51], [53],[54], [55], [56], [59], [61], [63], [64], [68], [124], [69], [71], [72], [74], [76], [77], [81], [80], [82], [83], [85], [86], [87], [90], [93], [99], [100], [101], [103], [104], [105], [106], [107], [109], [110], [114], [116], [117], [119], [125], [127], [128], [129], [130], [131], [132], [133], [134], [136], [137], [138], [141], [145], [148], [149], [151], [152], [153], [154], [156], [157], [159], [160], [161], [162], [163], [165], [168], [170], [171], [173], [172], [174], [175], [177]	109
Software Construction	[2], [67], [4], [11], [23], [34], [36], [38], [50], [54], [76], [90], [100], [103], [124], [129], [137], [148], [153], [160], [161], [163], [168]	23
Software Testing	[42], [91], [95], [98], [139], [146], [147], [169]	8
Software Maintenance	[96], [169], [170], [176]	4
Software Configuration Management		–
Software Engineering Management	[49]	1
Software Engineering Process		–
Software Engineering Models and Methods	[5], [12], [18], [19], [22], [26], [30], [58], [66], [70], [75], [92], [94], [108], [112], [113], [118], [121], [122], [130], [135], [140], [141], [142], [144], [150], [164], [166], [172], [178]	30
Software Quality	[21], [72]	2
Software Engineering Professional Practice		–
Software Engineering Economics		–

dressed by each in the form of a pie chart. Out of the 12 KAs that were identified for classification, Software Design KA was the most referenced. Majority of the research studies were positioned in the Software Design KA. Approximately 70% of the research studies dealt with architectural design decisions, architectural implementations and frameworks. No research study referenced KAs such as Software Configuration Management, Software Engineering Process, Software Engineering Professional Practice and Software Engineering Economics specifically in context of big data systems.

Table 4.2 provides the research studies that belong to each KA and the total counts of each. The KA which saw no research studies were left blank.

4.2.1 Software Requirements

According to SWEBOK, the Software Requirements KA is concerned with the elicitation, analysis, specification and validation of software requirements as well as the management of requirements during the entire life cycle of the software product [17].

Out of the 152 research studies identified by the literature review in this thesis, 11 of them proposed approaches, methods and practices which were categorized under Software Requirements KA.

The micro categories according to SWEBOK for the Software Requirements KA were:

1. Requirements Process
2. Requirements Elicitation
3. Requirements Analysis

4. Requirements Specification

5. Software Requirements Tools

Table 4.3 lists all the research studies that were identified under the Software Requirements KA micro categories.

Table 4.3: Software Requirements Micro Categories

Software Requirements Micro Categories	**Papers**	**Count**
Requirements Process		–
Requirements Elicitation	[23], [37], [52], [99], [148], [167]	6
Requirements Analysis	[75], [88], [111]	3
Requirements Specification	[60], [173]	2
Requirements Validation		–
Software Requirements Tools		–

4.2.1.1 Analysis

Of the eleven research studies identified, six were about Requirements Elicitation, three about Requirements Analysis and two studied Requirements Specification.

Eridaputra et al., proposed a generic requirement model for requirements elicitation of big data applications [52]. Their approach to building the model was by applying GORE (Goal Oriented Requirements Engineering). The authors state the advantages of using an approach involving GORE as opposed to traditional requirements engineering methods is that the requirements as well as the goal can be modeled. This is because of the fact that

GORE is motivated by the question "why the system should do it" rather than "what the system should do" which means that it focuses on the purpose behind designing and building a system. The authors list the general requirements for big data applications and discuss requirement modeling using the modeling language - i* framework and Knowledge Acquisition autOmated System (KAOS) in [52]. The general requirements for big data applications as listed by the authors were:

1. Huge database capacity
2. Fine database performance
3. Quality and structure of data
4. Guaranteed privacy and security of data

Yang-Turner et al., elaborate a process of eliciting requirements using a model-driven prototype evaluation technique [167]. According to the authors, in situations when clear requirements are not available, model-driven methods can be used to drive basic understanding of the unknown environment in which a tool would operate. The requirements are deduced from the evaluation results of the model which takes into account the design space of the users of the tool, activities expected to be performed by the tool and the technologies involved in the building and implementation of the tool. The requirements so extracted are then used to support the building of the next iteration of the prototype development [167].

Three papers discussed the process of Requirements Analysis. One was about analysing and modeling privacy requirements for big data systems, another about analyzing requirements for IT businesses looking to adopt big data systems and the last one was about analyzing quality requirements for big data systems.

Jutla et al., propose privacy extensions to Unified Modeling Language (UML) use case diagrams to help software engineers visualize and model privacy requirements [75]. The authors proposed and created an MS Visio extension ribbon in Visual Studio called "Privacy by Design (PbD)" to help software engineer to incorporate privacy requirements into the software being created in the early phases of development.

Lamba et al., list the requirements for big data adoption for organizations to maximize their Information Technology (IT) business value [88] - having a clearly defined data strategy to decide what sources to tap for their data, a big data strategy for embedding the knowledge gained from analysis of big data into business processes, big data analytic methods incorporating new programming paradigms for batch processing as well as real time processing of incoming data, and hiring the right talent for implementing the big data technologies suited to their business needs.

Noorwali et al., focus on an approach for understanding and specifying quality requirements i.e. requirements related to quality of a system namely, attributes such as performance, reliability, security, availability, scalability and more for big data applications [111]. The proposed approach incorporates three elements - big data characteristic, quality attributes, and quality requirement description. The main idea of the authors of this paper is to intersect a big data characteristic like variety or velocity with a quality attribute like performance, scalability e.g., $velocity \times performance$ or $volume \times scalability$.

Under the micro category "Requirement Specification", a research study by Girardi et al., found proposes creating domain models using MADEM (Multi-Agent Domain Engineering Methodology) techniques to guide the specification of the requirements for a family of multi-agent driven Web recommender systems based on usage mining and collaborative filtering [60]. Another research study uses AADL (Architecture Analysis & Design Lan-

guage) to specify and model the requirements of big data-driven cyber-physical systems [173].

4.2.1.2 Open Research Challenges

There were no research studies found that discussed Requirements Process, Requirements Validation or Software Requirements Tools during the software requirements phase of big data system development.

1. **Requirements Process** involves having measures and benchmarks for big data systems that are fundamental to ensure that the requirements laid down for big data systems are met throughout their development life cycle. One such important process is conflict management - identifying requirements conflicts and discussion among the various stakeholders involved to resolve and decide upon the trade-offs to be made which would end up being acceptable to the stakeholders all the while continuing to be within the constraints and expectations when it comes to system performance and project budgets. Conflict management is critical to big data systems where decisions about prioritizing attributes but remaining within technical, budgetary and regulatory constraints need to be made regularly. For example, greater computational power to support in-database analytics would always be desirable and could be achieved by setting up more clusters of commodity hardware. However, this approach could conflict with budget restrictions and resource management constraints.

2. **Requirements Validation** is the process of verifying and validating the requirements documents created as a result of the Requirements Analysis and Requirements

Specification. Requirements validation is important because the requirements documents created will guide the design, development, testing and even post-deployment maintenance of the software systems. Because of the nature of the cutting edge technologies used in big data systems, it is crucial that descriptions of the operational procedures and overall functioning of the complex technologies involved are clearly documented in the requirements documents. Requirements validation comes in play here to ensure that the descriptions in such documents are consistent with the accepted technical and organizational standards, non-contradictory and complete. To this end, formal notations and requirements model validation are two tasks that could be most accurate techniques to be used for big data system requirements validation. Another important technique to validate that the big data system is being developed to meet the expectations of the stakeholders, is to build prototypes and performing acceptance testing.

3. **Software Requirements Tools** can be used to perform two tasks - modeling of the requirements of a system and managing the requirements. For big data systems, managing the requirements is a really important task in order to ensure that none of the requirements identified during the elicitation process are overlooked. Revisions to the requirements of a big data system in the analysis phase need to be documented and traced in order to ensure that all the vital information required for the subsequent dependent tasks and processes are available. Because of the complexity and novelty of big data systems and the technologies involved, avoiding confusion and overcoming logical loop holes would only be possible if dedicated software requirements tools are used to manage requirements and make them easily available and be descriptive with change management and revision histories.

Open research challenges in the software requirements KA include:

- **Requirements Modeling**: Modeling the requirements of big data systems helps in detecting inconsistencies and contradictions in the requirements elicited from all the stakeholders. How to explicitly represent properties related to the big data 7 Vs through requirements models? How to capture the value of the applications through requirements models as understood by the application stakeholders? How to explicitly prioritize system requirements in the models? How to represent semantic properties of system model data entities together with the contextual operations on them that depend on these properties?

- **Requirements Conflict management**: Requirements conflicts can occur at various levels between stakeholders functional requirements and non-functional system requirements. How to perform non-functional conflict management involving attributes such as computational power, resource management constraints for big data systems? How can conflicting privacy and security requirements provided by different stakeholders be resolved in big data systems? How to prioritize the non-functional requirements that are indispensable to the big data system under development? How to evaluate the impact of design decisions on big data system functional and non-functional requirements?

- **Technology Selection and Software Mapping**: Mapping of the features and advantages of big data technologies and tools with the requirements of a big data system is fundamental to the success of the system. How to select technologies and software resources for big data system development that best suit the requirements? How to define technologies and resources in the abstract in order to explicitly represent the mapping between technology components and the requirements of the big

data system? How to use ontologies for this purpose? How to represent properties, separately and in combination, related to underlying technologies, massive data analysis tools, and end-user applications?

- **Formal Notations**: Formal notations are useful to support representation and verification of the big data system requirements. How to develop formal notations to map documented organization standards with widely accepted technical standards? How to use formal notations to verify the correctness of the critical properties of big data software systems especially when distributed technologies are involved?

- **Novel Prototyping Techniques and Methods**: Prototypes are created in order to verify that the big data system about to be developed will satisfy the requirements elicited from the stakeholders. What prototyping techniques can be used to validate the requirements of a big data system? How to decide between incremental versus a full system approach to prototyping?

4.2.2 Software Design

Software design is defined as both "the process of defining architecture, components, interfaces, and other characteristics of a system or component" and the result of [that] process [143]. Software design can be viewed as an activity in the software engineering life cycle and also as a product of the software engineering life cycle which details the architecture of the software namely the internal structure, the individual components and mechanisms by which they interact with each other. Out of the 152 research studies identified by the literature review in this thesis, 109 proposed approaches, methods and practices which were categorized under Software Design KA.

The micro categories according to SWEBOK for the Software Design KA were:

1. Software Structure and Architecture
2. User Interface Design
3. Software Design Quality Analysis and Evaluation
4. Software Design Notations
5. Software Design Strategies and Methods
6. Software Design Tools

Table 4.4 lists all the research studies that were identified under the Software Design KA micro categories.

4.2.2.1 Analysis

Of the 109 research studies categorised under this KA, 79 were about Software Structure and Architecture, two about User Interface Design, three dealt with Software Design Notations, 20 detailed Software Design Strategies and Methods and five were about Software Design Tools. Out of the 79 research studies that discussed Software Structure and Architecture, seven were specific to architecture design decisions, 66 were directly related to architectural design and implementation, four provided architectural frameworks, and two discussed design patterns for big data systems.

Anderson shared the experience of working with big data by designing a data intensive software system in support crisis informatics research using Twitter feeds [8]. His research

Table 4.4: Software Design Micro Categories

Software Design Micro Categories	**Papers**	**Count**
Software Structure and Architecture	[2], [3], [67], [4], [7], [8], [9], [11], [16], [20], [23], [28], [32], [34], [35], [36], [37], [38], [39], [40], [43], [44], [47], [50], [54], [55], [56], [59], [61], [63], [64], [124], [69], [71], [72], [76], [77], [81], [80], [82] [83], [87], [90], [93], [100], [103], [105] [106], [109], [110], [116], [117], [119], [125], [128], [129], [130], [131], [136], [137], [138], [141], [145], [148], [149], [151], [153], [154], [156], [157], [160], [161], [162], [163], [165], [168], [174], [175], [177]	79
User Interface Design	[6], [170]	2
Software Design Quality Analysis and Evaluation		–
Software Design Notations	[25], [173], [172]	3
Software Design Strategies and Methods	[29], [31], [45], [46], [48], [51], [53], [86], [99], [101], [104], [107], [114], [127], [132],[133], [134], [152], [159], [171]	20
Software Design Tools	[14], [26], [68], [74], [85]	5

study listed the following design challenges in building data-intensive software systems, the choices that could be made to address these challenges and the trade offs in opting for those choices:

1. Lack of developer support due to a scarcity of tools available to developers in designing and building the software architectures of big data systems.

2. Need for multidisciplinary teams because finding the right people with the prerequisite years of experience in fields like machine learning, statistics, and graph theory, natural language processing, information retrieval, information visualization, user interface (UI) design would be difficult.

3. Intensive life cycles and commitment to a domain would be needed to understand

the application domain of the system and the needs and culture of the end users of the big data system requiring multiple iterations of design and development.

4. Matching frameworks with requirements in designing big data systems, the characteristics of the distributed system frameworks available for use needed to properly match the requirements of the big data system being developed.

5. Easy becomes hard at scale because the huge amounts of data that are consumed, processed, stored and visualized by big data systems require new methods and algorithms.

Gorton et al., built a knowledge base QuABaseBD (Quality Attributes at Scale Knowledge for Big Data Base - pronounced *"k-baseBD"*) [64]. It is a Web-based wiki interface of a collection of computer science and software engineering. It links software design principles for big data systems with the database feature taxonomy of recently popular distributed database technologies through a semantic model. It was created specifically for designing big data systems with scalable database technologies. The authors hope QuABaseBD to become a trustworthy and enduring resource for supporting the design of big data systems, and to exemplify how curated, dynamic knowledge bases can be viewed as sources of highly reliable scientific knowledge and lay the foundation for the next generation of software engineering decision support tools.

The design and implementation of a user interface (UI) to view climate model data is described in [6] by Alder et al. Yongpisanpop et al., developed a bug tracking system for big data systems with interaction and visualization attributes to help keep track of reported bugs. These were the only two studies found to have done research in user interface design for big data systems.

Lichen Zhang proposes a method using software design notations based on Architecture Analysis and Design Language (AADL) to allow the design of an object-oriented and component model for big data driven cyber physical systems in [173]. The same author also proposed a conceptual design for a framework to model complex cyber physical systems based on the integration of AADL, Modelicaml and clock theory in [172]. The author illustrated the approaches taken in a case study of modeling an aviation cyber physical system in [173] and through a case study of conceptual design of aviation cyber physical systems in [172].

Marín-Ortega et al., propose a new model in designing business intelligence (BI) solutions for big data utilization in [101]. According to the authors, the main advantage of their approach is to reduce the amount of time spent in the design phase of building a BI solution and to achieve flexibility of the BI solution so designed by removing problems which come upon adding new data sources. Their model is called the ELTA (*E*xtract, *L*oad, *T*ransform and *A*nalyse) as opposed to the standard preprocessing method - ETL (Extract, Transform, Load). The authors argue that the conventional ETL process applies transformation after extraction and loads the transformed data into the data warehouse at the end, not leaving much room for flexibility if environmental changes need to be taken into account. On the other hand, the authors posit ELTA as a process that enables data extraction from heterogeneous sources (*Extract*), storing data into a storage system, like a database (*Load*), transforming the data from the original state upon demand or depending upon business decisions (*Transform*), and finally utilizing the preprocessed data to make informed decisions about the business and the functioning of the system (*Analyse*). The main advantage of using ELTA instead of the conventional ETL according to the authors is because the data is loaded first and only then transformed, the data transformations can be applied and re-applied or completely skipped depending on the business require-

ments. This flexibility is lost if the data is transformed and then loaded because different forms of transformations are no longer possible if the current set up does not allow storing of intermediate or metadata which in most cases is expensive and not always considered necessary.

Bersani et al., make a very valid point through their work on designing continuous and rapidly evolving architecture of stream based systems in [14] by stating big data application design to be a very new and emerging field which explains the dearth of software design patterns. The patterns and approaches taken by the authors in developing the software design tool - OSTIA (On-the-fly Static Topology Inference Analysis) were derived from related fields like pattern and cluster based graph analysis. The authors recall from experience that when it comes to designing and developing big data architectures, a key complexity is evaluating the effectiveness of the same. They explain that effectiveness of architecture in terms of big data is the ability of an architecture to support design, deployment, operation, refactoring and subsequent re-deployment of architectures continuously and consistently with runtime restrictions imposed by big data development frameworks. The authors state that OSTIA allows the visualization of big data architectures for the purpose of design-time refactoring while also maintaining constraints that would only be evaluated at later stages like deployment and run-time. It allows designers and developers to infer the application architecture through on-the-fly reverse engineering and architecture recovery.

4.2.2.2 Open Research Challenges

There were no research studies that addressed the task of Software Design Quality Analysis and Evaluation for big data systems.

1. **Software Design Quality Analysis and Evaluation** is related to the quality analysis and evaluation of software system design namely attributes like maintainability, testability, robustness, portability, usability, correctness etc. It is an important activity during the development of a software product because it involves analysis and evaluation of quality attributes that are distinguishable during runtime like availability, security and also those that can't be figured out during runtime like reusability, modifiability. For big data systems, there should be techniques for formal as well as informal quality analysis of software design artifacts like architecture and design reviews to ensure that the security of the system being developed would continue to be preserved throughout its lifetime operations. Design vulnerability analysis helps pinpoint security weakness in big data. Since security is a major concern in several application domains of big data system deployment, such as e-commerce websites handling customer credit card information, banking software allowing customers to transfer cash in between accounts, military organizations preventing information leaks and espionage, etc., software design quality analysis and evaluation is a major task to be performed during big data system design.

Open research challenges in the software design KA include:

- **Function Based Design**: The function basis method for designing a system is useful because it is helpful in formulating the system architecture and the flow of information between the different architectural components. How to develop big data system architecture using formalized function based methods? How to develop function-based design measures to assess the software design of big data systems quantitatively?

- **High Level System Design**: After the requirements gathering and analysis processes are complete, the overall design and architecture of a system is conceived in the form of reference architecture before deciding on the individual components. How to define general reference architectures and frameworks for big data systems addressing the 7Vs? How to instantiate general big data frameworks to specific application domains in an automated or semi-automated way? How to design big data systems based on primitive building blocks or operations in a constructive manner?

- **Component Analysis**: Overall architecture design is only the first step towards designing a system. Breakdown of the architecture into system components and identifying their individual functions is necessary for easy implementation through code and for promoting reuse. How to dissect the high level architectural design of a big data system into individual components that can be minimal and capable of reuse for multiple functions? How to measure the properties of the internal content of each class when object oriented design patterns are used to design big data systems? How to define architectures that move massive data analytical computations closer to data storage locations?

- **Non Functional Design Attributes**: Attributes like scalability or security and privacy are non-functional attributes of a system design because they are not vital for the functioning of a system but are important and desirable features to have in a big data system that deals with huge and unpredictable volumes of sensitive data. How to analyse the security of the software design for big data systems? How to design a big data system in order to handle more data processing capability than envisioned in the current requirements? How to design a big data system to make it robust so that failure of one system component will not lead to total production

outage? How to evaluate whether policies involving aspects such as privacy and security are satisfied by a given design or implementation?

- **Execution Attributes**: Design attributes that come into play during the runtime of the big data system code execution need to be accounted for during the software design phase. How to evaluate the status of big data system design attributes like availability during runtime? How to empirically measure the design attributes that are not measurable during the runtime of a big data system like testability?

- **Software Design Tools**: Tools that help in formulating, evaluating and visualizing the design of a big data system may be vital to ensure that no feature is misunderstood or even missed out when incorporating the software requirements into system design functionalities. How to design quality software design tools that can translate the complex performance requirements of big data systems into software design for the same? How to design developer support tools that help design and visualize the structure and architecture of big data systems?

4.2.3 Software Construction

Software Construction is the process of creating working software through a combination of coding, verification, unit testing, integration testing, and debugging [17].

The micro categories from the Software Construction KA according to SWEBOK are below:

1. Construction Technologies

2. Software Construction Tools

Table 4.5 shows the research studies classified under the Software Construction Micro Category.

Table 4.5: Software Construction Micro Categories

Software Construction Micro Categories	Papers	Count
Construction Technologies	[2], [67], [4], [11], [23], [34], [36], [38], [50], [54], [76], [90], [100], [103], [124], [129], [137], [148], [153], [160], [161], [163], [168]	23
Software Construction Tools		–

4.2.3.1 Analysis

23 research studies discussed construction technologies and none dealt with software construction tools for big data systems.

Akmal et al., discuss the architecture and implementation of a web based GIS running on cloud services like Amazon EC2 (Elastic Cloud Compute) or Microsoft Azure Cloud using off the shelf components like Microsoft technologies like Bing Maps, Silverlight, SQL Azure, .NET employing RESTful APIs [4].

Begoli et al., discuss technologies best suited for different tasks in the reference architecture they create for a knowledge discovery system [11] like using Hadoop - MapReduce as well as HDFS (Haddop Distributed File System) for data preparation and Apache Hive to be used as a batch oriented data warehouse while using Hive as well as HBase to deal with structured and semi-structured data.

Cheng et al., elaborate the motivations behind selecting specific technologies like CouchDB, HDFS and Spark for data for implementation of a smart city testbed to exemplify the design of an IoT system for smart city platforms [34].

Dajda et al., use object oriented programming concepts and functional programming by implementing a prototype in Python and Erlang along with CSV and graph visualization tools like GraphViz to realize a software architecture dedicated to distributed heterogeneous data integration in a big data system [36].

4.2.3.2 Open Research Challenges

According to SWEBOK, the entire process of software construction is centered around minimizing complexities and anticipating change. Both of these fundamental characteristics could be considered hallmarks for development of big data systems due to their use of different types of novel technologies, data types and sources and a constantly evolving operational environment. Additionally, in organizations that utilize legacy systems for handling transaction as well as historical data like mainframes and AS/400 systems, big data systems would have to integrate conventional structured data with newer unstructured, and even streaming data for analytical processing and information extraction. Construction of software powering these tasks would have to account for the differences in the input file formats, the latency of data extraction, the optimal techniques to create output files and meta data which could be stored for archival purposes or reused for more business intelligence operations. The newer big data technologies are disruptive and need good construction techniques to work in tandem with older Enterprise Resource Planning (ERP) and legacy systems used commonly in organizations like banks, and multinational retailers.

Numerous big data systems will use learning algorithms which are developed by specialists in machine learning. However, these specialists may not be experts in programming [115]. In such situations, user friendly tools would need to be developed to help write as well as debug code for big data system that do not have a steep learning curve and can be handled by newcomers to programming. Additionally, tools and frameworks are scarce not only for designing big data systems [8] but for assisted code development for the same.

Open research challenges in the software construction KA include:

- **Multisystem Integration**: Big data system adoption in the industry would involve merging the functioning of conventional modules handling structured data with newer processing modules that would consume unstructured, semi-structured or streaming data. How to construct software for big data systems to achieve smooth integration of legacy systems with the current distributed computing technologies? How to include structured historical data in the current data analysis processes that use unstructured data?

- **Assisted Code Development**: Software developers have been using tools and integrated development environments for faster and error free code generation for conventional software development. How to develop software developer support tools customized for big data systems? How to develop integrated development environments capable of supporting machine learning, non relational database querying, or streaming analytics? How to develop tools for writing code for software development that does not require data to reside in memory but can source the data from the different distributed data stores?

- **Development Techniques**: Newer development techniques would need to be used to create code that can handle non-relational, streaming or multimedia data for

complex data processing and analytics. How to develop Test Driven Development techniques for big data systems in order to make it easier to understand the actual functioning and output expected of the software system? How to develop software development techniques that can handle multiprocessing technologies as well as different types of multimedia data?

4.2.4 Software Testing

According to SWEBOK, software testing consists of the dynamic verification that a program provides expected behaviors on a finite set of test cases, suitably selected from the usually infinite execution domain [17].

Ten research studies were categorized under the Software Testing KA. The micro categories identified under this KA were:

1. Test Techniques
2. Test Related Measures
3. Test Process
4. Software Testing Tools

Table 4.6 lists all the papers that were found.

Table 4.6: Software Testing Micro Categories

Software Testing Micro Categories	Papers	Count
Test Techniques	[42], [91], [95], [98], [139], [146], [147], [169]	8
Test Related Measures	[147]	1
Test Process		–
Software Testing Tools	[147]	1

4.2.4.1 Analysis

Eight research studies dealt with test techniques for big data and one each with test related measures and software testing tools. B. Li et al., proposed a novel approach to protecting databases used in big data applications by minimizing and sanitizing the database for parent organizations to send their big data to outsourcing vendors for testing [91]. The argument made is the task of testing data intensive software systems is outsourced to test centers in order to keep costs low and quality high. Since data sets contain sensitive information and variations in privacy laws governing different locations where vendors may be based, sharing of raw data is risky. To avoid legal and ethical issues, information can be anonymized. In cases of big data systems, minimizing data sets by removing data to make anonymization easier would not be an option due to the existence of useful patterns in large data sets. The authors proposed an approach called *Protecting and mInimizing databases for Software TestIng taSks (PISTIS)* that sanitizes and minimizes data using a weight based data clustering algorithm that partitions data.

N. Li et al. focused on a method for generating small and representative datasets from very large sets of data in order to save on the costs of processing large amounts of data which restricts continuous integration and delivery in agile environments [95]. They intro-

duced a novel scalable big data test framework using various Amazon services like Amazon Web Services (AWS), Amazon Elastic MapReduce (EMR), and Amazon Simple Storage Service (S3) and Redshift to test ETL applications that use big data techniques. This is achieved by using characteristics of domain-specific constraints, business constraints, referential constraints, statistical distribution and other constraints.

Ding et al., developed a test framework using an iterative metamorphic testing technique for testing scientific software and for validating machine learning algorithms [42] and Sneed et al., proposed an automatic testing process to test big data because of the sheer size of the data sets making it difficult for developers [139].

The only research study that discussed both micro categories of Test Related Measures and Software Testing Tools discussed the requirements challenges of testing big data systems and proposed a factory model for big data systems with testing strategies, testing tools, principles and matrices of testing [147].

4.2.4.2 Open Research Challenges

A major issue with testing big data systems is the infeasibility of replicating the exact production big data environment onto an test environment [99]. Big data systems are huge and complicated; hence, replicating them involves a lot of resources in the form of technical skills and cost for the hardware. Another factor is the dynamic nature of the data involved, replicating the source data to behave exactly as it would in production could turn out to be very complex. The most obvious approach available is to scale down the resources - storage and computational - needed by the production system to fit the needs of testing it. But there has been no strategy developed to decide how much scaling down would be appropriate.

Another factor that requires scaling is the huge amounts of data in the big data systems. An approach proposed by N.Li et al., involved reducing large data sets into a smaller representational form [95] but it was specifically for ETL applications. The real challenge is how the same techniques of creating small representative data sets can be created for other types of big data systems, especially ones that deal with different multimedia data like text, audio, speech, video, etc.

The only micro category in the Software Testing KA for which no research studies were found was Test Process.

1. **Test Process** is the collection of testing concepts, techniques and measures for testing a software system [17]. A lot of factors contribute to the formulation of a test process like the attitudes of the programmers/developers, test documentation and the cost and effort budget allocated to testing the system. Due to the variety of technically skilled personnel required to work on big data system development, starting from statisticians to front end UI developers and business analysts, the attitudes of everyone involved may not be flexible or trusting enough to work with non-established and customized test processes or agree on the standards and terminology used in test documentation.

Open research challenges for the software testing KA include:

- **Traceability**: Tracing the functions and behaviour of the different system components of a big data system is vital to ensure that the component functions as desired or expected. How to devise unit test cases for complex and intricate big data technology components? How to keep track of the dynamic nature of the software components of a big data system while testing it?

- **Test Environment**: Having a dedicated test environment that is a replica of the production environment is the best option to test and understand the functioning of a software system in order to fix any actual and potential errors. In big data systems, implementing this may not be economically viable. How to scale a production big data system in order to replicate the same in a dedicated test environment? How to prioritize which system components may need exact replication and which components can be scaled to a minimum to keep hardware costs low? How to duplicate the resource intensive tasks and workflow of a production big data system in order to test it? How to capture the production workloads and replay them in case of testing big data systems?

- **Test Cases**: Devising test cases in order to perform testing on a big data system code base would require the input and processing logic be similar to the production environment. How to replicate the behaviour of the data sources of a big data system in order to keep the test cases are similar as possible to the actual use cases of the big data system? How to model behaviour of system components which cannot be replicated? How to create representational data sets of input data that can be used for testing a big data system? How to define test cases that are representative to big data systems by using a subset of the resources used by the system?

4.2.5 Software Maintenance

SWEBOK defines software maintenance as the totality of activities required to provide cost-effective support to software [17]. These activities involve preparation tasks before the deployment of software as well as all that is required to keep the software running

smoothly after deployment. In addition to all the pre and post deployment tasks, planning and scheduling these tasks are equally important.

The micro categories according to SWEBOK for the Software Maintenance KA are:

1. Maintenance Process
2. Techniques for Maintenance

Table 4.8 lists the research studies found related to this KA.

Table 4.7: Software Maintenance Micro Categories

Software Maintenance Micro Categories	Papers	Count
Maintenance Process		–
Techniques for Maintenance	[96], [169], [170], [176]	4

4.2.5.1 Analysis

Only 4 research studies were categorised under the Software Maintenance KA, all of which discussed techniques for maintenance. One of the techniques by Li et al., provided a performance evaluation framework for identifying potential performance issues in MapReduce and conduct performance optimization of big data system components powered by MapReduce [96]. Yim introduced a fault tolerant automation framework to automate end-to-end software deployment procedures and proposed principles and techniques designed to test the automation programs for such software deployment [169].

Yongpisanpop et al., designed a bug tracking system that could help keep track and visualize the bugs reported during the maintenance phase of a big data system [170]. Zhou et al., discuss an empirical study on quality issues of big data systems and a diagnosis of commonly adopted mitigation solutions for development and maintenance practices of production big data platforms [176].

4.2.5.2 Open Research Challenges

There were no research studies found that discussed maintenance processes for big data systems.

1. **Maintenance Process** - Once a software product is deployed, its performance needs to be closely and constantly monitored and any issues tracked in order to solve them as soon as possible. After some components have lived their life, and newer and better technology comes on the market, older components of the existing system need to be replaced or updated. According to the IEEE Std 14764-2006 [73], maintenance process activities mainly involve process implementation, problem and modification analysis, modification implementation, maintenance review/acceptance, migration and software retirement. The vigorous influx of new technologies in the big data scene means that there is always the possibility of finding more efficient and sometimes even cheaper substitutes to currently deployed algorithms and existing frameworks of a big data system. There should be established processes to guide the deployment and software migration of big data systems all the while keeping the integrity of the big data system components intact. Modification requests must be accepted and clearly understood by the maintenance team, developers who had been involved in the

whole system or specific component to be modified and the appropriate management team. Traditional incident management mechanisms and off-the-shelf software may be insufficient for big data systems because of the rapid changes that in-database analytics algorithms perform. For any error or bug to be found and an incident reported to the maintenance team(s), sophisticated state capturing mechanisms and stack traces should be in place to help in tracking all the incidents and getting all the information that could have caused the error or bug.

There is a good chance of failure(∼16%) during software deployment of big data systems developed through iterative software life cycle models like agile and that in many cases it is attributed to human errors, out of which ∼51% could have been prevented via automation [169]. Observations like these could move research into software maintenance towards automation of maintenance processes and techniques. The need for automation for big data systems is justifiable due to the complex and new technologies being used and with which most software developers, testers or system reliability engineers may not have familiarized themselves. Familiarity with the pitfalls of deploying a complex technical system in a production environment is only possible through experience, but most the technologies incorporated in big data systems have been around for less than a decade. More procedure driven processes that have been formalized and planned in advance to account for the unpredictable nature of big data technologies and big data sources and are automated to perform the deployment processes could be the answer to the challenges in software maintenance of big data systems.

Open research challenges in software maintenance KA include:

- **Maintenance Tools**: Software maintenance tools would help - in deploying corrected or updated code into production big data systems, to track errors and outages

in the production environment, and to report errors and performance issues to responsible personnel. How to create incident managements systems for the multiple interfacing technologies involved in a big data system? How to prioritize errors and fix service level agreements for each kind of error to manage the response appropriate for each error? How to capture the system state prior to a major production outage for efficient root cause analysis?

- **Maintenance Management**: Maintenance management involves identifying and prioritizing the maintenance tasks according to the degree of severity of problems created by not performing them and in devising techniques to perform them in the most efficient manner. How to analyse and develop a priority scale for conflicting maintenance tasks? How to decide which activities during and after deployment would be better off automated? How to automate software deployment and modification activities in a big data system? Given that big data systems are fairly complicated, how to decide when a maintenance task is too complex and needs to be assigned to the development team? How to decide on when to perform scheduled maintenance tasks?

4.2.6 Software Configuration Management

Software Configuration Management is defined as a discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record or report changes processing and implementation status, and verify compliance with specified requirements [143].

SWEBOK had the following micro categories for the Software Configuration Management KA:

1. Software Configuration Identification
2. Software Configuration Control
3. Software Configuration Status Accounting
4. Software Configuration Auditing
5. Software Release Management and Delivery
6. Software Configuration Management Tools

4.2.6.1 Analysis

There were no research publications found that studied software configuration management for big data systems. It could be indicative of the fact that big data software developers are applying existing configuration management practices for software projects due to lack of any configuration management systems or strategies specifically developed for big data systems. This is a cause for concern because increasing use of large digital and dynamic data sets demands processes of maintaining system integrity while handling changes to both the data set and the software system consuming it [97].

Configuration management is all about tracking a software system in order to mark any changes made to it, analyze the consequences of a proposed change and trace the performance and functions of the software system so as to provide a road map to each

encountered error, its point of origin as to when it could have entered into the system and providing information about how to debug it and even hint to solutions. This whole process becomes crucial to rollback from regressions that may appear in a software system and even resolving the problems that arise if a regression is encountered. The complexities inherent to big data systems demand that strict configuration management is practiced throughout the development and maintenance phases.

4.2.6.2 Open Research Challenges

Configuration management for big data systems would involve a range of tasks as identified in the micro categories of this KA:

1. **Software Configuration Identification** involves identifying the items to be controlled, the version schema to be used, the tools for tracking changes made to items, and establishing configuration control mechanisms. Identifying appropriate items like the components that drive the data analytics engine of the big data system would get the highest priority but what about the pre-processing techniques applied on the source big data? The algorithms running underneath the main analytics processes that deal with the extraction and archiving of the metadata generated at the different stages of the data analytics process may also need to be made part of configuration management if the business intelligence team prioritizes the metadata.

2. **Software Configuration Control** involves establishing a Configuration Control Board (CCB) that would oversee every change request being made to the big data system. All the stakeholders with appropriate authority would make up the CCB. Some projects tend to have the team leader, lead business analyst and manager of

the business team forming the CCB. However, in big data systems, at a minimum the team leads of each group of developers, the lead business analysts of each component and key members of the management team must form the CCB. Because of the variety of technologies involved in big data systems, at least one stakeholder from each component technology must have a seat at the table when change requests are proposed and approved/rejected in order to make sure the impact of each is well understood before arriving at a decision. Key stakeholders must be assigned who have enough subject matter expertise to form the team that signs off on each change request.

3. **Software Configuration Status Accounting** should be mandatory for big data systems in order to record and report all the technical reasons and business motivations involved in requesting a change so that the changes being made to the big data systems are traceable. Rigorous documentation can ensure that contradictory changes are avoided and provide a detailed history that could guide future changes to the same system component. In addition to documenting the reasons for a change request, current performance metrics of the system component about to be changed should be logged for comparative study to ensure a performance lag is not introduced. Periodic reports must be generated tracking the changes and the relative improvement in the system performance for the CCB.

4. **Software Audits** must be routinely performed by an audit team who are not directly involved with the big data system component under audit to ensure that the component continues to meet the requirements laid down at the time of its inception and according to characteristics mentioned in the subsequent change request documentation related to it. The audit team should be formed by members from separate

development projects having similar skill sets and experience in the domain in order to conduct the audit in a short period of time. There should audits specific to the design attributes of the big data system to ensure that multiple change requests driven by business factors have not deviated the product drastically from the specifications in the design documentation and specific to the functions of the big data systems to ensure quality standards and specifications continue to be satisfied.

5. **Software Release Management and Delivery** for big data systems would involve having a dedicated team for handling release managements tasks, a standard for any large scale software project. This team would be in charge of handling all the distributions of the software components being changed which can vary depending on different environments or even geographically separate markets in order to combine all the specific software versions and supplementary parts before deploying the release. There should be automated tools for big data system release management and delivery in order to avoid trivial errors that could arise due to the complexities of a big data system. Visualization techniques should also be important to track and view all the historical, recent and current releases of a big data system components to study the update patterns of the system and to detect anomalies that could arise due to change requests and releases made to specific components.

6. **Software Configuration Management Tools** mainly involve version control tools, build handling tools and change control tools. An automated tool for release management and tools for generating periodic reports for configuration status accounting should be necessities for configuration management of big data systems.

No significant development changes in identification, or status accounting, or auditing, or change control and release management has been made in the software configuration

management KA in the recent past. However, the advent of big data systems in the software engineering community should be a catalyst in bringing about much needed change to handle the mushrooming intricate big data software ecosystem [97].

The open research challenges for this KA are:

- **Component Identification**: Identifying the software components that must go through configuration control is important in order to account for the changes made to the system and to roll back to previous versions in case for deployment failures. How to identify and prioritize the components of a big data system that must be part of software configuration management? How to decide on the limit of historical versions to be preserved in case of extreme situations requiring rollback to older versions of the big data system or individual system components?

- **Tool Support**: Configuration management tools can also help in visualizing all the configuration changes a system has experienced in order to study the improvement in performance over time. How to develop an integrated tool environment to track system and configuration component status after change requests have been implemented?

- **Configuration Management Automation**: Configuration management tasks that are simple but involve repetitive and mundane tasks can be automated to avoid human errors. Also, parts of configuration management related to status accounting like recording current system metrics can be automated to generate a status accounting dashboard that can be used by the CCB to justify or negate change requests to big data system components. How to develop automated tool support for configuration status accounting in big data systems to avoid human errors? How to develop a

status accounting dashboard with a history of all the reasons for each change request and the pre- and post- change request performance metrics and systems status.

4.2.7 Software Engineering Management

Software engineering management can be defined as the application of management activities - planning, coordinating, measuring, monitoring, controlling, and reporting [1] to ensure that software products and software engineering services are delivered efficiently, effectively and to the benefit of the stakeholders [17].

The micro categories identified under this KA are:

1. Software Project Planning
2. Software Project Enactment
3. Review and Evaluation
4. Closure
5. Software Engineering Measurement
6. Software Engineering Management Tools

Table 4.8 lists the research study found related to this KA.

Table 4.8: Software Engineering Management Micro Categories

Software Engineering Management Micro Categories	**Papers**	**Count**
Software Project Planning	[49]	1
Software Project Enactment		–
Review and Evaluation		–
Closure		–
Software Engineering Measurement		–
Software Engineering Management Tools		–

4.2.7.1 Analysis

Only one study was found that had performed research in the software engineering management KA. Dutta et al., provided a comprehensive roadmap for organizations to conceptualize, plan and successfully implement big data projects [49]. They provide a framework for implementing a system incorporating big data and present its validity based on a case study at a manufacturing company, based on identifying and accomplishing these milestones:

1. Business Problem
2. Research
3. Cross Functional Team Formation
4. Project Roadmap

4.2.7.2 Open Research Challenges

Awareness of the importance and potential impact of adoption of big data into the software systems of an organization are fundamental for the success of developing a big data system. According to a report published in The Economist Intelligence Unit, one of the biggest hurdles to the adoption of big data in organizations in the Asia-Pacific were the organizations themselves [126]. Among the internal roadblocks to the adoption of big data identified in the report, the ones most likely to benefit from the application of software engineering management techniques were - lack of willingness to share data, lack of communication between departments, departmental divisions and no buy-in from management.

Lack of communication between different departments of an organization, between its employees and management, and even between the different levels of management are all factors for failure to even consider, let alone adopt a complex and unknown entity like big data systems.

A dedicated software engineering management team with project managers working along with all the stakeholders involved and tasked with software project planning, enactment, review and evaluation, closure and measurement would be instrumental in efficient big data system development and its subsequent success.

1. **Software Project Planning** involves selecting the correct software life cycle model to best fit the business requirements; determining the deliverables in each phase of the life cycle model - data cleansing processes, data analytics algorithms, distributed data storage rules and techniques; effort, schedule and cost estimation of each individual phase and the total project - number of trained personnel needed, number

of commodity hardware clusters required for the distributed file systems, etc.; resource allocation - funds to be set aside for creating servers to replicate the test and development environments off of the production environment; and risk and quality management - to analyze the cost of production outages and the necessary hours and people involved to overcome the same and the factors that would need to be focused on during development in order to avoid them altogether.

2. **Software Project Enactment** would involve the team of project managers working closely with subject matter experts (SME) or domain experts, developers, testers, business analysts and managers to ensure that all that had been agreed upon during the project planning phase is followed meticulously. Monitoring and control activities in big data systems must be in place to make sure that all deployment or production errors are reported in real time and not lost due to the dynamic factors of big data, e.g., corrupt data lost amidst the real time streaming data flow, and contained with minimum damage. Service Level Agreements (SLA) established during the planning phase would give an idea of the response times that each failure or outage must be contained in and in categorizing the risks associated with each type of error or outage in order to establish the severity of the situation and mobilize the appropriate resources required to mitigate them. Reporting of monitoring and control activities must be undertaken periodically for analysis by the software engineering management team and the team of managers responsible for the big data system components in order to have a pulse of the functioning of the big data system as a whole.

3. **Review and Evaluation** would involve the study of two factors - whether the big data system continues to satisfy the requirement specifications established in the past and review of performance of the big data system in order to identify bottle

necks and problem areas. The review and evaluation tasks would be a part of the big data system development process through its life cycle and would continue well after deployment.

4. **Closure** involves the task of evaluating if the plans and processes assigned for a particular project, iterative cycle or the entire development endeavour for a system were completed. Evaluation of closure would involve SMEs, developers and managers meeting with the project managers and going over how each big data system component involved had met its performance expectations based on the documentation maintained throughout the life cycle of the component. All the performance metrics recorded during the configuration management processes to implement new changes must be satisfied or exceeded in order to assign a particular development cycle successful and complete to graduate for closure.

5. **Software Engineering Measurement** for big data systems would involve an organizational commitment to establish measurement metrics and repeatedly measure the system being developed, maintained or tested. Identifying the constraints, goals and risks of the big data system and its multitude of components would the most important step towards measuring the big data system. Measurements needed to evaluate the system would involve metrics like database throughput, database latency, throughput of machine learning or natural language processing based analytics etc., and collecting data for the same. Communicating this data with all the stakeholders will help in plugging the loopholes if any in the understanding or expectations of the system.

6. **Software Engineering Management Tools** involve project planning and management tools and risk management tools. However, for big data systems an integrated

set of tools used during the entire duration of the project would be necessary to keep track of all the plans, decisions, project data, system monitoring data, service level agreements, resource allocation data, estimates for outages and error mitigation procedures, effort required in each development phase and a breakdown per component etc., by the project managers is mandatory to be able to analyze all relevant data and make business decisions.

Open research challenges in the software engineering management KA include:

- **Extended SLAs**: Conventional SLAs currently used in the software engineering industry may be inadequate in dealing with the dynamic and complicated nature of big data technologies and input heterogeneous data. How to define SLAs given the complexity of the big data system? How to relate SLAs with the 7Vs of big data like incorporating value? How to define SLAs for big data systems where an error may be produced at random due to a one time occurrence in input data and which cannot be replicated?

- **Evaluation and Measurement Methods**: Routine evaluation and measurement processes for the activities involved in big data system development is necessary to ensure that such activities are functioning and obtaining results as expected. How frequently must review and evaluation tasks be conducted in big data systems? How to establish software engineering measurement practices for streaming data processing engines because of the unpredicatbility of input data?

- **Tool Support**: Tools that help in managing and coordinating activities between the different stakeholders, primarily the technical and managerial teams, can greatly help in streamlining the activities and responsibilities involved in the development

and maintenance of big data systems. How to develop integrated tool support for project management activities of big data systems? How to relate the management tasks with analytical tasks so that such tasks can be reused and replayed in future projects?

4.2.8 Software Engineering Process

Software engineering processes, according to SWEBOK, are concerned with work activities accomplished by software engineers to develop, maintain, and operate software, such as requirements, design, construction, testing, configuration management, and other software engineering processes [17].

The micro categories of this KA according to SWEBOK are:

1. Software Lifecycles
2. Software Process Assessment and Improvement
3. Software Measurement
4. Software Engineering Process Tools

4.2.8.1 Analysis

There were no research studies found for the software engineering process KA.

4.2.8.2 Open Research Challenges

The software engineering process KA is closely related to all the other KAs identified in this thesis and part of SWEBOK. Additionally, using the most appropriate software engineering processes would rightly ensure the success of a big data system right from its planning and development phases into well beyond the post deployment stage.

1. **Software Lifecycles** encompass software life cycle models, and software process adaptations. With respect to big data systems, the most important part of the software life cycle would be choosing the software life cycle model. This emphasis on the software life cycle model is governed by the importance of the type of model chosen and how it can impact the entire development process of the big data system. Since big data systems are very complicated and the technologies used in creating them novel and complex, the life cycle model used for big data system development and its capabilities in adapting to change and unforeseen circumstances are vital. There has been research already conducted about the suitability of adaptive software development life cycle models like agile for development of big data software systems [29] [57]. Since agile models are designed for versatility in order to incorporate changing requirements and because big data systems and their requirements are highly susceptible to change, the pairing of agile software life cycle models with big data system development seems appropriate.

2. **Software Process Assessment and Improvement** involves appraisal and evaluations of software processes. These come into play when evaluating a software process performed by an entity internal to the big data software development project like a different development team or even external agents like third party software vendors,

before incorporating their data or changes into the source system in the interest of which the evaluation is being conducted. This micro category would also involve performing iterative improvements by measuring, analyzing and planning changes to be made to subprocesses part of the big data system. This micro category is strongly related to performing the activities in the Software Maintenance KA. Portions of code, algorithm logic flow, data fetching and analyzing and many such similar activities that make up the cogs of a big data system need to be routinely and iteratively assessed and improved using process improvement techniques such as the Plan-Do-Check-Act model.

3. **Software Measurement** is a topic that closely relates to the Software Engineering Measurement micro category of the Software Engineering Management KA. This micro category applies the methods and techniques for measuring the current conditions of a software component about to be changed for optimization or debugging purposes in order to provide a baseline for comparison. Root cause analysis, statistical analysis, orthogonal defect classification etc., are various software measurement techniques for analysing errors and bugs along with their frequency of occurrence that have been identified in the different components of a big data system.

4. **Software Engineering Process Tools** include code editors for notations like business process modeling notation (BPMN) which can be used at times to design big data systems [25]. Configuration management tools like Apache Subversion (SVN) or Git would also be considered as software engineering process tools and without which a complex software project like development of a big data system can never be performed systematically and accurately. Visualization systems like dashboards, another kind of software engineering process tool would also be mandatory for visu-

alization of very large data sets of input data. Visualization of bugs and their related historical information would also be beneficial in performing maintenance and testing operations of complex big data systems.

Open research challenges of the software engineering process KA include:

- **Lifecycle Models**: Among many factors, software lifecycle models are chosen depending on the requirements and urgency for developing the big data system. The type of model being chosen can greatly impact the development process and ultimate success of the big data system. How to develop a new software life cycle model that can handle all the changes and complexities of big data systems? How to customize an existing software life cycle model for big data system development if development of a new model is not viable?

- **Software Measurement and Process Tools**: Software measurement techniques and process tools help in measurement of the system performance and its general status and in enabling software developers to create robust and efficient big data systems by providing all the relevant system technical information and assistance. How to develop software measurement techniques for identifying areas for process improvement in a big data system? How to identify the best among the available software engineering process tools for the big data system under development? How to develop visualization tools that can handle large data sets and depict them in the most convenient human readable formats and frames. How can analytical process be captured explicitly so that they can be reused, replayed or assessed at runtime?

4.2.9 Software Engineering Models and Methods

Software engineering models and methods impose structure on software engineering with the goal of making that activity systematic, repeatable, and ultimately more success-oriented [17]. The scope of software engineering models and methods is wide and could range from a single software engineering life cycle phase to cover the entire software life cycle.

The micro categories in SWEBOK for this KA are:

1. Types of Models
2. Analysis of Models
3. Software Engineering Methods

Table 4.9 illustrates the micro categories and the research studies found for this KA:

Table 4.9: Software Engineering Models and Methods Micro Categories

Software Engineering Models and Methods	**Papers**	**Count**
Types of Models	[5], [12], [18], [26], [58], [66], [70], [75], [92], [94], [108], [112], [113], [118], [121], [122], [135], [140, 141, 142, 144], [164], [166], [172], [178]	25
Analysis of Models	[22]	1
Software Engineering Methods	[19], [30], [130], [150]	4

4.2.9.1 Analysis

From the number of research studies found under this KA, it can inferred that researchers are focusing on using software modelling techniques and methods to facilitate big data system development. 25 research studies proposed software models and modelling techniques for activities and processes that contribute towards big data software development like requirements mapping, modeling design and architecture, modeling the performance of I/O operations of program modules, models to generate semantically rich data, etc.

One research study provided a method to analyze software models and four research studies proposed software engineering methods for big data systems.

Al Zamil et al., proposed a framework to automate multilayered classification algorithms for the purpose of resolving organizations issues in big data collections [5]. It provides an ontology layer that establishes semantic interpretation of data coming from heterogeneous sources eventually letting information from multiple sources to be converged into one. Ontology based classifiers are applied to large sets of heterogeneous data coming generated by sensors in order to interconnect all the data sources so that the relevant data points can be extracted and analyzed to glean important information.

Belo et al., propose a framework for the conceptual modeling for the ETL process, one of the most important in data warehousing in [12]. Their approach is based on creating a pattern-oriented and task-reusable framework using YAWL (Yet Another Workflow Language) - a workflow modeling language. Through this framework, the authors try to model the requirements validation and the common tasks and behaviour of ETL processes. BY doing this they encourage reuse of ETL processes in populating a data warehouse with business critical information. Since ETL processes are central to businesses utilizing data

warehousing and information analysis and are altered over time by the evolution in technology, reuse of ETL processes can save time and resources when the businesses try to achieve scalability in order to satisfy new business and market demands and expectations.

Cecchinel et al., discuss specific aspects of the development of Large Scale Infrastructures (LSIs) which are used to collect data of its surroundings environment. LSIs are implementations of Cyber Physical Systems that involve continuous monitoring of its environment. The authors propose a tooled approach to generate an abstract model of the requirements of a given LSI system in order to map it to timed automata and code generation techniques that guide processes that enable reusable data collection behaviors in the LSI system. They propose the *COSmIC* framework, a set of *Composition Operators for Sensing InfrastruCtures* [22] to this end and validate the requirements they identified by setting acceptance criterion for each.

Chen et al., propose a cost effective and systematic risk management model for strategic prototyping for agile big data system development in [30]. They developed the RASP (Risk-Based, Architecture-Centric Strategic Prototyping) model. This model is meant to be applied in cases of big data system development when architecture analysis falls short of addressing and understanding all the risks that need to be accounted for in the architecture design. The authors consider three kinds of prototypes - throwaway prototypes, vertical evolutionary prototypes and minimum viable product (MVP). Throwaway prototypes also called rapid prototypes as the name suggests are used as proofs of concepts. Vertical evolutionary prototypes involve developing one or more system components with full functionality in each release. An MVP is an evolutionary prototype in which only those core features are developed that are required in product deployment in order to minimize the time spent on each iteration. MVP facilitates in hypothesis testing by helping in collecting

information from real users and usage data about features of a product helping developers in accepting and hence enhancing features that show promising results or rejecting them altogether in case of failure. The authors found through their case studies in [30] that vertical evolutionary prototypes were more suitable to big data system development.

4.2.9.2 Open Research Challenges

Open research challenges in software engineering models and methods KA include:

- **Quality Models**: Modeling the quality attributes of a big data system is important and in most cases a difficult task. How to develop software engineering models to formulate and analyze the quality requirements of big data systems? How to develop models to analyze the quality of input data in a big data system?
- **Process Analysis Models**: Models for analyzing the different processes involved in big data system development are important to simulate their behaviour and to verify their accuracy and performance. How to develop models for simulating the data transformations of big data during data preprocessing? How to model the analysis processes to be applied to sensor data in Cyber Physical Systems and Internet of Things (IoT) systems? How to model the integration of different data types belonging to relational and non relational data models for preparation towards data analytics? How to model the behaviour of streaming data? How to model big data component technologies like Hadoop MapReduce or stream based processing in Apache Spark?

4.2.10 Software Quality

SWEBOK states that the term "Software Quality" is overloaded because software quality may refer to the desirable characteristics of software products, to the extent to which the product possess those characteristics, and to processes, tools, and techniques used to achieve those characteristics [17].

The micro categories for this KA are:

1. Software Quality Management Processes
2. Software Quality Tools

Table 4.10 provides the two research studies identified under this KA.

Table 4.10: Software Quality Micro Categories

Software Quality	Papers	Count
Software Quality Management Processes	[72]	1
Software Quality Tools	[21]	1

4.2.10.1 Analysis

For both of the micro categories of this KA, there was one research paper each. Immonen et al., emphasize on metadata - structured information about the data itself, and "quality metadata" - metadata about the quality attributes and metrics of the data - in order to evaluate the quality of social media data being processed and moved around in the architecture and the logic pipeline of a big data system. They propose a big data reference

architecture which incorporates metadata management with a dedicated metadata store, and quality management for assigning value to the quality attributes based on the data sets and its metadata [72]. Metadata is divided into groups - navigational, process, descriptive, quality and administrative based on established standards for metadata. The authors go on to use quality variability and quality policies to generate quality metadata in different phases of the metadata management in the pipeline of their big data system.

Casale et al., propose a quality-aware model driven engineering methodology, DICE aimed at developing a tool chain for quality engineering of big data systems [21]. DICE tool chain aims to offer simulation, verification and architectural optimization for design development of big data systems in addition to feedback analysis methods for iterative improvements based on monitoring data from test or production environments.

4.2.10.2 Open Research Challenges

Software quality assurance is extremely necessary for big data systems because of the novelty of big data technologies used, which makes it hard for all the stakeholders to comprehend the impact, issues and performance requirements during the requirements or design phases of development. In order to develop, operate, maintain and routinely enhance a big data system successfully, quality of all the components involved has to be prioritized. The components of big data systems in need of quality assurance measures would include but not be limited to the data sources providing the "big data" to a big data system, the data cleansing processes that make the data fit for processing, the data sets of "big data" itself, the machine learning or clustering algorithms running on these data sets, the computational logic and the implementation through code of the data analytics engine, the processes involved in storing the processed data into distributed datastores and data

warehouses and finally the process to visualize the information extracted from the data processing engine. Any errors in evaluating the accuracy and reliability of any of these process or the data that they use can result in severe issues in the performance, robustness and eventual success of a big data system.

Open research challenges in software quality KA include:

- **Quality control using metadata**: Metadata is the underlying information about the data at hand and can be used to judge its importance and its suitability for different analytical processes. How to track the provenance of big data to generate metadata in order to appraise its quality before consumption in a big data system? How to use metadata in order to perform quality assurance of big data coming from all the heterogeneous sources? How to identify big data system deviation patterns by analyzing metadata collected at runtime?
- **Quality Assurance**: Practices put in place for quality assurance will ensure reduction of errors and adherence to requirements and design decisions made before big data system development leading to stakeholder satisfaction. How to formulate a software quality assurance plan for big data systems? How to develop techniques for quality control of big data technology components like the MapReduce engine or Cassandra datastore throughput? How to develop tools which would automate quality checks on database analytics processes and their results before they are used by the organization to make business decisions? How to select the appropriate techniques to perform multiple quality attribute evaluation for big data systems? How to complement testing with (formal) verification techniques for big data systems? How to monitor big data systems to ensure their quality at runtime?

4.2.11 Software Engineering Professional Practice

The Software Engineering Professional Practice KA is concerned with the knowledge, skills, and attitudes that software engineers must possess to practice software engineering in a professional, responsible, and ethical manner [17].

The micro categories identified in this KA are:

1. Group Dynamics and Psychology
2. Communication Skills

4.2.11.1 Analysis

There were no research studies found that dealt with this KA.

4.2.11.2 Open Research Challenges

The age of information explosion and big data's emergence on the scene has got the world of computing, technology and business enthusiastic to capture new markets, reap more profits and create deeper inroads into the minds of consumers and establish unwavering brand loyalties for products and services provided. In all this confusion about big data system design and architecture, complex machine learning and statistical analysis methods and business intelligence, the role of software engineers and software engineering teams of developers, testers, system reliability engineers, business analysts and managers is fundamental. A good team of talented, skilled and responsible people are required to create successful big data systems.

Also important are the ethics of the software engineers and the management personnel involved in the software development of big data systems especially because big data systems may deal with private information of customers like credit cards, health insurance, car insurance, or even interact with law enforcement, legal authorities and government organizations. In situations like these, compliance with all legal authorities and laws of the areas of operation are mandatory. In light of recent situations like the Greyball[1] program used by the transport network company Uber, to evade law enforcement authorities to operate in areas where their operations were not permitted, software engineers' adherence to ethics and professional responsibilities come under the limelight. The in-house tool built by Uber used location data, credit card information, and social media accounts of customers it suspected to be working in law enforcement. The tool would give such customers a fake view of its ride hailing app and "greyballed" them because it suspected that they were requesting rides as part of a sting operation to prove that Uber was violating regulations by operating in areas where it wasn't allowed. Reports reveal that this tool was used by Uber in cities all over the world to evade agents working for law enforcement authorities. In situations like these, the ethics and professionalism of software engineers come into question when the employer's or client organization's business interests conflict with the local laws and common standards, ethics and morals.

The ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices developed the Software Engineering Code of Ethics and Professional Practice (Version 5.2) as the standard for teaching and practicing software engineering [65]. These guidelines were developed to guide software engineering professionals to conduct their work with full realization of the obligations they have as creators of software that would affect

[1]https://www.theguardian.com/technology/2017/mar/03/uber-secret-program-greyball-resignation-ed-baker

the lives and even future of all its users.

1. **Group Dynamics and Psychology** of a big data development team is crucial to the success of a big data development team. Big data has caused the amalgamation of multiple disciplines for creating revolutionary software. Building big data software would involve hiring experts from fields that could not be more similar to each other - specialists in distributed computing technology or server engineers, vision and human psychology experts leading user interface design, cartography designers, and even mathematicians and statisticians who work in the data analytics teams. The social and work culture of these professions might be traditionally very different but due to big data, people from completely different educational backgrounds come together to develop software systems and that brings new issues to the forefront. These issues include the differences in problem solving and outlook towards work ethics and hours, socio-cultural differences in the practices of software engineers from the world of technology are very different from the conservative and formal work culture of the banking and finance that most experienced business analysts would have experience in. Stereotypes about each group of people involved and their cultural, language and technical backgrounds could be hindrances to efficient teamwork.

2. **Communication Skills** are the foundation of developing any kind of good working software. Communication makes it possible for envisioning and discussing the requirements of any software system. In the case of big data systems, clearly and successfully communicating the requirements of big data systems in light of all the complexities and scale of operation of big data becomes imperative. Participation of all the stakeholders in the process of gathering requirements, designing and implementing code to realize a big data systems is only possible if clear communication

channels are established between key players and all teh departments and teams involved in the process.

Open research challenges in software engineering professional practice KA include:

- **Personnel Training**: All the personnel involved in big data system development should be aware of the technology components involved irrespective of their technical or business backgrounds or technical specialization. How to communicate the novel big data technology components to clients and business teams from non-technical backgrounds? How to train non-technical personnel in the business and management teams in charge of making business decisions based on big data analytics to realize the operations, importance and impact of big data?
- **Compliance**: Compliance with all the standards, laws, regulations and statutory recommendations is mandatory in order to avoid future run in with legal authorities and to be responsible towards customers and clients. How to ensure that all legal, technical and business regulations are complied with during software development? How to make it possible for employees to report questionable practices and unethical practices by teammates or management. How to stop breach of private customer data from big data system datastores and warehouses by unscrupulous members of the development or support teams?

4.2.12 Software Engineering Economics

Software engineering economics is about making decisions related to software engineering in a business context [17].

The SWEBOK micro categories in this KA are:

1. Life Cycle Economics
2. Risk and Uncertainty
3. Economic Analysis Methods

4.2.12.1 Analysis

There were no research studies found that discussed this KA for big data systems.

4.2.12.2 Open Research Challenges

The most highlighted software engineering challenges are the ones that deal with the design and architecture of a software system but the challenges that are equally important are decisions for balancing the performance of the system with the costs involved in realizing it and analyzing if it would be viable for an organization to even try to achieve maximum performance. In the current scenario, virtual memory and cheap hardware have become comparatively affordable making it possible to have server farms and clusters of commodity hardware to run data analytics processes and use as distributed datastores. However, decisions still have to be made when the actualization of a vital feature of a software product is dependent on some key limited resource [15].

1. **Life Cycle Economics** play a very important role in the development and post deployment maintenance of big data systems. The different tasks involved in developing and later supporting and enhancing a big data system should be split into

different projects that are executed sequentially and sometimes even simultaneously and would involve project life cycle activities like Initiating, Executing, Monitoring and Controlling, and Closing [1]. Performance measurements must be put in place in order to evaluate the gradations in pricing to be transferred to customers based on the performance of the big data software product. The project management teams responsible for the different software components of a big data system must be prepared in advance to make replacement and retirement decisions because of the complexities and rapid innovation in technology on the big data scene, software components may have to be upgraded with better versions of existing technology component or even retired in order to be replaced with a completely new technology.

2. **Risk and Uncertainty** plays a very prominent role in big data systems because of the huge complex computational overheads involved. The risk of having to hire more skilled professionals than there is budget allocated for technical resources is always a possibility because of the novelty of big data technologies. This would create hurdles in the effort, schedule and cost estimates made during the planning phases for the projects of a big data system. The chances of errors causing major production outages and uncertainty because of the dynamic nature of big data systems are also factors that govern the measures taken by developers and project managers in making a big data system as fault tolerant as possible.

3. **Economic Analysis Methods** for big data systems for making decisions may be based on one of many techniques like ROI (Return On Investment), MARR (Minimum Acceptable Rate of Return), ROCE (Return On Capital Employed) etc. These decision techniques would only be used by for-profit organizations that would be running a business and providing a product or service powered by big data in order

to create customer satisfaction and profits for investors and stakeholders. In case of non-profit organizations using big data like government, healthcare organizations, environmental conservationists etc., would use techniques like cost-benefit analysis, cost-effective analysis, break-even analysis etc. Optimization analysis for the processes involved in a big data system is also vital in determining the economic impact and potential of a big data system and its product(s) or service(s). These would help identify problem areas where performance can be improved.

Open research challenges of software engineering economics KA include:

- **Economic Analysis**: Having economic analysis methods in place to calculate the effort and time and costs involved throughout the entire development process is instrumental in ensuring that the development project remain within budget and completes on time within the means of the capital or economic resources at hand. How to identify the best cost and economic analysis methods for a big data system project? How to select factors that could help in deciding between cost and performance of a big data system? How to select an economic analysis technique appropriate for the big data system under development? How to calculate ROI period and plan to support the costs till ROI is attained?

- **Risk Analysis**: Analyzing the causes and likelihood of system issues and business errors and managing the cost of the fallout due to the errors are part of preparing for the risks involved in developing and managing a big data system. How to assess the risks of effort, schedule and cost estimates during project planning phase of big data system? How to evaluate for uncertainty related to availability of input data to the big data system? How to cope with data and event uncertainty using probabilistic methods?

4.3 Big Data - Data Types and Technology Trends

All the buzz around big data has brought to the attention of the software engineering research community and the world at large of the different types of data and the different types of technologies that are becoming part of the big data ecosystem. This section provides a breakdown of the different types of big data that were identified through the research studies analyzed in this thesis and the trends in the big data technologies used in them.

4.3.1 Big Data - Data Types

The different types of big data identified from the research studies covered by the literature review in this thesis are described when the one of the V's of big data - Variety was discussed in the Introduction chapter of this thesis. Some research studies explicitly mentioned the type of data being used, others did not. In such cases, if the data source was not mentioned, no categorization was provided for the study.

Table 4.11 shows all the research studies that used one or a combination of the different types of big data.

Table 4.11 is meant to highlight the fact that most of the research is being carried out on applications dealing mainly with streaming data. Mining streaming data is the most difficult and in ways the most rewarding in terms of business opportunities and profits among all the big data types.

However, majority of the companies that are new on the big data scene and looking to developing big data systems are bound to have data warehouses maintaining histor-

Table 4.11: Big Data Types

Big Data Type	Papers	Count
Structured	[2], [11], [69], [72], [116], [139], [144], [156], [175]	9
Semi-Structured	[11], [48], [69], [72], [116], [139]	6
Unstructured	[2], [48], [54], [69], [72], [116], [136], [139], [162]	9
Streaming	[14], [38], [47], [53], [61], [72], [76], [116], [144], [145], [175]	11

ical data. There could be great benefit in mining this historical data mostly stored in relational databases and used in combination with current unstructured data to extract value and insights. More research from the software engineering community needs to focus on manipulating structured data from existing data warehouses in conjunction with the unstructured data coming in from newer data sources to make breakthroughs in business opportunities.

4.3.2 Big Data - Technology Trends

Unlike data types, the type of big data technology used or discussed by the authors of the research studies were in most occasions explicitly mentioned or referenced. The categorization of the research studies in this thesis based on their use of specific big data technologies was mainly to understand the comparative popularity and frequency of attention received by each technology when viewed in the context of software engineering for big data.

The trends in the big data technologies used specifically when applying software engineering for big data systems is illustrated in Table 4.12.

In addition to the technologies mentioned in Table 4.12, some technologies to look

out for on the big data scene are Apache Flume, YARN for processing streaming data, Apache Giraph for graph processing, Neo4j for graph database management - graph storage and processing, Amazon Redshift for data warehousing, Amazon S3 web services for data storage powered by Amazon DynamoDB.

Table 4.12: Big Data Technology Trends

Big Data Technology	Papers	Count
Machine Learning	[2], [11], [18], [28], [38], [42], [50], [56], [87], [94], [116], [133], [135], [151], [152], [159], [165], [175]	18
Clustering	[38], [45], [50], [85], [91], [174]	6
Cloud Computing	[2], [67], [4], [9], [16], [19], [25], [32], [37], [47], [55], [56], [61], [124], [71], [74], [83], [96], [100], [104], [106], [107], [108], [119], [137], [138], [140], [145], [147], [148], [149], [151], [154], [156], [165], [171], [175], [177]	38
MapReduce	[2], [67], [11], [19], [21], [28], [35], [40], [51], [53], [54], [59], [70], [74], [80], [82], [83], [95], [96], [100], [101], [103], [104], [108], [112], [116], [122], [137], [138], [144], [148], [151], [152], [154], [157], [160], [161], [163], [166], [168], [171], [175]	42
Data Mining	[11], [45], [48], [51], [56], [60], [124], [76], [85], [113], [114], [129], [145], [157]	14
HDFS	[67], [11], [34], [38], [44], [54], [59], [68], [80], [82], [83], [95], [98], [100], [109], [112], [116], [137], [138], [144], [147], [150], [151], [152], [157], [160], [161], [165], [168], [175]	30
Spark	[28], [34], [38], [59], [68], [80], [83], [100], [111], [112], [116], [137], [138], [144], [147], [151], [157], [160], [165], [175]	20
Pig	[51], [83], [100], [116], [122], [138], [160]	7
Cassandra	[8], [11], [26], [64], [81], [100]	6
MongoDB	[2], [3], [7], [20], [38], [44], [51], [54], [63], [64], [66], [71], [77], [81], [83], [87], [109], [147], [151], [157]	20
HBase	[11], [54], [61], [64], [68], [77], [80], [138], [148], [175]	10
Storm	[14], [47], [61], [66], [68], [111], [119], [138], [151], [175]	10
CouchDB	[34], [77]	2
Kafka	[116], [61]	2
Zookeeper	[61], [119], [137], [151], [175]	5

Chapter 5

Conclusions and Future Work

In this chapter conclusions for the work embodied in thesis as well as the future work to carry forward the approach of applying software engineering for big data system development is discussed.

5.1 Conclusions

This thesis revealed the application domains and the software engineering KAs specific to big data system development. The results of this thesis illustrate that there are noticeable differences in the amount of research attention received among application domains and software engineering KAs. From Table 4.1 it is clear that in the case of software engineering research for big data systems, the Information Technology application domain received much more research attention compared to other important and promising data rich domains such as Healthcare and Banking and Financial Industry.

Identifying these promising application domains is fundamental to future researchers looking for newer avenues to examine, in order to produce breakthrough studies that would be valuable to the world of technology and the global economy. There is a lot of potential for making technological advances using big data systems in domains identified by this review such as Aviation, Infrastructure, Transport and Environmental Monitoring/Conservation but unfortunately, these domains and many others have not yet witnessed much work from software engineering researchers. More focused software engineering research utilizing guidelines and best practices from the software engineering KAs for developing big data systems has the ability to transform the software management process in them and even the development of these domains themselves.

The requirements for development of big data systems continue to change and evolve constantly because of the arrival of even newer technologies that can extract and transform data into valuable information and big data systems designed today have to deal with these unknown but inevitable changes in the future, making the need for requirements engineering research even more urgent. Similarly, significant research efforts need to made towards enhancing and even customizing the existing methods and standard practices and developing novel methods for maintenance, testing, validation, verification, and quality assurance for big data system development. The high stakes and risks involved, such as system unpredictability and project failure in big data system development and support, can be mitigated by conducting extensive and thorough requirements analysis, design validation, unit and integration testing, and formal verification and developing software quality assurance techniques tailor made for big data systems.

This is mostly likely to be the first comprehensive study of existing research into software engineering KAs specifically for big data systems. The purpose of this literature

review was to understand how research in software engineering KAs to date was helping in development of big data systems and to highlight the gaps, specifically in the important activities part of each KA.

Open research challenges in each KA were suggested for providing pointers and perspective to future researchers looking into big data systems development from a software engineering point of view. It can help potential researchers identify promising but underexplored application domains and focus on using specific techniques from the different software engineering KAs to develop better big data systems. More research in these areas should motivate and help big data application developers and project managers to contribute more time, effort and resources in these different KAs for big data application development.

This thesis admittedly has constraints because of the search strategy used. The combination of automated and manual searches introduces limitations unique to each. An automated search is limited by the search strings and the underlying search engine of the targeted digital repositories. That the selection of journals and conferences is not exhaustive is the limitation of performing a manual search. Additionally, there may have been open research challenges in the KAs that were not identified even in this thesis and which would need more detailed research to discover.

5.2 Future Work

This thesis attempts to present the state of the art in software engineering specific to big data systems and reveal open challenges in the different software engineering KAs for development, maintenance and support of big data systems. Future work for this thesis

encompasses increasing the number of research outlets and digital repositories searched and incorporating new search processes to identify more research studies. Another factor that motivates future search for newer research studies is that no systematic literature review is complete as research papers' are being published after the initial search process for a review is complete. By widening the search, more promising application domains can be identified that could benefit from using big data applications.

With respect to newer search methods, snowballing can be applied to find more research studies relevant to the research questions set in this thesis. The snowballing search strategy developed by C. Wohlin [158] involves selecting a starting set of research studies and identifying further research studies by using the references of the starting set and also searching for research studies that cite the studies which formed the starting set. Repeated searches by performing snowballing on the research studies derived from the automated and manual searches can provide an extensive list of research studies covering the topic of this thesis.

An additional avenue for future work of this thesis could be investigating the limitations of the existing approaches towards the big data system development as revealed in the research studies included in this thesis. Enhancing and supplementing the existing approaches with newer processes can help the software engineering research community in generating more ideas and areas for future research.

References

[1] *A Guide To The Project Management Body Of Knowledge (PMBOK Guides)*. 5th edition. Project Management Institute and IEEE Computer Society, 2004.

[2] I. D. Addo, D. Do, R. Ge, and S. I. Ahamed. A Reference Architecture for Social Media Intelligence Applications in the Cloud. In: *2015 IEEE 39th Annual Computer Software and Applications Conference,* vol. 2. IEEE, 2015, pp. 906–913.

[3] R. Agrawal, A. Imran, C. Seay, and J. Walker. A Layer Based Architecture for Provenance in Big Data. In: *2014 IEEE International Conference on Big Data (Big Data)*. 2014, pp. 1–7.

[4] M. Akmal, I. Allison, and H. González-Vélez. Assembling Cloud-based Geographic Information Systems: A Pragmatic Approach using Off-the-Shelf Components. In: *Cloud Computing with e-Science Applications* (2015), pp. 141–162.

[5] M. G. H. Al Zamil and S. Samarah. The Application of Semantic-based Classification on Big Data. In: *2014 5th International Conference on Information and Communication Systems (ICICS)*. IEEE. 2014, pp. 1–5.

[6] J. R. Alder and S. W. Hostetler. Web based Visualization of Large Climate Data Sets. In: *Environmental Modelling & Software,* vol. 68 (2015), pp. 175–180.

[7] J. Anderson, R. Soden, K. M. Anderson, M. Kogan, and L. Palen. EPIC-OSM: A Software Framework for OpenStreetMap Data Analytics. In: *2016 49th Hawaii International Conference on System Sciences (HICSS)*. 2016, pp. 5468–5477.

[8] K. M. Anderson. Embrace the Challenges: Software Engineering in a Big Data World. In: *2015 IEEE/ACM 1st International Workshop on Big Data Software Engineering*. IEEE Press. 2015, pp. 19–25.

[9] S. Bazargani, J. Brinkley, and N. Tabrizi. Implementing Conceptual Search Capability in a Cloud-Based Feed Aggregator. In: *Third International Conference on Innovative Computing Technology (INTECH 2013)*. IEEE. 2013, pp. 138–143.

[10] E. Begoli. A Short Survey on the State of the Art in Architectures and Platforms for Large Scale Data Analysis and Knowledge Discovery from Data. In: *Proceedings of the WICSA/ECSA*. ACM. 2012, pp. 177–183.

[11] E. Begoli and J. Horey. Design Principles for Effective Knowledge Discovery from Big Data. In: *2012 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture*. 2012, pp. 215–218.

[12] O. Belo, A. Cuzzocrea, and B. Oliveira. Modeling and Supporting ETL Processes via a Pattern-Oriented, Task-Reusable Framework. In: *2014 IEEE 26th International Conference on Tools with Artificial Intelligence*. IEEE, 2014, pp. 960–966.

[13] F. Z. Benjelloun, A. A. Lahcen, and S. Belfkih. An Overview of Big Data Opportunities, Applications and Tools. In: *Intelligent Systems and Computer Vision(ISCV)*. 2015, pp. 1–6.

[14] M. M. Bersani, F. Marconi, D. A. Tamburri, P. Jamshidi, and A. Nodari. Continuous Architecting of Stream-Based Systems. In: *2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA)*. 2016, pp. 146–151.

[15] B. W. Boehm. *Software Engineering Economics*. 1st edition. Prentice Hall PTR, 1981.

[16] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu. Fog Computing: A Platform for Internet of Things and Analytics. In: *Big Data and Internet of Things: A Roadmap for Smart Environments*. Vol. 546. Studies in Computational Intelligence. Springer, 2014, pp. 169–186.

[17] P. Bourque and R. E. Fairley, eds. *SWEBOK: Guide to the Software Engineering Body of Knowledge*. Version 3.0. IEEE Computer Society, 2014. URL: http://www.swebok.org/.

[18] D. Breuker. Towards Model-Driven Engineering for Big Data Analytics–An Exploratory Analysis of Domain-Specific Languages for Machine Learning. In: *2014 47th Hawaii International Conference on System Sciences*. IEEE. 2014, pp. 758–767.

[19] M. Camilli. Formal Verification Problems in a Big Data World: Towards a Mighty Synergy. In: *Companion Proceedings of the 36th International Conference on Software Engineering*. ICSE Companion 2014. ACM. 2014, pp. 638–641.

[20] G. Cao, S. Wang, M. Hwang, A. Padmanabhan, Z. Zhang, and K. Soltani. A Scalable Framework for Spatiotemporal Analysis of Location-based Social Media Data. In: *Computers, Environment and Urban Systems*, vol. 51 (2015), pp. 70 –82.

[21] G. Casale, D. Ardagna, M. Artac, F. Barbier, E. Di Nitto, A. Henry, G. Iuhasz, C. Joubert, J. Merseguer, V. I. Munteanu, J. F. Pérez, D. Petcu, M. Rossi, C. Sheridan, I. Spais, and D. Vladušič. DICE: Quality-Driven Development of Data-intensive Cloud Applications. In: *Proceedings of the 7th International Workshop on Modeling in Software Engineering*. MiSE '15. IEEE Press, 2015, pp. 78–83.

[22] C. Cecchinel, S. Mosser, and P. Collet. Software Development Support for Shared Sensing Infrastructures: A Generative and Dynamic Approach. In: *Software Reuse for Dynamic Systems in the Cloud and Beyond ICSR 2015*. Vol. 8919. Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 221–236.

[23] C. Cecchinel, M. Jimenez, S. Mosser, and M. Riveill. An Architecture to Support the Collection of Big Data in the Internet of Things. In: *2014 IEEE World Congress on Services*. IEEE, 2014, pp. 442–449.

[24] S. Ceri, T. Palpanas, E. D. Valle, D. Pedreschi, J.C. Freytag, and R. Trasarti. Towards Mega-Modeling: A Walk through Data Analysis Experiences. In: *SIGMOD Record* vol. 42.no. 3 (2013), pp. 19–27.

[25] V. Chang and M. Ramachandran. A Proposed Case for the Cloud Software Engineering in Security. In: *The First International Workshop on Emerging Software as a Service and Analytics (ESaaSA)*. 2014.

[26] A. Chebotko, A. Kashlev, and S. Lu. A Big Data Modeling Methodology for Apache Cassandra. In: *2015 IEEE International Congress on Big Data*. 2015, pp. 238–245.

[27] C.L. P. Chen and C.Y. Zhang. Data-Intensive Applications, Challenges, Techniques and Technologies: A Survey on Big Data. In: *Information Sciences* vol. 275 (2014), pp. 314 –347.

[28] G. Chen, S. Wu, and Y. Wang. The Evolvement of Big Data Systems: From the Perspective of an Information Security Application. In: *Big Data Research,* vol. 2.no. 2 (2015). Visions on Big Data, pp. 65 –73.

[29] H. M. Chen, R. Kazman, and S. Haziyev. Agile Big Data Analytics for Web-based Systems: An Architecture-centric Approach. In: *IEEE Transactions on Big Data,* vol. 2.no. 3 (2016), pp. 234–248.

[30] H. M. Chen, R. Kazman, and S. Haziyev. Strategic Prototyping for Developing Big Data Systems. In: *IEEE Software,* vol. 33.no. 2 (2016), pp. 36–43.

[31] H. M. Chen, R. Kazman, S. Haziyev, and O. Hrytsay. 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 Press. 2015, pp. 44–50.

[32] J. Chen, J. Ma, N. Zhong, Y. Yao, J. Liu, R. Huang, W. Li, Z. Huang, Y. Gao, and J. Cao. WaaS: Wisdom as a Service. In: *IEEE Intelligent Systems,* vol. 29.no. 6 (2014), pp. 40–47.

[33] M. Chen, S. Mao, and Y. Liu. Big Data: A Survey. In: *Mobile Networks and Applications,* vol. 19.no. 2 (2014), pp. 171–209.

[34] B. Cheng, S. Longo, F. Cirillo, M. Bauer, and E. Kovacs. Building a Big Data Platform for Smart Cities: Experience and Lessons from Santander. In: *2015 IEEE International Congress on Big Data*. 2015, pp. 592–599.

[35] C. E. Cuesta, M. A. Martínez-Prieto, and J. D. Fernández. Towards an Architecture for Managing Big Semantic Data in Real-Time. In: *Proceedings of the 7th European Conference on Software Architecture (ECSA)*. Springer-Verlag, 2013, pp. 45–53.

[36] J. Dajda and G. Dobrowolski. Architecture Dedicated to Data Integration. In: *Proceedings of the 7th Asian Conference on Intelligent Information and Database Systems (ACIIDS), Part I*. Springer International Publishing, 2015, pp. 179–188.

[37] H. Demirkan and D. Delen. Leveraging the Capabilities of Service-oriented Decision Support Systems: Putting Analytics and Big Data in Cloud. In: *Decision Support Systems,* vol. 55.no. 1 (2013), pp. 412–421.

[38] L. Deng, J. Gao, and C. Vuppalapati. Building a Big Data Analytics Service Framework for Mobile Advertising and Marketing. In: *2015 IEEE First International Conference on Big Data Computing Service and Applications*. 2015, pp. 256–266.

[39] M. Deng and L. Di. Building Open Environments to Meet Big Data Challenges in Earth Sciences. In: *Big Data : Techniques and Technologies in Geoinformatics*. CRC Press, 2014, pp. 69–90.

[40] A. Desai and K. S. Nagegowda. Advanced Control Distributed Processing Architecture (ACDPA) using SDN and Hadoop for Identifying the Flow Characteristics and Setting the Quality of Service (QoS) in the Network. In: *2015 IEEE International Advance Computing Conference (IACC)*. 2015, pp. 784–788.

[41] R. M. Devens. *Cyclopaedia of Commercial and Business Anecdotes: Comprising Interesting Reminiscences and Facts, Remarkable Traits and Humors ... of Merchants, Traders, Bankers ... etc. in all Ages and Countries ...* New York, London, D. Appleton and company, 1865, p. 465.

[42] J. Ding, D. Zhang, and X. H. Hu. A Framework for Ensuring the Quality of a Big Data Service. In: *2016 IEEE International Conference on Services Computing (SCC)*. 2016, pp. 82–89.

[43] M. J. Divn, Y. B. Saibene, M. D. L. Martn, M. L. Belmonte, G. Lafuente, and J. Caldera. Towards a Data Processing Architecture for the Weather Radar of the INTA Anguil. In: *2015 International Workshop on Data Mining with Industrial Applications (DMIA)*. 2015, pp. 72–78.

[44] C. Dobre and F. Xhafa. Intelligent Services for Big Data Science. In: *Future Generation Computer Systems* vol. 37 (2014). Special Section: Innovative Methods and Algorithms for Advanced Data-Intensive Computing, Special Section: Semantics,

Intelligent processing and services for big data, Special Section: Advances in Data-Intensive Modelling and Simulation, Special Section: Hybrid Intelligence for Growing Internet and its Applications, pp. 267 –281.

[45] S. D'Oca and T. Hong. Occupancy Schedules Learning Process through a Data Mining Framework. In: *Energy and Buildings,* vol. 88 (2015), pp. 395 –408.

[46] C. C. Douglas. An Open Framework for Dynamic Big-Data-Driven Application Systems (DBDDAS) Development. In: *Procedia Computer Science,* vol. 29 (2014), pp. 1246 –1255.

[47] A. Doyle, G. Katz, K. Summers, C. Ackermann, I. Zavorin, Z. Lim, S. Muthiah, L. Zhao, C. T. Lu, P. Butler, R. P. Khandpur, Y. Fayed, and N. Ramakrishnan. The EMBERS Architecture for Streaming Predictive Analytics. In: *2014 IEEE International Conference on Big Data (Big Data)*. 2014, pp. 11–13.

[48] E. E. A. Durham, A. Rosen, and R. W. Harrison. A Model Architecture for Big Data Applications using Relational Databases. In: *2014 IEEE International Conference on Big Data (Big Data)*. 2014, pp. 9–16.

[49] D. Dutta and I. Bose. Managing a Big Data project: The case of Ramco Cements Limited. In: *International Journal of Production Economics,* vol. 165 (2015), pp. 293 –306.

[50] R. Dutta, A. Morshed, J. Aryal, C. D'este, and A. Das. Development of an Intelligent Environmental Knowledge System for Sustainable Agricultural Decision Support. In: *Environmental Modelling & Software,* vol. 52 (2014), pp. 264–272.

[51] S. B. Elagib, A. R. Najeeb, A. H. Hashim, and R. F. Olanrewaju. Big Data Analysis Solutions Using MapReduce Framework. In: *2014 International Conference on Computer and Communication Engineering*. IEEE. 2014, pp. 127–130.

[52] H. Eridaputra, B. Hendradjaya, and W. Danar Sunindyo. Modeling the Requirements for Big Data Application using Goal Oriented Approach. In: *2014 International Conference on Data and Software Engineering (ICODSE)*. IEEE. 2014, pp. 1–6.

[53] C. Esposito, M. Ficco, F. Palmieri, and A. Castiglione. A Knowledge-based Platform for Big Data Analytics Based on Publish/Subscribe Services and Stream Processing. In: *Knowledge-Based Systems,* vol. 79 (2015), pp. 3–17.

[54] S. O. Fadiya, S. Saydam, and V. V. Zira. Advancing Big Data for Humanitarian Needs. In: *Procedia Engineering,* vol. 78 (2014), pp. 88 –95.

[55] S. Fang, L. D. Xu, Y. Zhu, J. Ahati, H. Pei, J. Yan, and Z. Liu. An Integrated System for Regional Environmental Monitoring and Management Based on Internet of Things. In: *IEEE Transactions on Industrial Informatics,* vol. 10.no. 2 (2014), pp. 1596–1605.

[56] A. Forkan, I. Khalil, A. Ibaida, and Z. Tari. BDCaM: Big Data for Context-aware Monitoring - A Personalized Knowledge Discovery Framework for Assisted Healthcare. In: *IEEE Transactions on Cloud Computing,* no. 99 (2015), pp. 1–1.

[57] P. Franková, M. Drahošov, and P. Balco. Agile Project Management Approach and its Use in Big Data Management. In: *Procedia Computer Science* vol. 83 (2016). The 7th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 6th International Conference on Sustainable Energy Information Technology (SEIT), pp. 576 –583.

[58] K. Gai, M. Qiu, L. C. Chen, and M. Liu. Electronic Health Record Error Prevention Approach Using Ontology in Big Data. In: *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th*

International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems. IEEE, 2015, pp. 752–757.

[59] R. Giachetta. A Framework for Processing Large Scale Geospatial and Remote Sensing Data in MapReduce Environment. In: *Computers & Graphics,* vol. 49.no. C (2015), pp. 37–46.

[60] R. Girardi and L. B. Marinho. A Domain Model of Web Recommender Systems based on Usage Mining and Collaborative Filtering. In: *Requirements Engineering,* vol. 12.no. 1 (2007), pp. 23–40.

[61] M. O. Gökalp, A. Koçyigit, and P. E. Eren. A Cloud Based Architecture for Distributed Real Time Processing of Continuous Queries. In: *2015 41st Euromicro Conference on Software Engineering and Advanced Applications*. 2015, pp. 459–462.

[62] I. Gorton, A. B. Bener, and A. Mockus. Software Engineering for Big Data Systems. In: *IEEE Software,* vol. 33.no. 2 (2016), pp. 32–35.

[63] I. Gorton and J. Klein. Distribution, Data, Deployment: Software Architecture Convergence in Big Data Systems. In: *IEEE Software,* vol. 32.no. 3 (2015), pp. 78–85.

[64] I. Gorton, J. Klein, and A. Nurgaliev. Architecture Knowledge for Evaluating Scalable Databases. In: *Proceedings of the 2015 12th Working IEEE/IFIP Conference on Software Architecture (WICSA)*. IEEE Computer Society, 2015, pp. 95–104.

[65] D. Gotterbarn, K. Miller, and S. Rogerson. Software Engineering Code of Ethics and Professional Practice. In: *Commun. ACM,* vol. 40.no. 11 (1997), pp. 110–118.

[66] G. Gousios, D. Safaric, and J. Visser. Streaming Software Analytics. In: *2016 IEEE/ACM 2nd International Workshop on Big Data Software Engineering (BIGDSE)*. ACM, 2016, pp. 8–11.

[67] W. A. Goya, M. R. D. Andrade, A. C. Zucchi, N. M. Gonzalez, R. D. F. Pereira, K. Langona, T. C. M. D. B. Carvalho, J.E. Mngs, and A. Sefidcon. The Use of Distributed Processing and Cloud Computing in Agricultural Decision-Making Support Systems. In: *2014 IEEE 7th International Conference on Cloud Computing*. IEEE, 2014, pp. 721–728.

[68] M. Guerriero, S. Tajfar, D. A. Tamburri, and E. D. Nitto. Towards a Model-driven Design Tool for Big Data Architectures. In: *2016 IEEE/ACM 2nd International Workshop on Big Data Software Engineering (BIGDSE)*. 2016, pp. 37–43.

[69] K. Holley, G. Sivakumar, and K. Kannan. Enrichment Patterns for Big Data. In: *2014 IEEE International Congress on Big Data*. 2014, pp. 796–799.

[70] Y. Huai, R. Lee, S. Zhang, C. H. Xia, and X. Zhang. DOT: A Matrix Model for Analyzing, Optimizing and Deploying Software for Big Data Analytics in Distributed Systems. In: *Proceedings of the 2nd ACM Symposium on Cloud Computing (SOCC)*. 2011, pp. 1 –14.

[71] Q. Huang and C. Xu. A Data-Driven Framework for Archiving and Exploring Social Media Data. In: *Annals of GIS,* vol. 20.no. 4 (2014), pp. 265–277.

[72] A. Immonen, P. Pääkkönen, and E. Ovaska. Evaluating the Quality of Social Media Data in Big Data Architecture. In: *IEEE Access,* vol. 3 (2015), pp. 2028–2043.

[73] International Standard - ISO/IEC 14764 IEEE Std 14764-2006 Software Engineering 2013; Software Life Cycle Processes 2013; Maintenance. In: *ISO/IEC 14764:2006 (E) IEEE Std 14764-2006 Revision of IEEE Std 1219-1998)* (2006).

[74] Y. Jararweh, M. Jarrah, Mazen K., Z. Alshara, M. N. Alsaleh, M. Al-Ayyoub, et al. CloudExp: A Comprehensive Cloud Computing Experimental Framework. In: *Simulation Modelling Practice and Theory* vol. 49 (2014), pp. 180 –192.

[75] D. N. Jutla, P. Bodorik, and S. Ali. Engineering Privacy for Big Data Apps with the Unified Modeling Language. In: *2013 IEEE International Congress on Big Data*. IEEE, 2013, pp. 38–45.

[76] K. Kanoun, M. Ruggiero, D. Atienza, and M. v. d. Schaar. Low Power and Scalable Many-Core Architecture for Big-Data Stream Computing. In: *2014 IEEE Computer Society Annual Symposium on VLSI*. 2014, pp. 468–473.

[77] K. Kaur and R. Rani. A Smart Polyglot Solution for Big Data in Healthcare. In: *IT Professional,* vol. 17.no. 6 (2015), pp. 48–55.

[78] M. A. u. d. Khan, M. F. Uddin, and N. Gupta. Seven V's of Big Data Understanding Big Data to extract Value. In: *Proceedings of the 2014 Zone 1 Conference of the American Society for Engineering Education*. 2014, pp. 1–5.

[79] B. Kitchenham and S. Charters. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. Tech. rep. Keele University, United Kingdom, 2007.

[80] J. Klein, R. Buglak, D. Blockow, T. Wuttke, and B. Cooper. A Reference Architecture for Big Data Systems in the National Security Domain. In: *Proceedings of the 2nd International Workshop on BIG Data Software Engineering (BIGDSE)*. ACM, 2016, pp. 51–57.

[81] J. Klein, I. Gorton, L. Alhmoud, J. Gao, C. Gemici, R. Kapoor, P. Nair, and V. Saravagi. Model-Driven Observability for Big Data Storage. In: *2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA)*. 2016, pp. 134–139.

[82] G. Kousiouris, G. Vafiadis, and T. Varvarigou. Enabling Proactive Data Management in Virtualized Hadoop Clusters based on Predicted Data Activity Patterns. In: *2013 Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*. IEEE. 2013, pp. 1–8.

[83] M. Krämer and I. Senner. A Modular Software Architecture for Processing of Big Geospatial Data in the Cloud. In: *Computers & Graphics,* vol. 49.no. C (2015), pp. 69–81.

[84] V. D. Kumar and P. Alencar. Software Engineering for Big Data Projects: Domains, Methodologies and Gaps. In: *2016 IEEE International Conference on Big Data (Big Data)*. 2016, pp. 2886–2895.

[85] B. T. G. S. Kumara, I. Paik, J. Zhang, T. H. A. S. Siriweera, and K. R. C. Koswatte. Ontology-Based Workflow Generation for Intelligent Big Data Analytics. In: *2015 IEEE International Conference on Web Services*. 2015, pp. 495–502.

[86] N. Kushiro, S. Matsuda, and K. Takahara. Model Oriented System Design on Big-Data. In: *Procedia Computer Science* vol. 35 (2014), pp. 961 –968.

[87] S. Kwoczek, S. D. Martino, T. Rustemeyer, and W. Nejdl. An Architecture to Process Massive Vehicular Traffic Data. In: *2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*. 2015, pp. 515–520.

[88] H. S. Lamba and S. K. Dubey. Analysis of Requirements for Big Data Adoption to Maximize IT Business Value. In: *2015 4th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)*. IEEE. 2015, pp. 1–6.

[89] D. Laney. *3D Data Management: Controlling Data Volume, Velocity and Variety.* Tech. rep. 2001. URL: http://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf.

[90] C. Ledur, D. Griebler, I. Manssour, and L. G. Fernandes. Towards a Domain-Specific Language for Geospatial Data Visualization Maps with Big Data Sets. In: *2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA)*. 2015, pp. 1–8.

[91] B. Li, M. Grechanik, and D. Poshyvanyk. Sanitizing and Minimizing Databases for Software Application Test Outsourcing. In: *2014 IEEE Seventh International Conference on Software Testing, Verification and Validation*. IEEE, 2014, pp. 233–242.

[92] C. Li, L. Huang, and L. Chen. Breeze Graph Grammar: A Graph Grammar Approach for Modeling the Software Architecture of Big Data-oriented Software Systems. In: *Software: Practice and Experience,* vol. 45.no. 8 (2014), pp. 1023–1050.

[93] C. Li, Y. Liu, R. Li, and H. Zhang. Research and Application of One-Key Publishing Technologies for Meteorological Service Products. In: *2016 IEEE International Conference on Big Data Analysis (ICBDA)*. 2016, pp. 1–5.

[94] H. Li, D. Parikh, Q. He, B. Qian, Z. Li, D. Fang, and A. Hampapur. Improving Rail Network Velocity: A Machine Learning Approach to Predictive Maintenance. In: *Transportation Research Part C: Emerging Technologies* vol. 45 (2014). Advances in Computing and Communications and their Impact on Transportation Science and Technologies, pp. 17 –26.

[95] N. Li, A. Escalona, Y. Guo, and J. Offutt. A Scalable Big Data Test Framework. In: *2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 2015, pp. 1–2.

[96] Y. Li, K. Wang, Q. Guo, X. Li, X. Zhang, G. Chen, T. Liu, and J. Li. Breaking the Boundary for Whole-System Performance Optimization of Big Data. In: *International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE Press. 2013, pp. 126–131.

[97] C. Lindkvist, A. Stasis, and J. Whyte. Configuration Management in Complex Engineering Projects. In: *Procedia CIRP,* vol. 11 (2013), pp. 173 –176.

[98] Z. Liu. Research of Performance Test Technology for Big Data Applications. In: *2014 IEEE International Conference on Information and Automation (ICIA)*. 2014, pp. 53–58.

[99] N. H. Madhavji, A. Miranskyy, and K. Kontogiannis. Big Picture of Big Data Software Engineering: With Example Research Challenges. In: *Proceedings of the First International Workshop on BIG Data Software Engineering (BIGDSE)*. IEEE Press. 2015, pp. 11–14.

[100] S. Marchal, X. Jiang, R. State, and T. Engel. A Big Data Architecture for Large Scale Security Monitoring. In: *2014 IEEE International Congress on Big Data*. 2014, pp. 56–63.

[101] P. M. Marín-Ortega, V. Dmitriyev, M. Abilov, and J. M. Gómez. ELTA: New Approach in Designing Business Intelligence Solutions in Era of Big Data. In: *Procedia Technology,* vol. 16 (2014), pp. 667 –674.

[102] B. A. Marron and P. A. D. de Maine. Automatic Data Compression. In: *Communications, ACM* (Nov. 1967), pp. 711–715.

[103] M. A. Martínez-Prieto, C. E. Cuesta, M. Arias, and J. D. Fernández. The SOLID Architecture for Real-Time Management of Big Semantic Data. In: *Future Generation Computer Systems* vol. 47 (2015), pp. 62–79.

[104] S. Meng, W. Dou, X. Zhang, and J. Chen. KASR: A Keyword-Aware Service Recommendation Method on MapReduce for Big Data Applications. In: *IEEE Transactions on Parallel and Distributed Systems,* vol. 25.no. 12 (2014), pp. 3221–3231.

[105] N. Mishra, C. C. Lin, and H. T. Chang. A Cognitive Oriented Framework for IoT Big-Data Management Prospective. In: *2014 IEEE International Conference on Communiction Problem-solving*. IEEE. 2014, pp. 124–127.

[106] E. Moguel, J. C. Preciado, F. Sánchez-Figueroa, M. A. Preciado, and J. Hernández. Multilayer Big Data Architecture for Remote Sensing in Eolic Parks. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing,* vol. 8.no. 10 (2015), pp. 4714–4719.

[107] M. Müller, L. Bernard, and D. Kadner. Moving Code – Sharing Geoprocessing Logic on the Web. In: *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 83 (2013), pp. 193 –203.

[108] I. Mytilinis, D. Tsoumakos, V. Kantere, A. Nanos, and N. Koziris. I/O Performance Modeling for Big Data Applications over Cloud Infrastructures. In: *2015 IEEE International Conference on Cloud Engineering*. 2015, pp. 201–206.

[109] A. Naseer, B. Y. Alkazemi, and E. U. Waraich. A Big Data Approach for Proactive Healthcare Monitoring of Chronic Patients. In: *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*. 2016, pp. 943–945.

[110] D. Ning, P. Chen, G. Yuan, J. Xu, and L. Xu. Research on Warship Communication Operation and Maintenance Management Based on Big Data. In: *2014 International Conference on Cloud Computing and Big Data*. IEEE. 2014, pp. 126–129.

[111] I. Noorwali, D. Arruda, and N. H. Madhavji. Understanding Quality Requirements in the Context of Big Data Systems. In: *2016 IEEE/ACM 2nd International Workshop on Big Data Software Engineering (BIGDSE)*. ACM, 2016, pp. 76–79.

[112] R. J. Nowling and J. Vyas. A Domain-Driven, Generative Data Model for Big Pet Store. In: *2014 IEEE Fourth International Conference on Big Data and Cloud Computing*. 2014, pp. 49–55.

[113] A. Ochian, G. Suciu, O. Fratu, and V. Suciu. Big Data Search for Environmental Telemetry. In: *2014 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*. 2014, pp. 182–184.

[114] C. Ordonez, S. Maabout, David S. M., and W. Cabrera. Extending ER Models to Capture Database Transformations to Build Data Sets for Data Mining. In: *Data & Knowledge Engineering,* vol. 89 (2014), pp. 38–54.

[115] C. E. Otero and A. Peter. Research Directions for Engineering Big Data Analytics Software. In: *IEEE Intelligent Systems,* vol. 30.no. 1 (2015), pp. 13–19.

[116] P. Pääkkönen and D. Pakkala. Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems. In: *Big Data Research,* vol. 2.no. 4 (2015), pp. 166–186.

[117] E. W. Patton, P. Seyed, P. Wang, L. Fu, F. J. Dein, R. S. Bristol, and D. L. McGuinness. SemantEco: A Semantically Powered Modular Architecture for Integrating Distributed Environmental and Ecological Data. In: *Future Generation Computer Systems,* vol. 36 (2014), pp. 430 –440.

[118] N. Pelekis, Y. Theodoridis, and D. Janssens. On the Management and Analysis of Our LifeSteps. In: *SIGKDD Explor. Newsl.,* vol. 15.no. 1 (2014), pp. 23–32.

[119] L. M. A. Tchana Pham, D. Donsez, V. Zurczak, P. Y. Gibello, and N. de Palma. An Adaptable Framework to Deploy Complex Applications onto Multi-Cloud Platforms. In: *The 2015 IEEE RIVF International Conference on Computing Communication Technologies - Research, Innovation, and Vision for Future (RIVF)*. IEEE, 2015, pp. 169–174.

[120] Roger S. Pressman. *Software Engineering: A Practitioner's Approach*. Palgrave Macmillan, 2005.

[121] M. Rahmes, G. Lemieux, K. Fox, and C. Casseus. Multi-Disciplinary Ontological Geo-Analytical Incident Modeling. In: *2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC)*. 2015, pp. 77–83.

[122] A. Rajbhoj, V. Kulkarni, and N. Bellarykar. Early Experience with Model-Driven Development of MapReduce based Big Data Application. In: *21st Asia-Pacific Software Engineering Conference,* vol. 1. 2014, pp. 94–97.

[123] B. Randell. Software Engineering in 1968. In: *Proceedings of the 4th International Conference on Software Engineering (ICSE)*. Munich, Germany: IEEE Press, 1979, pp. 1–10.

[124] M. Habib ur Rehman, C. S. Liew, and T. Y. Wah. UniMiner: Towards a Unified Framework for Data Mining. In: *2014 4th World Congress on Information and Communication Technologies (WICT)*. IEEE. 2014, pp. 134–139.

[125] C. Restrepo-Arango, A. Henao-Chaparro, and C. Jiménez-Guarín. Using the Web to Monitor a Customized Unified Financial Portfolio. In: *Proceedings of the 2012 In-*

ternational Conference on Advances in Conceptual Modeling. Springer-Verlag, 2012, pp. 358–367.

[126] C. Ross. The Hype and the Hope: The Road to Big Data Adoption in Asia-Pacific. In: *The Economist Intelligence Unit* (2013).

[127] R.Šendelj, I.Ognjanović, E.Ammenwerth, and W.Hackl. Towards Semantically Enabled Development of Service-Oriented Architectures for Integration of Socio-Medical Data. In: *2016 5th Mediterranean Conference on Embedded Computing (MECO)*. 2016, pp. 436–440.

[128] S. J. Rysavy, D. Bromley, and V. Daggett. DIVE: A Graph-based Visual-Analytics Framework for Big Data. In: *IEEE Computer Graphics and Applications,* vol. 34.no. 2 (2014), pp. 26–37.

[129] Muhammad A. S., Y.K. Lee, and S. Lee. Trajectory Patterns Mining Towards Lifecare Provisioning. In: *Wireless Personal Communications,* vol. 76.no. 4 (2014), pp. 747–762.

[130] A. Samuel, M. I. Sarfraz, H. Haseeb, S. Basalamah, and A. Ghafoor. A Framework for Composition and Enforcement of Privacy-Aware and Context-Driven Authorization Mechanism for Multimedia Big Data. In: *IEEE Transactions on Multimedia,* vol. 17.no. 9 (2015), pp. 1484–1494.

[131] E. Sciacca, C. Pistagna, U. Becciani, A. Costa, P. Massimino, S. Riggi, F. Vitello, M. Bandieramonte, and M. Krokos. Towards a Big Data Exploration Framework for Astronomical Archives. In: *2014 International Conference on High Performance Computing Simulation (HPCS)*. IEEE. 2014, pp. 351–357.

[132] T. Shah, F. Rabhi, and P. Ray. Investigating an Ontology-based Approach for Big Data Analysis of Inter-dependent Medical and Oral Health Conditions. In: *Cluster Computing,* vol. 18.no. 1 (2015), pp. 351–367.

[133] F. Shen. A Pervasive Framework for Real-Time Activity Patterns of Mobile Users. In: *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. 2015, pp. 248–250.

[134] L. Shi, F. Lin, T. Yang, J. Qi, W. Ma, and S. Xu. Context-based Ontology-driven Recommendation Strategies for Tourism in Ubiquitous Computing. In: *Wireless Personal Communications,* vol. 76.no. 4 (2014), pp. 731–745.

[135] W. Shi, Y. Zhu, J. Zhang, X. Tao, G. Sheng, Y. Lian, G. Wang, and Y. Chen. Improving Power Grid Monitoring Data Quality: An Efficient Machine Learning Framework for Missing Data Prediction. In: *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*. IEEE, 2015, pp. 417–422.

[136] S. Shukla and G. Sadashivappa. A Distributed Randomization Framework for Privacy Preservation in Big Data. In: *2014 Conference on IT in Business, Industry and Government (CSIBIG)*. IEEE. 2014, pp. 1–5.

[137] S. Singh and Y. Liu. A Cloud Service Architecture for Analyzing Big Monitoring Data. In: *Tsinghua Science and Technology,* vol. 21.no. 1 (2016), pp. 55–70.

[138] R. O. Sinnott, L. Morandini, and S. Wu. SMASH: A Cloud-Based Architecture for Big Data Processing and Visualization of Traffic Data. In: *2015 IEEE International Conference on Data Science and Data Intensive Systems*. 2015, pp. 53–60.

[139] H. M. Sneed and K. Erdoes. Testing Big Data (Assuring the Quality of Large Databases). In: *2015 IEEE Eighth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 2015, pp. 1–6.

[140] G. Suciu, C. Dobre, V. Suciu, G. Todoran, A. Vulpe, and A. Apostu. Cloud Computing for Extracting Price Knowledge from Big Data. In: *2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems*. IEEE. 2015, pp. 314–317.

[141] N. Sun, J. G. Morris, J. Xu, X. Zhu, and M. Xie. iCARE: A Framework for Big Data-based Banking Customer Analytics. In: *IBM Journal of Research and Development,* vol. 58.no. 5/6 (2014), 4:1–4:9.

[142] W. Sun, F. Li, W. Guo, Y. Jin, and W. Hu. Store, Schedule and Switch - A New Data Delivery Model in the Big Data Era. In: *2013 15th International Conference on Transparent Optical Networks (ICTON)*. IEEE. 2013, pp. 1–4.

[143] Systems and Software Engineering – Vocabulary. In: *ISO/IEC/IEEE 24765:2010(E)* (2010), pp. 1–418. URL: http://ieeexplore.ieee.org/servlet/opac?punumber=5733833.

[144] K. Taneja, Q. Zhu, D. Duggan, and T. Tung. Linked Enterprise Data Model and Its Use in Real Time Analytics and Context-Driven Data Discovery. In: *2015 IEEE International Conference on Mobile Services*. 2015, pp. 277–283.

[145] K. Tao, C. Hauff, G. J. Houben, F. Abel, and G. Wachsmuth. Facilitating Twitter Data Analytics: Platform, Language and Functionality. In: *2014 IEEE International Conference on Big Data (Big Data)*. IEEE, 2014, pp. 421–430.

[146] D. G. Tesfagiorgish and L. JunYi. Big Data Transformation Testing Based on Data Reverse Engineering. In: *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing, 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing, 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom)*. 2015, pp. 649–652.

[147] M. Thangaraj and S. Anuradha. State of Art in Testing for Big Data. In: *2015 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*. 2015, pp. 1–7.

[148] D. Tracey and C. Sreenan. A Holistic Architecture for the Internet of Things, Sensing Services and Big Data. In: *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*. 2013, pp. 546–553.

[149] H.L. Truong and S. Dustdar. Sustainability Data and Analytics in Cloud-Based M2M Systems. In: *Big Data and Internet of Things: A Roadmap for Smart Environments*. Vol. 546. Studies in Computational Intelligence. Springer, 2014, pp. 343–365.

[150] M. Vanauer, C. Bhle, and B. Hellingrath. Guiding the Introduction of Big Data in Organizations: A Methodology with Business- and Data-Driven Ideation and Enterprise Architecture Management-Based Implementation. In: *2015 48th Hawaii International Conference on System Sciences*. IEEE. 2015, pp. 908–917.

[151] M. Villari, A. Celesti, M. Fazio, and A. Puliafito. AllJoyn Lambda: An Architecture for the Management of Smart Environments in IoT. In: *2014 International Conference on Smart Computing Workshops*. IEEE. 2014, pp. 9–14.

[152] A. Vinay, V. S. Shekhar, J. Rituparna, T. Aggrawal, K.N. Balasubramanya Murthy, and S. Natarajan. Cloud Based Big Data Analytics Framework for Face Recognition

in Social Networks Using Machine Learning. In: *Procedia Computer Science* vol. 50 (2015), pp. 623 –630.

[153] C. Wang, X. Li, and X. Zhou. SODA: Software Defined FPGA Based Accelerators for Big Data. In: *2015 Design, Automation Test in Europe Conference Exhibition (DATE)*. EDA Consortium. 2015, pp. 884–887.

[154] W. Q. Wang, X. Zhang, J. Zhang, and H. B. Lim. Smart Traffic Cloud: An Infrastructure for Traffic Applications. In: *2012 IEEE 18th International Conference on Parallel and Distributed Systems*. IEEE, 2012, pp. 822–827.

[155] J. S. Ward and A. Barker. Undefined by data: A Survey of Big Data Definitions. In: *arXiv preprint arXiv:1309.5821* (2013).

[156] M. Westerlund, U. Hedlund, G. Pulkkis, and K.-M. Björk. A Generalized Scalable Software Architecture for Analyzing Temporally Structured Big Data in the Cloud. In: *New Perspectives in Information Systems and Technologies, Volume 1*. Ed. by Á. Rocha, A. M. Correia, F. B. Tan, and K . A. Stroetmann. Springer International Publishing, 2014, pp. 559–569.

[157] N. Wilder, J. M. Smith, and A. Mockus. Exploring a Framework for Identity and Attribute Linking Across Heterogeneous Data Systems. In: *Proceedings of the 2Nd International Workshop on BIG Data Software Engineering (BIGDSE)*. ACM, 2016, pp. 19–25.

[158] C. Wohlin. Guidelines for Snowballing in Systematic Literature Studies and a Replication in Software Engineering. In: *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering (EASE)*. ACM. 2014, pp. 1–10.

[159] C. L. Wu, T. C. Chiang, L. C. Fu, and Y. C. Zeng. Nonparametric Discovery of Contexts and Preferences in Smart Home Environments. In: *2015 IEEE International Conference on Systems, Man, and Cybernetics*. 2015, pp. 2817–2822.

[160] D. Wu, L. Zhu, X. Xu, S. Sakr, D. Sun, and Q. Lu. Building Pipelines for Heterogeneous Execution Environments for Big Data Processing. In: *IEEE Software,* vol. 33.no. 2 (2016), pp. 60–67.

[161] F. J. Wu, X. Zhang, and H. B. Lim. A Cooperative Sensing and Mining System for Transportation Activity Survey. In: *2014 IEEE Wireless Communications and Networking Conference (WCNC)*. 2014, pp. 3284–3289.

[162] E. Xinhua, J. Han, Y. Wang, and L. Liu. Big Data-as-a-Service: Definition and architecture. In: *2013 15th IEEE International Conference on Communication Technology*. 2013, pp. 738–742.

[163] L. Xu, M. Li, and A. R. Butt. GERBIL: MPI + YARN. In: *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. 2015, pp. 627–636.

[164] Z. Xu, X. Wei, X. Luo, Y. Liu, L. Mei, C. Hu, and L. Chen. Knowle: A Semantic Link Network Based System for Organizing Large Scale Online News Events. In: *Future Generation Computer Systems,* vol. 4344 (2015), pp. 40 –50.

[165] J. Yang and Jun M. A Big-Data Processing Framework for Uncertainties in Transportation Data. In: *2015 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. 2015, pp. 1–6.

[166] S. Yang, W. Yu, Y. Hu, K. Wang, J. Wang, and S. Li. An Automatic Discovery Framework of Cross-Source Data Inconsistency for Web Big Data. In: *2015 Third International Conference on Advanced Cloud and Big Data*. 2015, pp. 73–79.

[167] F. Yang-Turner, L. Lau, and V. Dimitrova. A Model-Driven Prototype Evaluation to Elicit Requirements for a Sensemaking Support Tool. In: *2012 19th Asia-Pacific Software Engineering Conference,* vol. 1. IEEE. 2012, pp. 380–385.

[168] Q. Yao, Y. Tian, P.F. Li, L.L. Tian, Y.-M. Qian, and J.S. Li. Design and Development of a Medical Big Data Processing System based on Hadoop. In: *Journal of Medical Systems,* vol. 39.no. 3 (2015), pp. 1–11.

[169] K. S. Yim. Norming to Performing: Failure Analysis and Deployment Automation of Big Data Software Developed by Highly Iterative Models. In: *2014 IEEE 25th International Symposium on Software Reliability Engineering.* IEEE. 2014, pp. 144–155.

[170] P. Yongpisanpop, H. Hata, and K. Matsumoto. Bugarium: 3D Interaction for Supporting Large-Scale Bug Repositories Analysis. In: *Companion Proceedings of the 36th International Conference on Software Engineering.* ICSE Companion 2014. ACM. 2014, pp. 500–503.

[171] F. Zhang, J. Cao, S. U. Khan, K. Li, and K. Hwang. A Task-Level Adaptive MapReduce Framework for Real-Time Streaming Data in Healthcare Applications. In: *Future Generation Computer Systems,* vol. 43–44 (2015), pp. 149 –160.

[172] L. Zhang. A Framework to Model Big Data Driven Complex Cyber Physical Control Systems. In: *2014 20th International Conference on Automation and Computing.* IEEE. 2014, pp. 283–288.

[173] Lichen Zhang. Designing Big Data Driven Cyber Physical Systems based on AADL. In: *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC).* 2014, pp. 3072–3077.

[174] M. Zhang, H. Wang, Y. Lu, T. Li, Y. Guang, C. Liu, E. Edrosa, H. Li, and N. Rishe. TerraFly GeoCloud: An Online Spatial Data Analysis and Visualization System. In: *ACM Transactions on Intelligent System and Technology,* vol. 6.no. 3 (Apr. 2015), 34:1–34:24.

[175] W. Zhang, L. Xu, Z. Li, Q. Lu, and Y. Liu. A Deep-Intelligence Framework for Online Video Processing. In: *IEEE Software,* vol. 33.no. 2 (2016), pp. 44–51.

[176] H. Zhou, J. G. Lou, H. Zhang, H. Lin, H. Lin, and T. Qin. An Empirical Study on Quality Issues of Production Big Data Platform. In: *Proceedings of the 37th International Conference on Software Engineering - Volume 2 (ICSE)*. 2015, pp. 17–26.

[177] A. Zimmermann, M. Pretz, G. Zimmermann, D. G. Firesmith, I. Petrov, and E. El-Sheikh. Towards Service-Oriented Enterprise Architectures for Big Data Applications in the Cloud. In: *Proceedings of the 2013 17th IEEE International Enterprise Distributed Object Computing Conference Workshops*. EDOCW '13. IEEE Computer Society, 2013, pp. 130–135.

[178] A. Zimmermann, B. Gonen, R. Schmidt, E. El-Sheikh, S. Bagui, and N. Wilde. Adaptable Enterprise Architectures for Software Evolution of SmartLife Ecosystems. In: *2014 IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations*. IEEE, 2014, pp. 316–323.