

Electronic Thesis and Dissertation Repository

1-29-2020 3:30 PM

Requirements Engineering in the Context of Big Data Software Applications

Darlan Florencio de Arruda, *The University of Western Ontario*

Supervisor: Nazim H. Madhavji, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Doctor of Philosophy degree in Computer Science

© Darlan Florencio de Arruda 2020

Follow this and additional works at: <https://ir.lib.uwo.ca/etd>

Recommended Citation

Florencio de Arruda, Darlan, "Requirements Engineering in the Context of Big Data Software Applications" (2020). *Electronic Thesis and Dissertation Repository*. 6808.
<https://ir.lib.uwo.ca/etd/6808>

This Dissertation/Thesis is brought to you for free and open access by Scholarship@Western. It has been accepted for inclusion in Electronic Thesis and Dissertation Repository by an authorized administrator of Scholarship@Western. For more information, please contact wlsadmin@uwo.ca.

Abstract

Big Data applications, like traditional applications, serve end-user needs except that underlying the software system is Big Data which the system operates upon to improve or provide different end-user experience with the application. In comparison to traditional software development where the development processes are usually well-established, the development of Big Data applications is - to our knowledge - not explored to any significant degree. With Big Data, characterised by the well-known “V” attributes, questions arise as to how to elicit, specify, analyse, and document system requirements. While requirements engineering (RE) has long been recognised as critical for downstream development of computer systems, the field is currently passive about how to deal with characteristics of data in the RE process in the development of Big Data software applications. This problem is compounded by the fact that the RE field had no domain model (until recently) for Big Data systems depicting the various artefacts, activities, and relationships amongst them that, in turn, can be used to support RE specifications, product design, project decisions, and maintenance. In this thesis research, we investigated empirically a number of issues in RE involving Big Data applications, leading to the following research contributions: (i) knowledge concerning (a) the state of RE research involving Big Data applications, and (b) RE practices on real-world Big Data applications projects; (ii) a set of RE challenges in creating Big Data applications; (iii) a meta-model depicting the various RE artefacts and their inter-relationships in the context of Big Data software development projects; (iv) a goal-oriented approach (composed of a systematic process, requirements logging templates, checklists, and a requirements language) for modelling quality requirements for Big Data applications; and (v) a prototype tool that implements the proposed Big Data goal-oriented requirements language. These results lay a foundation in RE research involving Big Data applications development with anticipated impact in real-world projects and in RE research.

Keywords: Big Data Software Applications, Big Data Artefact Model, Big Data Goal-oriented Requirements Language, Modelling Approach, Quality Requirements, Requirements Modelling Tool.

Summary for Lay Audience

Big Data applications, like traditional applications, serve end-user needs except that underlying the software system is Big Data which the system operates upon to improve or provide different end-user experience with the application. Big Data is a term applied to data sets whose size or type is beyond the ability of traditional relational databases to capture, manage, and process the data. There is ample literature that suggests that the field of Big Data is growing rapidly. Also, there is emerging literature on the need to create end-user Big Data applications. However, just yet there is not a recognisable body of knowledge concerning the development of such applications. This situation is also reflected in the field of Requirements Engineering (RE). RE is the process of finding out, analysing, documenting and checking requirements and its constraints for a particular project. It forms the ground for every software project, defining what the stakeholders (e.g., users and customers) need from it, and what it must do to satisfy that needs. Therefore, in this thesis, we investigated several aspects of RE involving the development of Big Data applications leading to the following contributions: (i) knowledge concerning (a) the state of RE research involving Big Data applications, and (b) RE practices on a real-world Big Data application project; (ii) a set of RE challenges in creating Big Data applications; (iii) a descriptive depicting the various RE artefacts (e.g., documents) and their inter-relationships in the context of Big Data software development projects; (iv) an approach for modelling quality requirements for Big Data applications; and (v) a prototype tool that enables the use of the proposed approach.

Co-Authorship Statement

This thesis was written in the integrated article format. The candidate of this thesis is the primary author of all the thesis related publications (chapters 3 - 6). The candidate planned, designed, conducted the studies, analysed and interpreted the results, and wrote the published papers - included as chapters in this thesis - under the supervision of Dr. Nazim H. Madhavji, who is a co-author in the aforementioned publications. Additionally, two other researchers co-authored papers appearing as thesis chapters. Rodrigo Laigner - a current MS.c. Candidate in Computer Science at PUC-Rio - helped in the gathering of data used in the research presented in Chapter 4. Ibtehal Noorwali - a current Ph.D. candidate in Computer Science at UWO - helped in the analysis of the threats to validity of the research described in Chapter 5.

*To my loving and caring mother,
Erivane Florencio de Arruda.*

Acknowledgements

Martin Luther King Jr. once said, “*Faith is taking the first step, even when you don’t see the whole staircase*”. That describes the way I face life in both personal and professional dimensions. I put a lot of faith in everything I believe in. This attitude has led me to become the first person in my family to get a masters degrees. The only one to move away to another country to climb more stairs, explore new horizons and find new ways to believe.

Today, I am here, sitting in front of my computer writing this acknowledgements page with my heart full of gratitude for all the experience and opportunities, sometimes harder than I have anticipated, that God has allowed me to experience. During these almost five years of my PhD program, I had the privilege to meet and work with several amazing people. I gratefully acknowledge the support received from the faculty members and staff of the Department of Computer Science at the University of Western Ontario. In particular, Janice Weirsma, our graduate secretary, who plays an important role in our department.

Likewise, I acknowledge the support of all my colleagues from the Software Engineering laboratory, friends, and family. In particular, the support received from Zachary Woloch and Yasmen Wahba who made themselves available and eager to help me in critical moments of this journey. Special thanks to my friend and research partner, Ibtehal Noorwali for all the joyful moments, discussion, and research collaborations we had throughout these years of study. Thanks for always encouraging me with beautiful and motivational words even when everything seemed not to be working as planned.

I also extend my outspoken gratitude to all my industry and academic partners (especially Rodrigo Laigner and Vijay Dakshinamoorthy) for their time, patience and willingness to collaborate with me and my research.

Last but not least, I would like to acknowledge all the help, guidance, support and supervision received from my thesis supervisor, Dr Nazim H. Madhavji. For that, I am eternally grateful.

Contents

Abstract	i
Summary for Lay Audience	ii
Co-Authorship Statement	iii
Dedication	iv
Acknowledgements	v
List of Figures	xiv
List of Tables	xvi
List of Appendices	xviii
1 Introduction	1
1.1 Research Problem	3
1.2 Why RE for Big Data Software Applications?	4
1.3 Big Data Applications <i>versus</i> Traditional Data-centric Applications	6
1.4 Research Contributions	7
1.5 Thesis Structure	8
Bibliography	12
2 Background	15
2.1 Requirements Engineering	15
2.1.1 Requirements Levels of Description	16

2.1.2	Types of Requirements	17
2.1.3	Requirements Engineering Activities	17
	Elicitation	17
	Analysis	18
	Specification and Modelling	18
	Validation	18
	Management	19
	Negotiation and Prioritisation	19
2.2	Goal-Oriented Requirements Engineering	20
2.3	Big Data	20
	2.3.1 Defining Big Data	21
	2.3.2 Characterising Big Data	21
2.4	Software Engineering for Big Data Applications	22
2.5	Summary	23
Bibliography		23
3	State of Requirements Engineering Research in the Context of Big Data Software Applications	26
3.1	Introduction	26
3.2	Research Methodology	27
	3.2.1 Research Questions	27
	3.2.2 Search Strategy	28
	3.2.3 Selection Criteria	29
	3.2.4 Selection Process	29
	3.2.5 Data Extraction	30
3.3	Descriptive Data and Analysis	30
	3.3.1 Discussion	34
3.4	Results and Discussion	35

3.4.1	(Q1) What are the activities in the RE process, types of requirements and application domains targeted by the identified RE research involving the development of Big Data Applications?	36
3.4.2	(Q2) What are the Requirements Engineering Challenges in the context of Big Data Applications?	38
3.4.3	(Q3) What Solutions have been proposed in the domain of RE and Big Data Applications?	39
3.5	Recommendations for Further Research	43
3.6	Threats to Validity	43
3.6.1	Construct Validity	44
3.6.2	Internal Validity	44
3.6.3	External Validity	44
3.6.4	Conclusion Validity	44
3.7	Summary	44
3.8	Chapter Addendum	45
3.8.1	Summary	47
	Bibliography	47
4	Requirements Engineering Practices and Challenges in the Context of a Big Data Software Development Project: Insights from a Case Study	50
4.1	Introduction	50
4.2	Research Design	52
4.2.1	Research Goal	52
4.2.2	Research Questions	53
4.2.3	Data Collection	53
4.2.4	Data Analysis	54
4.3	The Case	55
4.3.1	Context	55
4.3.2	The Big Data Software Solution	56
4.3.3	The Big Data Software Development Project	57

Distribution of Team Members and Roles	57
Software Development Methodology and RE Process	58
4.4 Results	58
4.4.1 (Q1) - What are the sources for elicitation of Big Data-related software and non-software requirements? What is the proportion of the Big Data-related requirements in relation to the total number of requirements in the project?	58
4.4.2 (Q2) - How are the systems requirements (specifically the Big Data-related ones) elicited, documented, analysed, and prioritised within the project?	59
4.4.3 (Q3) - What is the role of Big Data technologies and characteristics in Requirements Engineering?	62
4.4.4 (Q4) - What are the challenges in eliciting, documenting, and analysing systems' requirements while engineering Big Data software applications?	63
4.4.5 Discussion	65
RE Practices	65
Distribution of Requirements	66
RE Challenges	67
4.5 Directions for Further Research	67
4.6 Threats to Validity	68
4.6.1 Reliability Validity	68
4.6.2 Construct Validity	68
4.6.3 External Validity	69
4.6.4 Conclusion Validity	69
4.7 Summary	70
Bibliography	70
5 An Empirically Derived Requirements Engineering Artefact Model in the Context of Big Data Software Development Projects	73

5.1	Introduction	73
5.2	Model Creation Process	75
	Identification of Elements and Concepts	75
	Definition of the Artefact Relationships	76
	Definition of Cardinalities	76
	Synthesising the Artefact Model	76
	Evaluation of the Artefact Model	76
5.3	Pre-validation version of the RE Artefact Model in the context of Big Data Software Development Projects	76
5.4	Model Evaluation Study	78
	5.4.1 Model Evaluation Process	79
	5.4.2 Descriptive Statistics	80
5.5	Evaluation Results	82
	5.5.1 Accuracy and Completeness	82
	5.5.2 Usefulness and Generalisibility	84
	5.5.3 Comparison between the proposed and improved versions of artefact model	86
5.6	Post-validation version of the RE Artefact Model in the context of Big Data Software Development Projects	87
5.7	Threats to Validity	92
	5.7.1 Construct Validity	92
	5.7.2 Internal Validity	92
	5.7.3 External Validity	92
	5.7.4 Reliability	93
	5.7.5 Selection Bias	93
	5.7.6 Experience bias	93
	5.7.7 Conclusion Validity	94
5.8	Summary	94
5.9	Chapter Addendum	94

Bibliography **98**

6 An Approach for Modelling Quality Requirements for Big Data Applications **101**

- 6.1 Introduction 101
- 6.2 Related Work 103
 - 6.2.1 Discussion 105
- 6.3 The QualiBD Approach 105
 - 6.3.1 Concepts and Assumptions 106
 - 6.3.2 Overall process of the QualiBD Approach 107
 - 6.3.3 Checking for consistency while transitioning between phases 108
 - 6.3.4 The Big Data Goal-oriented Requirements Language 109
 - The Model Elements 110
 - How are the modelling elements graphically organised? 111
 - Illustrative Example 112
 - Considerations for using the QualiBD Approach 114
- 6.4 Tool Support 114
 - 6.4.1 Tool Features 115
 - 6.4.2 Tool Implementation 115
- 6.5 Validation Case Studies 117
 - 6.5.1 Case 1 117
 - 6.5.2 Case 2 119
- 6.6 Summary 122

Bibliography **123**

7 Implications and Discussion **127**

- 7.1 Implications 127
 - Industrial practice 127
 - Academic Research 128
 - Tool Support 128
- 7.2 Discussion 129

7.2.1	Big Data Requirements Engineering Artefact Model - BD-REAM . . .	129
	Applying the Artefact Model to Agile Projects	129
	Defining Specific RE Process	130
	Analysing the Cost for Adopting the Artefact Model in Industry	131
	Generalising the Artefact Model	131
7.2.2	The QualiBD Approach and Tool	132
	Refining the Approach	132
	Identifying Solution Alternatives	132
	Why identifying solution alternatives while modelling quality require- ments?	133
	Stakeholders benefitting with the approach	133
	Using the Tool	133
7.3	Summary	134
Bibliography		134
8 Conclusions and Future Work		136
8.1	Conclusions	136
8.2	Future Work	140
Bibliography		141
A Instrument for Data Collection: Case Study Questionnaire		143
B Instrument for Data Collection: Artefact Model Validation		146
C QualiBD Tool: End-user Interface and Features		154
D QualiBD Tool: Implementation Details		158
D.1	Tool Implementation	158
D.1.1	Modelling and Code Generation	159
D.1.2	Graphics Editor	160
	Defining Node Elements	161

Defining Element-based Edges	162
Defining Tools and Operations	163
Define Validation Rules	165
D.2 Summary	166
Bibliography	166
E QualiBD Tool: Java Methods	168
F QualiBD Tool: AQL Expressions	171
Curriculum Vitae	173

List of Figures

1.1	Thesis Contributions by Chapter	8
1.2	Thesis Structure	9
2.1	Example of User and Systems Requirements.	16
3.1	Distribution of papers identified and selected distributed by phased of the selection process	31
3.2	Distribution of papers by venue of publication	32
3.3	Papers by contribution and type of research organised according to the RE activities they address	35
3.4	Updated distribution of Papers by contribution and type of research organised according to the RE activities they address	46
4.1	Scope of the exploratory case study depicting its context, case, and unit of analysis.	52
4.2	Example of tags used in coding and categorising the gathered data.	55
5.1	Model Creation Process	75
5.2	Pre-validation Big Data RE artefact model [13]	78
5.3	Model Validation Process	79
5.4	Graphical Representation of the Post-validation version of the Big Data RE artefact model (BD-REAM).	91
6.1	Process overview of the proposed approach	107
6.2	Checklist for the completeness and accuracy of the encoded quality requirement	109
6.3	Model Elements Description and Graphical Notation	110

6.4	Example of a Big Data quality requirement modelled using the proposed QualiBD approach.	113
6.5	Overview of the frameworks used in the prototype tool creation	115
6.6	QualiBD Tool Graphical User-Interface	116
6.7	Big Data quality requirement encoded using the QualiBD Approach and its corresponding checklist document (Case 1).	118
6.8	Big Data quality requirement modelled using the QualiBD Tool (Case 1).	118
6.9	Big Data quality requirement encoded using the QualiBD Approach and its corresponding checklist document (Case 2 -Requirement 1)	120
6.10	Big Data quality requirement encoded using the QualiBD Approach and its corresponding checklist document (Requirement 2 - Case 2).	120
6.11	Big Data quality requirement modelled using the QualiBD Tool (Case 2 - Requirement 1).	121
6.12	Big Data quality requirement modelled using the QualiBD Tool (Case 2 - Requirement 2).	122
7.1	RE Activities for Process Creation (Adapted from [2])	130
C.1	Graphical User-Interface of the QualiBD Tool	155
C.2	Validation of model elements added without a label or description	156
C.3	Validation of missing refinements amongst model elements	157
D.1	Ecore Meta-model of the QualiBD Tool.	159
D.2	Steps taken in the definition of the graphical editor portion of the QualiBD Tool.	161
D.3	Properties of an element-based relation for the <i>Permutation Link</i> element in Sirius.	162
D.4	Properties of an edge creation tool in Sirius for the <i>Permutation Link</i> element.	164

List of Tables

1.1	Core chapters and their associated submitted or published articles	11
3.1	Number of Papers by Year	31
3.2	Publication Venue and Number of Papers from each Venue	32
3.3	Overall distribution of papers by type of research	34
3.4	Overall distribution of papers by research contribution	34
3.5	Types of Requirements, Activities of the RE process targeted by available RE and Big Data Research	37
3.6	Overview of the solutions proposed in RE and Big Data Research	39
3.7	Additions to the SLR results reported in this chapter	46
3.8	Additions to the RE Research Challenges discussed in this chapter	47
4.1	Definition of Research Goal	53
4.2	Decomposed Research Questions	53
4.3	Technologies adopted according to the Big Data Pipeline	57
4.4	RE practices and Supporting Tools/Techniques	60
4.5	RE challenges in creating Big Data Applications	64
5.1	Descriptive statistics of the study participants.	81
5.2	Results of the accuracy and completeness questions.	84
5.3	Results of the usefulness question	84
5.4	Reasons for usefulness of the model juxtaposed by participant groups	85
5.5	Results of the generalisability question	86
5.6	Comparative statistics between the pre- and post-validation artefact models.	86
5.7	Model changes	87

5.8	Descriptions of artefacts inter-relationships	89
5.9	Main Artefact Descriptions	89
5.10	Descriptions of artefacts inter-relationships	95

List of Appendices

Appendix A 143
Appendix B 146
Appendix C 154
Appendix D 158
Appendix E 168
Appendix F 171

Chapter 1

Introduction

Requirements Engineering (RE) plays an essential role in the software engineering process, being considered one of the most critical phases of the software development life-cycle [1]. It forms the ground for every software project, defining what the stakeholders (e.g., users, customers, developers, businesses, etc.) need from it, and what it must do to satisfy that needs. Requirements engineering provides the appropriate mechanism for understanding what the customer wants, assessing feasibility, negotiating a reasonable solution, specifying the solution unambiguously, validating the specification, and managing the requirements [2].

As we might expect, then, Requirements Engineering would play a similar role in the development of Big Data software applications. These applications, like traditional applications, serve customers and end-user needs except that we expect improved, even different, experience from the system as it leverages the underlying Big Data to provide responses [3].

A 2018 International Data Corporation (IDC) study [4] predicts that the collective sum of the world's data will grow from 33 zettabytes in 2018 to 175ZB by 2025, for a compounded annual growth rate of 61%. In this context, the generation and consumption of data continues to grow so fast that it presents companies with opportunities to invest in Big Data hardware, software, and services in order to gain competitive advantage. In fact, another study from IDC [5] predicts that, in aggregate, the Big Data technology and services market is estimated to grow at a compound annual growth rate (CAGR) of 22.6% from 2015 to 2020 and reach \$58.9

billion in 2020. Revenue for Big Data infrastructure is estimated to grow at a CAGR of 20.3% from 2015 to 2020 and reach \$27.7 billion in 2020. Revenue for Big Data software is estimated to grow at a CAGR of 25.7% from 2015 to 2020 and reach \$15.9 billion in 2020. Revenue for Big Data services, which consists of professional and support services, is estimated to grow at a CAGR of 23.9% from 2015 to 2020 and reach \$15.2 billion in 2020.

However, although scientific literature [6–8] and economic reports [5, 9] indicate that the field of Big Data is growing rapidly, as yet there is no recognisable body of knowledge concerning the development of Big Data software applications. In comparison to traditional software development where the development processes are normally well-defined, the processes for the development of applications involving Big Data are not clear just yet from the scientific literature [10] - That is not only because of the intrinsic characteristics of Data (such as volume, velocity, variety, and veracity) and exponential growth of data sets and rates [11, 12], but also due to the fact that Big Data applications are complex solutions with several dynamic components, being distributed computation nodes, networks, databases, middleware, and business intelligence layers [13].

The complexities arising from designing and engineering such complex applications lead to project failures, money loss, schedule overruns, and low quality project outcomes. For instance, a Gartner report from 2015 [14] predicted that, until 2017, 60% of Big Data projects would have fail or would have not provide the expected benefits. Those projects would end up not going beyond piloting and experimentation, thus, resulting in abandonment. However, in November 2017, Nick Heudecker, a Gartner analyst, posted in his twitter account that his company was too conservative. The Big Data project failure rate as of November 2017 was now close to 85%. Then, an article from Infoworld [15] states that as of May 2019 the situation is the same, nothing has changed, the failure rates are still high.

The reasons are not only related to technology itself [14]. It is a mix of environmental, technological, and managerial problems. In fact, it is estimated that the lack of skills in organisations contributes 30% of the failure [16], for instance. Other reasons Big Data projects fail are: At the project level [17, 18]: missing link to business objectives, lacking Big Data skills, relying too much on the data, failing to convince executives, and poor planning; At the technical level [10]: Rapid technology changes, difficulty in selecting Big Data technologies

to address the systems and project requirements, complex integration between new and old systems, computation of intensive analytics, and the necessity of high scalability, availability and reliability, to name a few.

Further, as reported in [13] there was approximately a 80:20 split in the industry focus in favor of algorithms for analytics and infrastructure, thereby shortchanging the aspects of creating and evolving applications and services concerned with Big Data. This situation is also reflected in the scientific community where not much of the attention has been given to RE in the development of Big Data software applications. Since RE for Big Data applications is an emerging area, a clearer understanding is needed, separating requirements for infrastructures, analytic tools and techniques, and end-user applications [13].

1.1 Research Problem

The elicitation, specification, analysis, prioritisation and management of system requirements for large projects are known to be challenging. It involves a number of diverse issues, such as: different types of stakeholders and their needs, relevant application domains, knowing about product and process technologies, regulatory issues, and applicable standards. The advent of Big Data and, in turn, the need for software applications involving Big Data, has further complicated Requirements Engineering. In part, this is due to the lack of clarity in the RE literature and practices on developing Big Data software applications.

While Requirements Engineering has long been recognised as critical for downstream development of computer systems, the entire field is basically passive about how to deal with characteristics of data in the RE process in the development of Big Data software applications. With Big Data, characterised by the well-known V attributes, questions arise as to how to elicit, specify, analyse, and document system requirements. This problem is compounded by the fact that the RE field had no domain model (until recently) for Big Data systems depicting the various artefacts, activities, and relationships amongst them that, in turn, can be used to support RE specifications, product design, project decisions, and maintenance.

Therefore, the vision of this PhD thesis was to take a first step toward understanding RE

involving Big Data software applications. In particular, we sought to investigate RE practices, specifications, and artefact models in the context of such applications.

In the next sections, we further discuss the need for investigating RE involving Big Data software applications (see Section 1.2). We compare Big Data applications with other traditional data-centric applications such as Business Intelligence and Decision Supporting systems (see Section 1.3). Then, we overview the contributions of this thesis (see Section 1.4) and its structure (see Section 1.5).

1.2 Why RE for Big Data Software Applications?

With the common and constant presence of new paradigms and technology domains (e.g., big data, IoT, blockchain, etc.) the disciplines in the Software Engineering process (e.g., Requirements Engineering, Software Design, Software Testing, etc.) must evolve and improve their approaches in order to support the development of applications operating upon these new paradigms and technologies [12]. In the context of Big Data software application projects, specific RE methods and approaches are needed due to the following reasons:

- **Complexity:** Big Data applications are complex solutions with several dynamic components, being distributed computation nodes, networks, databases, middleware, and business intelligence layers [13]. Therefore, traditional RE tools, techniques, gathering artefacts and templates do not work very well for a Big Data projects [12].
- **Big Data Characteristics and System's Quality Requirements:** The V characteristics (e.g., Volume, Velocity, Veracity, and Variety) of Big Data pose significant challenges to achieving high system quality standards for security, performance, scalability, privacy and other quality requirements [19]. Example challenges are: (i) addressing system's scalability as data volume increases substantially [20]; (ii) addressing system's performance as the injection of real-time (velocity) data increases in a fast pace [10]; (iii) guaranteeing high levels of privacy while capturing, processing, analysing and visualising Big Data [21, 22]; and (iv) data governance and storage efficiency is affected due

to the volume and temporal aspect of Big Data. Thus, it is essential that while specifying the scenarios of desirable system responses, the characteristics of Big Data are represented in requirements notations so that software solution can be created to meet the specification [13].

- **Concept Drift and the Specification of Testable Big Data Requirements:** The idea of data analytics - when applying machine learning techniques for predictive analytics - is to identify the likelihood of future outcomes based on historical data [23]. Since programming models (e.g., machine learning algorithms) are built and trained based on existing old data, they no longer reflect the distribution of the incoming data, calling for constant updates to the model [24]. This is known, in statistics, as concept drift, which means that statistical properties of the target variable, which the model is trying to predict, change in unexpected ways, resulting in less accurate predictions [25]. This pose critical challenges in specifying testable requirements for Big Data software applications.
- **Myriad of Available Big Data technologies:** Requirements Engineering is a multidisciplinary, cross-functional discipline. It provides support across all phases of the software development process [26]. In supporting the design of Big Data applications, for instance, when it comes to eliciting the appropriate technologies to address the Big Data envisaged requirements, it is essential to consider how to select the existing technologies, frameworks, and software resources as well as the extent to which they help in addressing those requirements, both system and software. Because there are a large number of distributed systems frameworks and technologies available in the context of Big Data, the efficient specification of requirements can lead to a more accurate selection of those technologies and frameworks. The same holds true for the activity of selecting software resources such as external services. Those software resources often refer to a large variety of outsourceable functions to the cloud usually accessible through APIs (*application programming interfaces*) [27]. It can range from supply of data to the supply of analytical software tools [27, 28], for instance. 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 project [8].

1.3 Big Data Applications *versus* Traditional Data-centric Applications

The crucial differences between Big Data software applications and traditional data-centric applications (such as Business Intelligence - BI and Decision Support Systems-DSS) lie on aspects related to the types of data, data characteristics, and use of data [29]. For example:

- Traditional data-centric solutions such as BI solutions, carry the data to the processing functions [30] (e.g., the system employs a consistent set of metrics to measure both past performance and guide business future planning [29]), whereas Big Data solutions take the processing functions to the data (e.g., the system applies machine learning and analytics techniques to uncover insights from the data) [30].
- Traditional data-centric solutions are designed from the ground up to work with data that has previously been structured [31]. They are usually based on the principle of combining all business data sets into a central server. They have been traditional and successful with data that is much less huge and less varied [29], whereas Big Data solutions can process and analyze data in different formats, both structured and unstructured [32] stored on a distributed file system.
- Traditional data-centric solutions (such as DSS) primarily accentuate access to and manipulation of a time series of internal company data [29] whereas Big Data solutions emphasises on data from a variety sources.
- Big Data solutions can process historical and real-time data [32], whereas traditional data-centric solutions focuses on historical data [30].

Therefore, it is not possible to use same technologies (from a DSS or BI environments, for example) for data which is highly varied (from text, to logs, to multimedia), intensely complex, huge in terms of volume (from GB to PB), and unstructured in nature [30]. In summary, when traditional data-centric solutions are brought into the world of Big Data, they fail to perform as they cannot deal with the massive increase in the data volume, the eruption of cardinality and dimensions, and the large variety of data sources [33].

1.4 Research Contributions

The contributions of this thesis are organized into four core studies that are structured into four discrete chapters (chapters 3-6). Figure 1.1 depicts a profile of the contributions made in this thesis and their corresponding chapters. The overarching contributions are new, empirical knowledge on Requirements Engineering in the context of Big Data software applications. The contributions are:

- Analysis and discussion of the state of Requirements Engineering Research involving the development of Big Data software applications.
- Analysis and discussion of the results of a case study conducted within a Big Data software development project aimed at determining the RE practices and challenges in this type of projects.
- Identification and analysis of a set of RE challenges in creating and evolving Big Data software applications.
- Identification of the project artefacts (e.g., Big Data Technological Requirements Specifications, Big Data Software Requirements Specifications, Big Data Scenarios, etc.), organized into three groups of artefacts according to the Requirements Engineering Reference Model (Business Needs Artefacts, Systems Specification Artefacts, and Requirements Specification Artefacts), and connected by six types of inter-relationships, such as *“Is-Composed-Of”*, *“Is-derived-from”*, *“Is-part-of”*, *“Assist-in”*, *“Contains”*, and *“Used-In”*.
- Construction and validation of a RE Artefact model in the context of Big Data software development projects that depicts project artefacts and their inter-relationships based on empirical findings and evaluation.
- Approach for modelling quality requirements for Big Data applications composed of (i) a process, (ii) requirements logging templates, (iii) checklist, and (iv) a goal-oriented requirements modelling language.

- Modelling tool that implements the proposed goal-oriented requirements modelling language enabling end-users to model Big Data quality requirements.

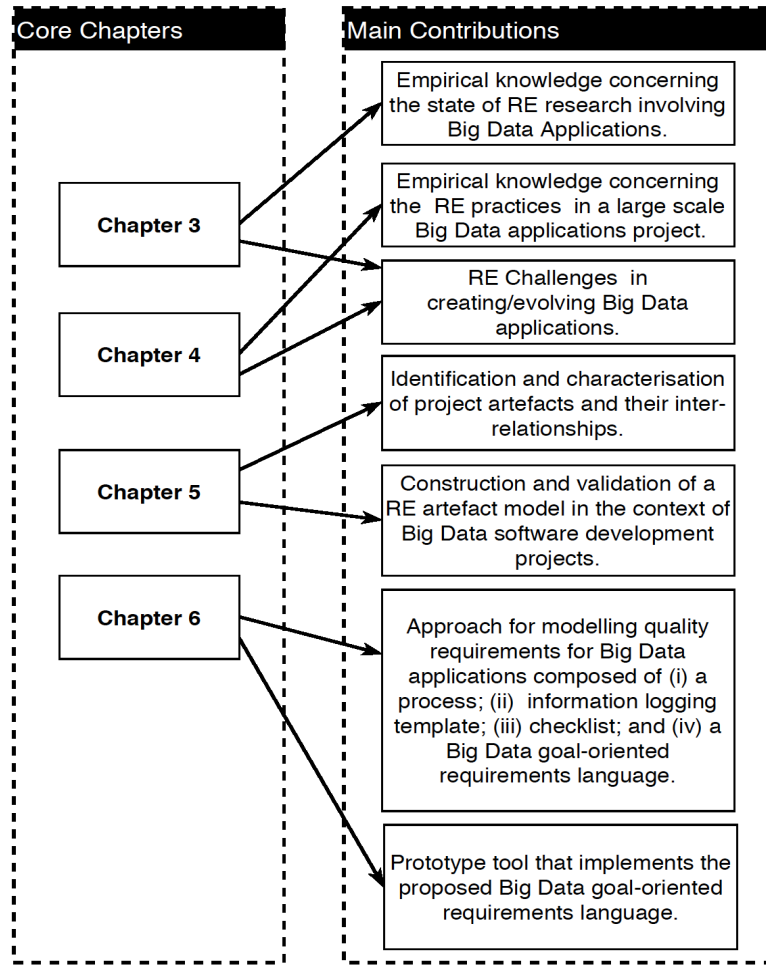


Figure 1.1: Thesis Contributions by Chapter

This figure depicts the core chapters of this thesis (on the left) and their associated contributions (on the right). The contributions are connected to their corresponding chapters by a simple association line. One chapter can be associated with one or more contributions.

1.5 Thesis Structure

This thesis is structured in the integrated article format. It is composed of eight discrete chapters. Figure 1.2 presents graphically the structure of this thesis and shows how each chapter relates to one another.

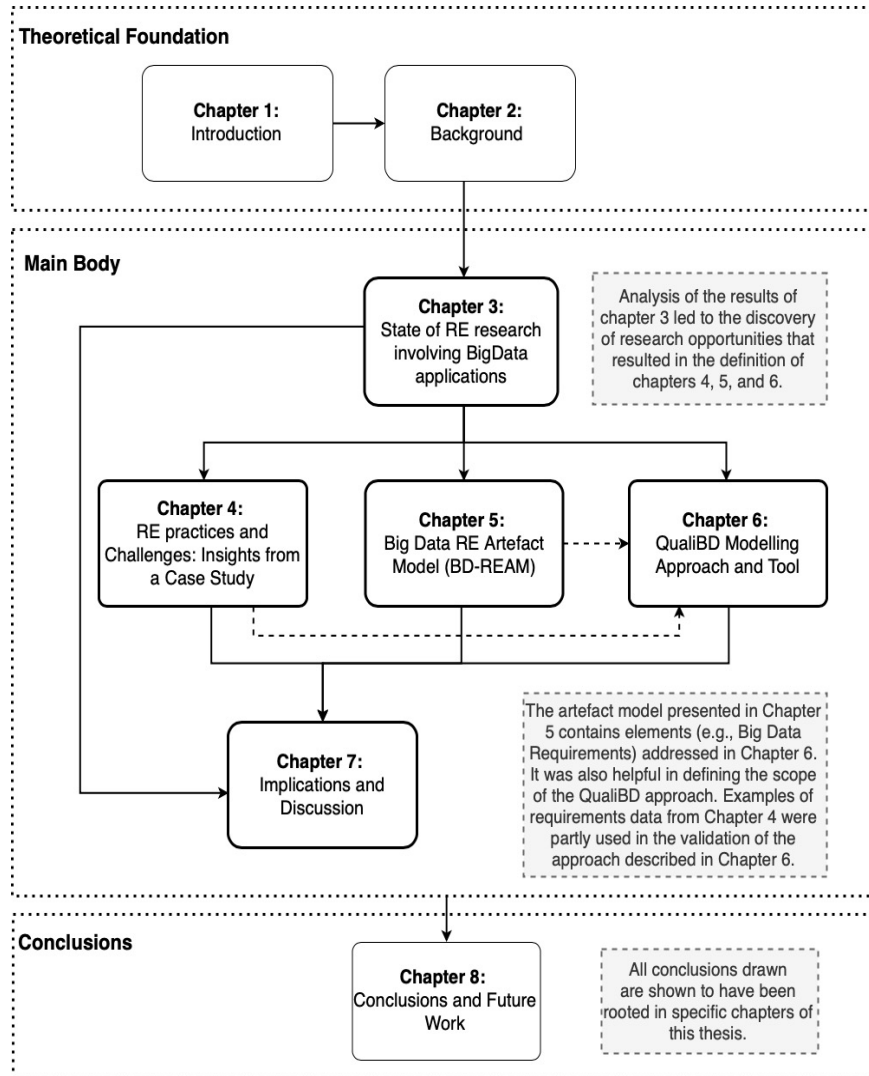


Figure 1.2: Thesis Structure

This figure depicts the overall structure of this thesis. It also shows how each chapter is related to one another. The chapters are represented by a regular shape with rounded corners. They are organized into three groups: theoretical foundation, main body, and conclusion. The core chapters of this thesis are represented with darker borders. The rectangular shapes coloured in grey represent clarification notes that have been provided to help better understand the structure provided in this figure.

With reference to Figure 1.2, we briefly overview the chapters composing this thesis:

Chapters 1 and 2 together form the theoretical foundation of this thesis. Chapter 1 introduces the topic as well as the research problem addressed in this thesis. Chapter 2 describes the background concepts. Chapter 3 describes the results of a SLR conducted with the aim to identify the current state of the Requirements Engineering research in the context of Big Data software applications as well as the existing research challenges and opportunities in this field.

Chapter 4 describes the results of an exploratory case study conducted within a large scale Big Data applications development project in the Oil&Gas domain with the aim to determine the current RE practices and challenges in such projects, currently bereft in the scientific literature.

In Chapter 5, we identify and characterise the several types of artefacts and their inter-relationships that exist in Big Data software applications projects. Based on this identification, we developed an artefact-model that depicts such characterisations. The model was first created based on knowledge from the scientific literature (chapters 2 and 3) and further validated internationally by practitioners working on Big Data Software projects in industry.

Chapter 6 described our proposed requirements approach modelling quality requirements for Big Data applications. The approach is composed of (i) a process, (ii) requirements logging template, (iii) checklist; (iv) a Big Data goal-oriented requirements language; and (v) a prototype tool that implements our proposed language. We further validate and illustrate the feasibility of this approach by modelling requirements collected from real Big Data applications development projects.

In Chapter 7, we discuss the implications the results of this research have in industrial practice, academic research, and tool support. We further discuss some of the contributions made in this thesis focusing on their limitations and potential for improvement and adaptability to other project settings and domains. Finally, Chapter 8 describes the conclusions drawn from the studies discussed in this thesis and identifies directions for future work.

With the exception of Chapters 1, 2, and 7, all the core chapters of this thesis have been published or submitted for publication in international scientific venues (e.g., conferences, workshops, and journals). Table 1.1 depicts the chapters of this thesis and their corresponding published/submitted articles.

Table 1.1: Core chapters and their associated submitted or published articles

Chapters	Associated Publications
Chapter 3: State of Requirements Engineering Research in the Context of Big Data Applications	<ul style="list-style-type: none"> — D. Arruda and N. H. Madhavji, “<i>State of requirements engineering research in the context of bigdata applications</i>”, in Requirements Engineering: Foundation for Software Quality, E. Kamsties, J. Horkoff, and F. Dalpiaz, Eds. Cham: Springer International Publishing, pp. 307 - 323, 2018.
Chapter 4: Requirements Engineering Practices and Challenges in the Context of a Big Data Software Development Project: Insights from a Case Study	<ul style="list-style-type: none"> — Submitted: Journal of Software and Systems (Elsevier).
Chapter 5: Empirically Derived RE artefact Model in the Context of Big Data Systems Development Projects	<ul style="list-style-type: none"> — D. Arruda, N. H. Madhavji, and I. Noorwali, “<i>A Validation Study of a Requirements Engineering Artefact Model for Big Data Software Development Projects</i>” in Proceedings of ICSOFT - International Conference on Software Technologies, pp. 106 - 116, 2019. — D. Arruda and N. H. Madhavji, “<i>Towards a Requirements Engineering Artefact Model in the context of Big Data Software Development Projects</i>” in Proceedings of the IEEE International Conference on Big Data, pp. 2232 - 2237, 2017.
Chapter 6: An Approach for Modelling Quality Requirements for Big Data Software Applications	<ul style="list-style-type: none"> — Under Review: Information and Software Technology Journal (Elsevier). — D. Arruda, and N. H. Madhavji, “<i>QualiBD : A tool for modelling Quality Requirements for Big Data Applications</i>” in Proceedings of the IEEE International Conference on Big Data, 2019. — D. Arruda, <i>QualiBD tool: Implementation details</i>, CoRR, vol. abs/1912.03866, 2019.[Online]. Available: http://arxiv.org/abs/1912.03866

Bibliography

- [1] B. Nuseibeh and S. Easterbrook, “Requirements engineering: A roadmap,” in *Proceedings of the Conference on The Future of Software Engineering*, ser. ICSE '00. New York, NY, USA: ACM, 2000, pp. 35–46.
- [2] R. S. Pressman, *Software engineering: a practitioners approach*, 2001.
- [3] D. Arruda and N. H. Madhavji, “QualiBD: A tool for modelling quality requirements for Big Data Applications,” *Proceedings of the IEEE International Conference on Big Data*, 2019.
- [4] D. Reinsel, J. Gantz, and J. Rydning, “The Digitization of the World From Edge to Core,” no. November, 2018.
- [5] V. D. Nadkarni, A. (2016) Worldwide big data technology and services forecast, 2016 - 2020. [Online]. Available: <https://www.marketresearch.com/IDC-v2477/Worldwide-Big-Data-Technology-Services-10510864/>
- [6] 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*, May 2015, pp. 19–25.
- [7] R. N. Laigner, M. Kalinowski, S. Lifschitz, R. S. Monteiro, and D. D. Oliveira, “A Systematic Mapping of Software Engineering Approaches to Develop Big Data Systems,” no. July, 2018.
- [8] V. Dipti Kumar and P. Alencar, “Software engineering for big data projects: Domains, methodologies and gaps,” *Proceedings - 2016 IEEE International Conference on Big Data*, pp. 2886–2895, 2016.
- [9] R. . Davenport. T.H., Bean. (2019) Big companies are embracing analytics, but most still dont have a data-driven culture. [Online]. Available: <https://hbr.org/2018/02/big-companies-are-embracing-analytics-but-most-still-dont-have-a-data-driven-culture>
- [10] H.-M. Chen, R. Kazman, S. Haziyevev, and O. Hrytsay, “Big Data System Development: An Embedded Case Study with a Global Outsourcing Firm,” *2015 IEEE/ACM 1st International Workshop on Big Data Software Engineering*, pp. 44–50, 2015.
- [11] D. Arruda and N. H. Madhavji, “Towards a Requirements Engineering Artefact Model in the context of Big Data Software Development Projects,” *Proceedings of the IEEE International Conference on Big Data*, pp. 2232–2237, 2017.
- [12] R. Narayanan, “Evolving and improving the requirements approach to big data projects: a roadmap to implementing big data projects.”

- [13] N. H. Madhavji, A. Miranskyy, and K. Kontogiannis, “Big Picture of Big Data Software Engineering: With Example Research Challenges,” *Proceedings - 1st International Workshop on Big Data Software Engineering, BIGDSE 2015*, pp. 11–14, 2015.
- [14] L. Goasduff. (2015) Gartner says business intelligence and analytics leaders must focus on mindsets and culture to kick start advanced analytics. [Online]. Available: shorturl.at/jrAIQ
- [15] A. Patrizio. (2019) 4 reasons big data projects fail and 4 ways to succeed: Nearly all big data projects end up in failure, despite all the mature technology available. here’s how to make big data efforts actually succeed. [Online]. Available: <https://www.infoworld.com/article/3393467/4-reasons-big-data-projects-fail-and-4-ways-to-succeed.html>
- [16] A. Sharala. (2019) Why 85% of big data projects fail. [Online]. Available: <https://www.digitalnewsasia.com/insights/why-85-big-data-projects-fail>
- [17] A. Miranskyy, A. Hamou-Lhadj, E. Cialini, and A. Larsson, “Operational-log analysis for big data systems: Challenges and solutions,” *IEEE Software*, vol. 33, no. 2, pp. 52–59, Mar 2016.
- [18] J. Heidrich, A. Trendowicz, and C. Ebert, “Exploiting big data’s benefits,” *IEEE Software*, vol. 33, no. 4, pp. 111–116, July 2016.
- [19] A. Suwalska, M. Breager, M. Brightwell, E. Koufakis, R. Martini, and P. Sollander, “High-availability monitoring and big data: Using java clustering and caching technologies to meet complex monitoring scenarios,” 10 2013.
- [20] I. Noorwali, D. Arruda, and N. H. Madhavji, “Understanding quality requirements in the context of big data systems,” *Proceedings of the 2nd International Workshop on BIG Data Software Engineering - BIGDSE ’16*, pp. 76–79, 2016.
- [21] H. Kupwade Patil and R. Seshadri, “Big data security and privacy issues in healthcare,” in *2014 IEEE International Congress on Big Data*, June 2014, pp. 762–765.
- [22] M. Jensen, “Challenges of privacy protection in big data analytics,” in *2013 IEEE International Congress on Big Data*, June 2013, pp. 235–238.
- [23] P. Woods. (2017) Why big data demands new technology. [Online]. Available: <https://www.itproportal.com/features/why-big-data-demands-new-technology>.
- [24] A. LHeureux, K. Grolinger, H. F. Elyamany, and M. A. M. Capretz, “Machine learning with big data: Challenges and approaches,” *IEEE Access*, vol. 5, pp. 7776–7797, 2017.
- [25] C. E. Otero and A. Peter, “Research directions for engineering big data analytics software,” *IEEE Intelligent Systems*, vol. 30, no. 1, pp. 13–19, 2015.
- [26] I. Sommerville, *Software Engineering*, 2009.
- [27] A. Yasin, L. Liu, Z. Cao, J. Wang, Y. Liu, and T. S. Ling, “Big data services requirements analysis,” in *Requirements Engineering for Internet of Things*, M. Kamalrudin, S. Ahmad, and N. Ikram, Eds. Singapore: Springer Singapore, 2018, pp. 3–14.
- [28] M. A. Serhani, H. T. E. Kassabi, and I. Taleb, “Quality profile-based cloud service selection for fulfilling big data processing requirements,” in *2017 IEEE 7th International Symposium on Cloud and Service Computing (SC2)*, Nov 2017, pp. 149–156.
- [29] V. Kale, *Enterprise Performance Intelligence and Decision Patterns*, 2018.

- [30] (2020) Business intelligence vs big data. [Online]. Available: <https://www.educba.com/business-intelligence-vs-big-data/>
- [31] G. Trujillo, R. Garcia, S. Jones, C. Kim, and J. Murray. (2015) Understanding the big data world. [Online]. Available: <http://www.pearsonitcertification.com/articles/article.aspx?p=2427073&seqNum=2>
- [32] M. Chen, S. Mao, and Y. Liu, "Big data: A survey," *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.
- [33] B. Desai. (2018) Understanding olap on big data: Why do you need it. [Online]. Available: <https://www.kyvosinsights.com/understanding-olap-on-big-data-why-do-you-need-it/>

Chapter 2

Background

This chapter presents background concepts used in this thesis organised into three sections. The first section provides an introduction to the field of Requirements Engineering and the remaining sections provide an overview regarding Big Data and Big Data Software Engineering, respectively.

2.1 Requirements Engineering

Requirements Engineering (RE) is the process of finding out, analysing, documenting and checking requirements and its constraints for a particular project [1]. It forms the ground for every software project, defining what the stakeholders need from it, and what it must do to satisfy that needs. In general, requirements offer support to (i) project planning, (ii) risk management, (iii) acceptance testing, and (iv) changing control [2]. However, practicing Requirements Engineering is a challenging and complex task. It involves (i) stakeholders with diverse backgrounds and levels of knowledge, (ii) different application domains, (iii) it is expensive and error-prone, (iv) it should be aligned with the business goals, to name a few [3]. In this section, we present the basic concepts and activities involved in Requirements Engineering process in the software development life-cycle.

2.1.1 Requirements Levels of Description

Some of the problems that arise during the requirements engineering process are a result of failing to make a clear separation between the various levels of description [1]. To deal with the diversity of software requirements, Sommerville [1] suggests organising them into two levels of description: (i) user requirements and (ii) systems requirements, described below. An illustrative example is provided in Figure 2.1 (extracted from Sommerville [1], p.84).

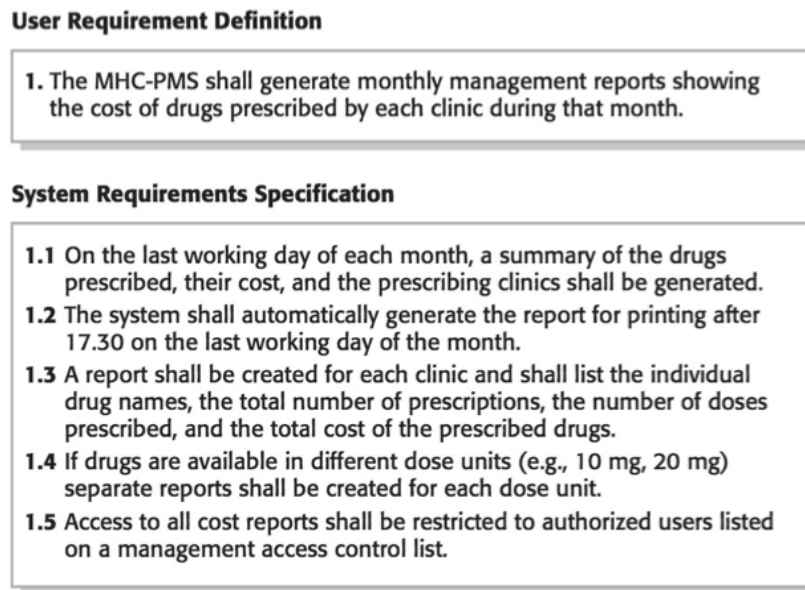


Figure 2.1: Example of User and Systems Requirements.

- **User requirements:** user requirements are statements (usually in a combination of natural language, UML-like diagrams, and mock outputs) of the services the system is expected to provide to system users and the constraints under which it must operate. User requirements for a system should describe the functional and non-functional requirements so that they are understandable by system users who don't have detailed technical knowledge [1]. Laplante [4] states that in many cases user stories can play the role of user requirements.
- **Systems requirements:** system requirements are more detailed descriptions of the software systems functions, services, and operational constraints. The system requirements

document should define exactly what is to be implemented [1]. These requirements are derived from analysis of the user requirements [4].

2.1.2 Types of Requirements

The requirements are analysed and described as functional requirements or non-functional requirements.

- **Functional Requirements:** describe what the system should do, how the system should react to particular inputs, and how the system should behave in particular situations [1]. Functional requirements are sometimes called behavioural or operational requirements because they specify the inputs (stimuli) to the system, the outputs (responses) from the system, and behavioural relationships between them [5].
- **Non-Functional Requirements/Quality Requirements:** these are constraints on the services or functions offered by the system. They include timing constraints, constraints on the development process, and constraints imposed by standards. Non-functional requirements often apply to the system as a whole, rather than individual system features or services [1].

2.1.3 Requirements Engineering Activities

The Requirements Engineering process is composed of the following activities: (i) requirements elicitation, (ii) requirements analysis and negotiation, (iii) requirements specification and modelling, (v) requirements validation and (vi) requirements management. In the following subsections, we briefly describe each of these activities.

Elicitation

Requirements elicitation, also known as requirements discovery, is one of the crucial tasks of the requirements engineering process, as it allows one to discover which requirements the users want to see incorporated into the system to be developed [6]. The requirements elicitation

process incorporates the following fundamental steps: (1) study the domain, (2) identify the requirements sources, (3) consult and engage the stakeholders, (4) select the techniques to be adopted (e.g., brainstorm, interview, questionnaire, scenarios, document analysis, etc.), and (5) elicit the requirements from the stakeholders and other identified sources.

Analysis

Requirements analysis is defined as the activity related to the refinement of stakeholders needs into formal product specifications [3]. Also, this activity aims to make informed decisions about concerns and issues raised in the elicitation process [7]. Issues and concerns with requirements include (i) they don't always make sense, (ii) they may be inconsistent, incomplete and vague, (iv) there may be unclear dependencies between requirements, to name a few [4]. Important to notice that many of the elicitation techniques (e.g., brainstorm, interview, questionnaire, scenarios, document analysis, etc.) are intended to avoid or alleviate these problems.

Specification and Modelling

Requirements specification is the process of writing down the user and system requirements in a requirements document. Ideally, the user and system requirements should be clear, unambiguous, easy to understand, complete, and consistent [1]. The input of the specification is a set of agreed statements of different types (e.g., general objectives, systems requirements, user requirements, relevant domain properties, etc.). The output of the specification is the first version of the requirements document [7]. Software requirements can be specified in natural language and modelled using diagrams and visualisations, for example. Proper requirements representation facilitates communication of requirements and translation into system's design [4].

Validation

Laplante [4] defines the requirements validation activity as the process of determining if the specification is a correct representation of the customers needs. The purpose of this activity is quality assurance [7]. Sommerville [1] explains that requirements validation usually overlaps

with analysis as it is concerned with finding problems with the defined requirements. Requirements validation is an important activity because errors in requirements documents can lead to a considerable amount of rework costs when these problems are discovered during development or after the system is in production [1, 7]. The techniques used to support requirements validation can include requirements review, prototyping and tests cases, and they can be used in conjunction with one another [1].

Management

The requirements management involves managing the realities of changing requirements over time [4]. It is to manage all of projects or products requirements post - elicitation, and to identify inconsistencies between those requirements and the project plan or work products [3]. It is important to keep track of individual requirements and maintain links between dependent requirements so that one can assess the impact of requirements changes in the software project. Also, its necessary to define a formal process for making change proposals and linking these to system requirements. The formal process of requirements management should start as soon as a draft version of the requirements document is available [1].

Negotiation and Prioritisation

Negotiation is, in its essence, a process of decision-making carried out in a context of strategic interaction or inter-dependency in the project [6]. Requirements negotiation should not be considered as a one-time task in a software project, but should be used early on and repeated in later stages. It contributes to the goal of defining feasible and mutually satisfactory requirements that accommodate all stakeholder goals and expectations [8]. In addition, in a software project, it is important to define the set of candidate requirements (i.e., the requirements that are susceptible to be incorporated in the system, due to their relevance from the onset). Then, the subset that includes the most important requirements must be selected. Requirements prioritization is a technique that aids in identifying those important requirements and that can be viewed as the process that sorts a set of requirements, according to various criteria defined by the project [6].

2.2 Goal-Oriented Requirements Engineering

Goal-oriented requirements engineering (GORE) is concerned with the use of goals for eliciting, elaborating, structuring, specifying, analysing, negotiating, documenting, and modifying requirements [9]. Yu and Mylopoulos [10] state that the notion of goals is increasingly being used in the field of RE. They further explain that goals are used to address various purposes (e.g., requirements acquisition, relating requirements to organizational and business context, clarifying requirements, dealing with conflicts, driving design, etc.) [9]. In essence, goals capture, at distinct levels of abstraction, the several objectives that the system under consideration should accomplish. The most common goal-oriented requirements approaches and frameworks are i^* [11], Kaos [12], and NFR framework [13]. These are general purpose frameworks that throughout the years have been either extended or served as foundation for the development of domain specific goal-oriented requirements approaches.

2.3 Big Data

In the past couple of years Big Data has caught the attention of industry interested in the high potential of Big Data, and many government agencies announced major plans to accelerate Big Data research and applications [14]. In a study conducted by CSC in 2014 with more than 300 IT employees revealed that approximately 52% of the respondents were involved in a Big Data project. Also, IDG Enterprise study presented by Columbus [15] reveals that 36% of the participated companies have plans to increase their budgets for data-driven initiatives within the organization. As the top priority, 61% of the respondents stated that the main goal in investing in data-driven initiatives within the organization is to improve the quality of the decision making as they considered Big Data Analytics as an important tool to accelerate and gain important business insight and value from data.

2.3.1 Defining Big Data

There has been a considerable effort towards the definition for the term “Big Data”, and according to Chen et al., [14], the effort arises from both industry and academia. The truth is that to date, there is no standard definition for the term “Big Data”. The existing Big Data definitions differ across different context and perspectives of use [16]. For example, in the context of infrastructure, Big Data can be defined as data with high volume, velocity, and variety, and unpredictability that require a scalable architecture for efficient storage, manipulation, and analysis [17]. Big data does not only cover huge data-sets themselves, but also space problems, technologies and opportunities to create business value [18]. In the analytics context, Otero and Peter [16] define Big Data as data so large that contains significant low probability events that would be absent from traditional statistical sampling methods. In the application’s perspective, it can be defined as any regular application that serve end-users except that underlying the software system is Big Data which the system operates upon [19]. Finally, from a business’s perspective, Big Data represents opportunities for achieving competitive advantage by making improved decisions [20].

2.3.2 Characterising Big Data

Big Data is characterised by the “V” attributes. The most common are volume, velocity, veracity and variety. *Volume* is related to the generation and collection of masses of data [20]. For instance, a study conducted by the International Data Corporation (IDC) predicts that 40 zettabytes (43 trillion gigabytes) of data will be generated by 2020 [21]. This represents an increase of approximately 300 times on the amount of data in 2005 [22]. *Velocity* means that the speed of growth and transfer of data are fast, and comes from different sources such as sensors, mobile devices, social media, to name a few [23]. *Veracity* is related to how accurate the data is for use, free of biases, noise and abnormalities [23]. *Variety* is related to the several different types of data, which can be structured, semi-structured and unstructured data [14].

More recently, other “V” attributes have been added to the Big Data domain such as (i) *value*: Substantial value can be found in processing Big Data, including understanding customers needs, targeting them accordingly, optimising processes, and improving machine or

business performances [24]; (ii) *variability*: In the context of Big Data, it can refer either the number of inconsistencies in the data (usually found by anomaly and outlier detection methods) or the multitude of data dimensions resulting from multiple disparate types data and sources [24]; (iii) *Visualisation*: It allows for a comprehensible visual representation of data patterns, enabling users to easily gather insights from Big Data [25]; (iv) *Validity*: Similar to veracity, validity refers to how accurate and correct the data is for its intended use [24]; (v) *Volatility*: refers to the time of validity of Big Data as well as the time the data should be stored and considered relevant for use [24, 26].

2.4 Software Engineering for Big Data Applications

The Systems and software engineering vocabulary described within the ISO/IEC/IEEE 24765 standard [27] Software Engineering (SE) as the “*systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software systems*”. As the field of SE has matured, it has developed a number of approaches to areas such as software requirements, design, testing, and maintenance [28]. Moreover, software development processes and methodologies such as waterfall, incremental development, spiral, and agile, for instance, have been successfully applied to produce quality software on time and within budget [28].

However, as new technologies domains and paradigms (e.g., IoT, Big Data, Artificial intelligence, etc.) gain popularity and are enabled by recent technological advances, the field of software engineering has to reinvent itself in order to adapt or define approaches and processes to support the engineering of applications underlying such paradigms. In the context of Big Data, for example, the Big Data systems development - also known as Big Data software engineering [29] - has a relatively short history, starting a trend in 2011 when the term was presented by IBM [14] with the years of 2013 and 2014 being then, the years of Big Data experimentation [14]. Big Data Software Engineering refers to the development of systems that incorporate Big Data in serving the end-users, for example, through features with which users interact [29]. However, engineering such applications pose significant challenges to the field of

software engineering (as described in the Introduction of this thesis) that must be investigated in order to advance this field of SE for Big Data applications.

2.5 Summary

In this chapter, we described some of the background concepts relevant to the research reported in this thesis. We first discussed the RE process, classification of requirements, and its associated activities. We further described some of the most common approaches, more specifically the goal-oriented requirements engineering approaches, used in the field of RE engineering to support the modelling and specification of systems requirements. Then, we provided an overview of Big Data, and its several definitions and characteristics. Finally, we provided a brief description of Software Engineering for applications that operates upon Big Data.

Bibliography

- [1] I. Sommerville, *Software Engineering*, 2009.
- [2] E. Hull, K. Jackson, and J. Dick, *Requirements Engineering*, 2011.
- [3] B. Berenbach, D. Paulish, J. Kazmeier, and A. Rudorfer, *Soft. Systems Requirements Eng. In Practice*, 2009.
- [4] P. A. Laplante, *Requirements Engineering for Software and Systems*, 2nd ed. Boston, MA, USA: Auerbach Publications, 2013.
- [5] R. S. Pressman, *Software engineering: a practitioners approach*, 2001.
- [6] J. M. Fernandes and R. J. Machado, *Requirements in Engineering Projects*, 1st ed. Switzerland: Springer International Publishing, 2016.
- [7] A. v. Lamsweerde, *Requirements Engineering: from system goals to UML models to software specifications*, 2009.
- [8] P. Grünbacher and N. Seyff, *Requirements Negotiation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 143–162.
- [9] A. van Lamsweerde, “Goal-oriented requirements engineering: a guided tour,” in *Proceedings Fifth IEEE International Symposium on Requirements Engineering*, Aug 2001, pp. 249–262.
- [10] E. Yu and J. Mylopoulos. (1998) Why goal-oriented requirements engineering. [Online]. Available: <https://www.cs.toronto.edu/pub/eric/REFSQ98.html>
- [11] E. Yu, “Modelling strategic relationships for process reengineering,” Ph.D. dissertation, University of Toronto, 1995. [Online]. Available: <https://pdfs.semanticscholar.org/2b5c/ea3171e911c444f8253c68312d93c1545572.pdf>
- [12] E. Kavakli, “Goal-oriented requirements engineering: A unifying framework,” *Requirements Engineering*, vol. 6, no. 4, pp. 237–251, 2002.
- [13] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, *The NFR Framework in Action*. Boston, MA: Springer US, 2000, pp. 15–45.
- [14] M. Chen, S. Mao, and Y. Liu, “Big data: A survey,” *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.
- [15] L. Columbus. (2015) Data analytics dominates enterprises’ spending plans for 2015. [Online]. Available: <https://www.forbes.com/sites/louiscolumbus/2015/03/15/data-analytics-dominates-enterprises-spending-plans-for-2015/#12613a951801>

- [16] C. E. Otero and A. Peter, "Research directions for engineering big data analytics software," *IEEE Intelligent Systems*, vol. 30, no. 1, pp. 13–19, 2015.
- [17] NIST Big Data Public Working Group: Use Cases and Requirements Subgroup, "NIST Big Data Interoperability Framework: Volume 3, Use Cases and General Requirements," vol. 3, p. 260, 2015. [Online]. Available: <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-3.pdf>
- [18] R. Mahanti, "Critical success factors for implementing data profiling: The first step toward data quality," *Software Quality Professional Magazine*, 2014.
- [19] D. Arruda and N. H. Madhavji, "State of requirements engineering research in the context of big data applications," in *Requirements Engineering: Foundation for Software Quality*, E. Kamsties, J. Horkoff, and F. Dalpiaz, Eds. Cham: Springer International Publishing, 2018, pp. 307–323.
- [20] H. Kościelniak and A. Puto, "Big data in decision making processes of enterprises," *Procedia Computer Science*, vol. 65, pp. 1052 – 1058, 2015, international Conference on Communications, management, and Information technology (ICCMIT'2015).
- [21] J. Thomas. (2015) Where is the world supposed to put all of its data? [Online]. Available: <https://www.forbes.com/sites/ibm/2015/02/17/where-is-the-world-supposed-to-put-all-of-its-data/#4cb7b2cf1483>
- [22] L. Lambert. (2017) Why 2017 will be the year of big data. [Online]. Available: <https://themarketmogul.com/big-data-2017/>
- [23] K. Normandeau. (2013) Beyond volume, variety and velocity is the issue of big data veracity. [Online]. Available: <https://insidebigdata.com/2013/09/12/beyond-volume-variety-velocity-issue-big-data-veracity/>
- [24] G. Ferican. (2017) The 10 vs of big data. [Online]. Available: <https://tdwi.org/articles/2017/02/08/10-vs-of-big-data.aspx>
- [25] P. Rubens. (2017) Big data visualization. [Online]. Available: <https://www.datamation.com/big-data/big-data-visualization.html>
- [26] (2017) Veracity: The most important v go big data. [Online]. Available: <https://www.gutcheckit.com/blog/veracity-big-data-v/>
- [27] "Iso/iec/ieee international standard - systems and software engineering–vocabulary," *ISO/IEC/IEEE 24765:2017(E)*, pp. 1–541, Aug 2017.
- [28] T. Arndt, "Big data and software engineering: prospects for mutual enrichment," *Iran Journal of Computer Science*, vol. 1, no. 1, pp. 3–10, Apr 2018.
- [29] N. H. Madhavji, A. Miransky, and K. Kontogiannis, "Big Picture of Big Data Software Engineering: With Example Research Challenges," *Proceedings - 1st International Workshop on Big Data Software Engineering, BIGDSE 2015*, pp. 11–14, 2015.

Chapter 3

State of Requirements Engineering Research in the Context of Big Data Software Applications

3.1 Introduction

Big Data application systems, like traditional applications, serve end-user needs except that underlying the software system is Big Data which the system operates upon. In comparison to traditional software development where the development processes are usually well-established, the processes for the development of applications involving Big Data are not clear just yet from the scientific literature given the nature of computing involved and data characteristics such as volume, variety, veracity, and velocity. In exploring the scientific literature on Big Data Software Engineering, it is difficult to fail to notice that not much attention has been given to RE in the development of Big Data applications. This situation motivated us to formally conduct a systematic literature review (SLR) [1] of RE research in the context of Big Data applications and synthesise any insight for further research in this domain.

Following deliberations, we arrived at the following core points to be used in this investigation: (i) types of requirements and (ii) activities of the RE process addressed in Big Data RE research; (iii) RE research challenges identified in the literature; (iv) application domains

covered; and (v) any advances made in the area (e.g., RE solutions proposed in the development of Big Data applications). The types of requirements would give an insight into where the emphasis lies (e.g., functionality, quality, data, etc.). The activities would give an idea of the extent of coverage of the RE process. The RE challenges highlight the documented dark alleys of this emerging field. The application domains give an insight into practical areas of foray with Big Data and RE. Finally, advances describe the knowledge and technology gains made to date by the research community. While one may find complementary points to add to this core, in this investigation we felt that the listed set of core points cover a significant ground in the RE field. The implications of the results of this study are anticipated for research as the gained knowledge will be a step forward to a better understanding of the actual state of the RE research involving Big Data applications.

This rest of this chapter is organised as follows: Section 3.2 discusses the research methodology. Section 3.3 presents the descriptive data. The results are discussed in Section 3.4. Section 3.5 provides some recommendations for further research. Section 3.6 discuss threats to validity of this study. Section 3.7 summarises this chapter. Finally, section 3.8 provides an addendum to this chapter in order to present some of the papers identified from 2018 to 2019 which is the period after this chapter has been published.

3.2 Research Methodology

In this section, we present the methodological procedures followed in this study. We adapted and followed the steps for conducting a SLR proposed in [1]. The following are described: (i) research questions, (ii) search strategy, (iii) selection criteria, (iv) data extraction and, (v) the selection process.

3.2.1 Research Questions

We ask the following main research question: **What are the early signs of the ways Big Data applications are treated in Requirements Engineering (RE)?** As described in the Introduction section, informal observations and exploration of the literature made it compelling

to investigate further the state of RE research in this domain. We thus decomposed the overall question into the following constituent questions:

- Q1. What are the activities in the RE process, types of requirements and application domains targeted by the identified RE research involving the development Big Data applications?
- Q2. What are the RE research challenges in the context of Big Data applications?
- Q3. What solutions have been proposed in the domain of RE and Big Data applications?

In section 3.4, the chapter explores the answers to each of these questions.

3.2.2 Search Strategy

This study focused mainly on searches in electronic databases such as ACM Digital Library, Science Direct, IEEE Xplore and Scopus as they index a considerable amount of papers published in conferences, journals and workshops proceedings - including the Big Data and RE conferences (e.g., IEEE Big Data, Big Data Congress, RE conference, etc.).

In order to use the electronic databases in a way they would return relevant results we defined and used the search terms (e.g., Big Data, requirements engineering, elicitation, analysis, specification, validation, negotiation, prioritization, management) related to the research topic of this chapter. We performed various searches using different combinations of search terms before deciding upon a final version of the search string. We observed that, when using the search string without the word requirements preceding each term, the number of irrelevant papers were greater. For example, many papers related to Big Data but not to Big Data software and requirements engineering, used terms such as analysis, validation and negotiation to convey different ideas (e.g., data analysis, Big Data negotiation, etc.) from the focus of this study - the Requirements Engineering aspect of it.

To ensure that the literature review adheres to the topic of this study Requirements Engineering for Big Data Applications, we decided to add the term requirements preceding each search term in our search string. The final version of the search string used for this review is:

(*“Big Data” AND (“Requirements Engineering” OR “Requirements Elicitation” OR “Requirements Specification” OR “Requirements Analysis” OR “Requirements Validation” OR “Requirements Negotiation” OR “Requirements Prioritization” OR “Requirements Management”)*))

Moreover, while performing the search for relevant papers using the databases commented above, we kept (manually) searching for scientific works in specific Big Data and Software Engineering conferences proceedings (such as International Conference on Software Engineering, RE International conference) and journals (such as IEEE Transactions on Software Engineering, Empirical Software Engineering, Journal of Systems and Software, IEEE Transactions on Big Data, Journal of Big Data, Big Data Research, The Services Transactions on Big Data and the Requirements Engineering Journal). The manual search consisted of accessing specific journals and conferences proceedings so as to search for relevant results. If the venue (journal or conference) website provides a search engine, we then searched for specific terms such as Big Data in order to identify possible results. Otherwise, we checked the Table of Contents and abstracts with the aim to identify relevant papers.

3.2.3 Selection Criteria

For this review, we set the following selection criteria: (i) studies must be in paper/article/chapter formats, (ii) must be written in English, and, (iii) must address any aspect of RE in the context of Big Data software applications.

3.2.4 Selection Process

The selection process - adapted from [1] - used in this study is composed of three steps. In step 1, the results were filtered by their title and abstract. The papers considered relevant for this study were selected, and in step 2 analysed by reading their introduction and conclusion sections. The papers deemed pertinent to the context of this research were potentially chosen

for the next step (step 3), which consisted of reading the entire paper. Then, a final list of selected papers was created, and all the relevant information was logged into the appropriate data extraction documents.

3.2.5 Data Extraction

In order to better organise the selected papers included into the SLR, a document composed of the following attributes was used: study id, title, authors, source, year of publication, full reference and the designated questions they address as well as important statements to help to answer the defined questions. Also, we created and used a spreadsheet to log important information (such as types of requirements, type of research, contributions, venues of publication, etc.) that helped in the descriptive analysis.

3.3 Descriptive Data and Analysis

During the automatic search, a total of 311 papers were identified. However, it is important to note that, as also pointed out by Kitchenham and Charters [1], initial SLR searches tend to result in many irrelevant papers. For example, in this study, numerous papers appeared in the search results because these papers contained terms such as Requirements Engineering or Big Data but they did not actually address any aspect of RE in the context of Big Data applications. After applying the selection criteria and reading the title and abstract (in step 1 see subsection 3.2.4 for the three-step process), only 24 papers were considered relevant. In step 2, the resultant papers were examined by reading their introduction and conclusion; thirteen papers were deemed relevant. Note that in these steps, if the cumulative information analysed till then in a paper was not decisive as for relevance, we then scanned internal sections of the paper to determine whether or not it addressed the topic of this SLR. Thus, we anticipate minimal false negative cases in the selection process. Additionally, a total of five papers were selected during the manual search based on the selection criteria as well. These papers were carried out to the final step in the selection process (step 3) which consists of reading the entire paper. In the end of the selection process, 14 papers [2–15], were considered relevant to be

used in our investigation. Figure 3.1 presents the selection process and the number of results for each step.

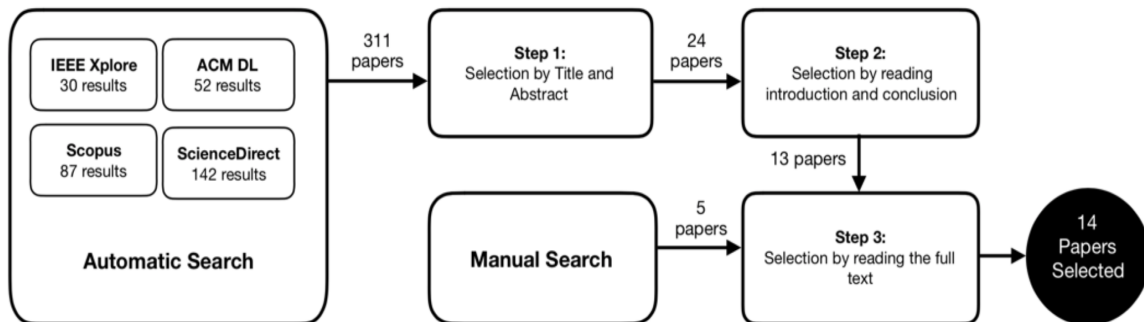


Figure 3.1: Distribution of papers identified and selected distributed by phased of the selection process

Figure 3.2 shows the number of selected papers by venue of publication. Table 3.1 shows their distribution by year. Most of the selected papers were published in 2015. Together, 2014 and 2016 represent six of the published papers. The years of 2013 and 2017, are represented by one and two papers, respectively. Regarding the venue of publication, the majority of the papers were published in workshops and conferences proceedings (four papers in each venue). Two studies were published as chapters in books and two other papers were published in journals. One study was published in a magazine (RE magazine by the International Requirements Engineering Board - IREB) [4] and another study was published online in a report format by the NIST Big Data Public Working Group [11]. The complete list of the venues of publication is presented in Table 3.2.

Table 3.1: Number of Papers by Year

2013	2014
1 (7%)	3 (21%)
2015	2016
5 (37%)	3 (21%)
2017	
2 (14%)	

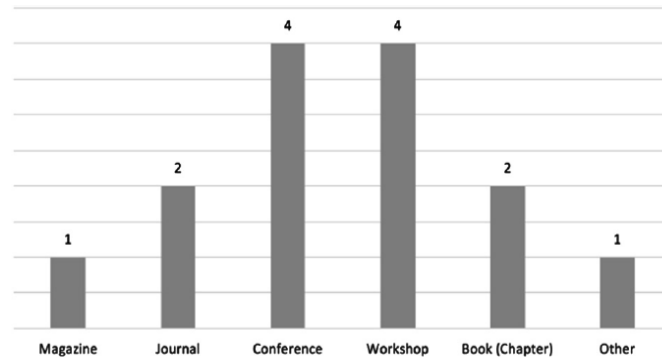


Figure 3.2: Distribution of papers by venue of publication

Table 3.2: Publication Venue and Number of Papers from each Venue

Publication Venue	Issue, Volume or Year	Paper Count
Conferences		
IEEE International Congress on Big Data	2013	1
International Conference on Data and Software Engineering	2014	1
International Conference on Cloud Computing, Data Science & Engineering	2017	1
IEEE International Conference on Big Data	2017	1
Workshops		
IEEE/ACM International Workshop on Big Data Software Engineering	2015	2
	2016	1
International Workshop on Quality-Aware DevOps	2016	1
Journals		
International Journal of Ambient Systems and Applications	Vol. 2, No. 2/2014	1
IEEE Intelligent Systems	Vol. 30/2015	1
Books and Magazines		
Studies in Big Data Springer	Vol. 05/2014	1
New Trends in Databases and Information Systems - Springer	Vol. 539/2015	1
Requirements Engineering Magazine	Issue 2016-01	1
Other (Online Publications)		
NIST Special Publication	Vol. 3/2015	1
Total		14

The papers selected were also classified with respect the type of research they present. For such classification, we used the classification for RE research proposed by Wieringa et al. [16] which consists of the following classes of papers:

- *Evaluation Research*: refers to the investigation of a RE problem or the implementation of a RE technique in practice. In this case, the novelty of the technique is not a criterion by which the paper should be evaluated.
- *Proposal of Solution*: refers to the proposal of solution technique that argues for its relevance, but without being validated.
- *Validation Research*: in this type of research the properties of a solution that has not been implemented in practice is investigated and analysed.
- *Philosophical Papers*: presents a new way of looking at existing things, a new conceptual framework, etc.
- *Opinion papers*: These types of papers present the authors opinions regarding an existing problem/issue.
- *Personal Experience Papers*: in these types of research, the emphasis is on what and may concern to multiple projects. It also must be the authors personal experience. It is also important that the paper provides the reader with a set of lessons learnt by the authors from their experience.

The overall distribution of papers by type of research and their contribution is presented in Tables 3.3 and Table 3.4. Also, we analysed the selected papers with respect to their contribution and research type organised by the RE activities they addressed (see Figure 3.3).

Table 3.3: Overall distribution of papers by type of research

Type of Research	Paper Citation	Paper Count
Evaluation Research	[2, 6, 14]	3
Proposal Solution	[8–11, 13, 15]	6
Validation Research	[5, 7]	2
Philosophical Papers	[3–5]	3
Opinion Papers	–	–
Experience Papers	–	–

Table 3.4: Overall distribution of papers by research contribution

Type of Contribution	Paper Citation	Paper Count
Method/Approach	[2, 7, 8]	3
Model	[14, 15]	2
Tool	[5, 10]	2
Framework/Architecture	[6, 9, 11, 13]	4
Processes/Methodologies	–	–
State-of-the-art	[3, 4, 12]	3

3.3.1 Discussion

One observation from the results of this study is that, surprisingly, the RE conferences such as the RE Conference and the International Working conference on Requirements Engineering: Foundations for Software Quality (REFSQ) and the Requirements Engineering Journal have not yet published papers on aspects of RE in context of the development of Big Data applications.

For instance, in conducting manual searches for Big Data related publications in the Requirements Engineering Journal, we found only one result matching with the term Big Data. However, the resultant paper does not deal with RE for Big Data applications; it simply used the term Big Data within the paper. In regard to the searches of the RE and REFSQ conferences, we analysed proceedings (title and abstract) from 2009 to 2017, since Big Data was not widely known in previous years.

Regarding the REFSQ proceedings, we did not find any papers discussing Big Data. For the RE Conference proceedings, we found one talk abstract from 2016, as well as a paper from 2017. However, the paper does not address any aspect of RE for the development of Big

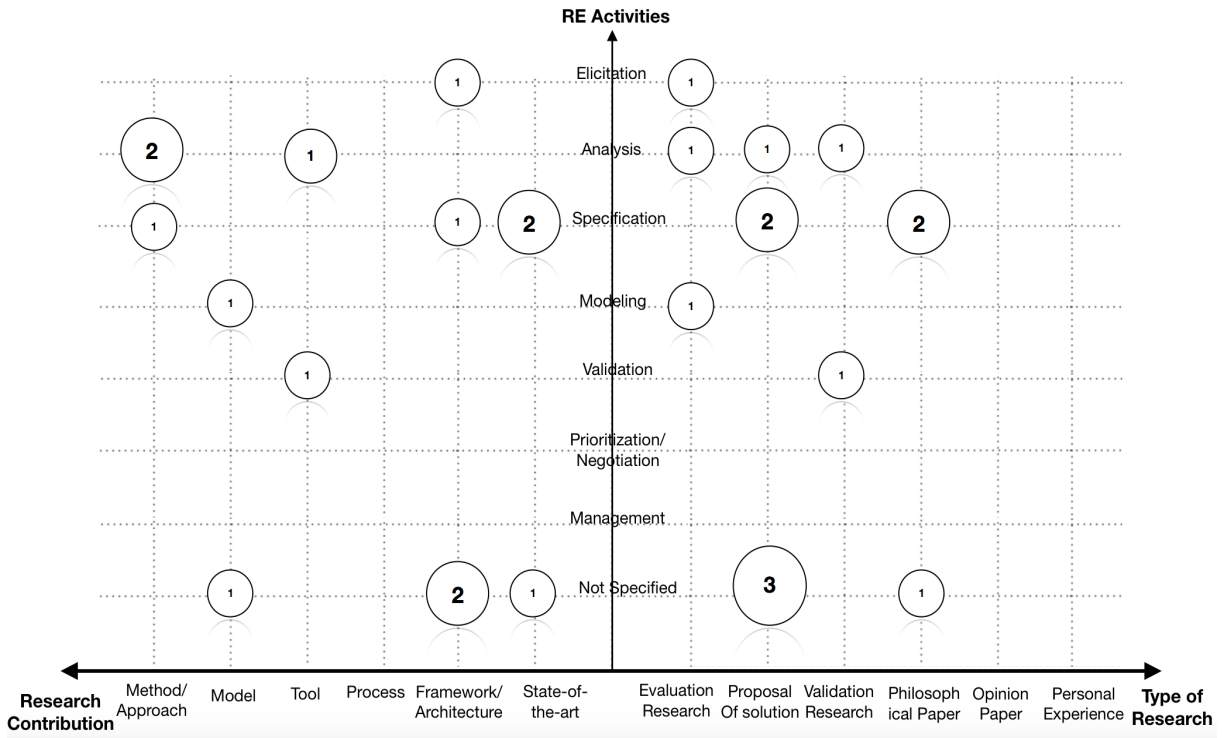


Figure 3.3: Papers by contribution and type of research organised according to the RE activities they address

Data applications. Instead, as is the case with the RE Journal resultant paper, it used the term Big Data within the text. Thus, no papers were selected from these sources to be used in this SLR. It is important to note that (repeated from section 3.3 for convenience), if the information analysed (title and abstract) in a paper was not decisive as for relevance, we then scanned internal sections of the paper to determine whether it should be included in this study.

Next section presents the results and discussion of this investigation.

3.4 Results and Discussion

Research aimed at addressing RE in the context of Big Data applications is currently at an early stage. In this section, we discuss the results of this study with the aim to provide the state of the art of RE research in the context of Big Data applications. To answer to the main question of this chapter, we have taken a close look at the selected papers with respect to the secondary

research questions represented by the following core points (repeated from subsection 3.2.1 for convenience): (i) types of requirements, activities in the RE process and application domains they address, (ii) RE research challenges, and (iii) RE solutions that have been proposed in the context of Big Data applications. The subsections that follow discuss these core points.

3.4.1 (Q1) What are the activities in the RE process, types of requirements and application domains targeted by the identified RE research involving the development of Big Data Applications?

As presented in Table 3.5, with regards to the activities in the RE process they discuss, most of the papers selected discussed either the analysis (three papers) or specification (four papers) phases. Elicitation, modelling and validation were discussed by only one study each. No papers were found discussing requirements negotiation, prioritization and management in the context of Big Data applications. Also, our analysis shows that the RE research involving Big Data applications fell into one of the following application domains: Healthcare, Biomedical Research, Government, Marketing, IT/Telecom, Astronomy and Physics, Earth, Environmental and Polar Science, Defense, commercial, and Social Media.

Unfortunately, none of the selected papers actually discusses the applicability or details on how to deal with Big Data requirements for a specific domain. However, in [11], use case descriptions were collected from various contributors within different application domains and used to derive a set of generic requirements for Big Data applications. Overall, the selected papers discussed - to some extent - functional, quality and data requirements. Important to note that one paper could have discussed one or more types of requirements. Therefore, the sum of the papers presented in Table 3.5 can be greater than the total number of papers selected in this review.

Functional Requirements. As well-known in the literature, functional requirements (FR) describe what the system should do, how the system should react to particular inputs, and how the system should behave in particular situations [17]. That wouldnt be different in the RE research involving Big Data software applications. From our analysis, the selected papers discussed the importance of addressing functional requirements for Big Data software appli-

Table 3.5: Types of Requirements, Activities of the RE process targeted by available RE and Big Data Research

RE Activities	Citation	Requirements Type	Citation
Elicitation	[6]	Functional Requirements	[3, 6, 12, 14, 15]
Analysis	[2, 7, 10]	Quality Requirements	[3, 5–8, 10, 12–15]
Specification	[3, 8, 9, 12]	Data Requirements	[9, 11]
Modelling	[14]	Architecturally Significant Requirements	[2]
Validation	[5]	Not Specified	[4]

cations. However, very few studies (two papers) actually provided examples of functional requirements. Also, these examples relate to generic functional requirements any Big Data application should address. For instance, extracted from [11] and [14]: (i) database capacity; (ii) data properties (e.g., system should check the completeness and accuracy of the data); (iii) backup routines; (iv) domain specific FRs (not discussed in detail); (v) data transformation (e.g., Needs to support batch and real-time analytic processing), (vi) data source (e.g., Needs to support slow, bursty, and high-throughput data transmission between data sources and computing clusters).

Quality Requirements. Basically, the selected papers discuss the following quality attributes a Big Data system must address: privacy and security [10, 11, 14] performance [7, 14]; availability [2, 8]; scalability, consistency, elasticity and low latency [2]. While some papers (10 papers) discuss the quality attributes for Big Data applications and others propose solutions to deal with quality requirements in the development of Big Data applications only one study actually gave examples (e.g., Req1: System needs to protect and preserve security and privacy of sensitive data) of security and privacy requirements [11].

Data Requirements. Having the right specification of data requirements is important for defining some of the systems functional requirements (e.g., systems needs to support diversified output file formats for visualization, rendering, and reporting; systems needs to support legacy, large, and advanced distributed data storage [11], etc.). In our investigation, only two papers [4, 9] discussed the necessity of selecting the right type of data as well as the data properties that must be taken into consideration when eliciting and specifying data requirements (e.g., data size, data types, file formats, rate of growth, at rest or in motion). However, none of

them actually provided concrete examples of what a data requirement looks like. In [4], two different templates that can be used to support the definition of data requirements are presented: (a) template for sourcing the data and, (b) a template to match the business problems with the data. In [9], a requirements specification framework for Big Data collection is proposed (section 3.4.3).

In the next section, we present and discuss some of the RE research challenges identified in this review.

3.4.2 (Q2) What are the Requirements Engineering Challenges in the context of Big Data Applications?

Four papers were the source of the research challenges in the context of Big Data applications [3, 8, 12, 14]. Basically, the challenges identified in this review are related to the necessity to understand and take the Big Data specific characteristics (such as volume, velocity, variety, etc.) into consideration while dealing with the systems requirements. Examples of the challenges are presented below.

Big Data Characteristics: The need to properly address the Big Data V-characteristics in the definition, analysis and specification of both functional and quality requirements [3, 8, 14]. It is essential that while eliciting the scenarios of desirable system responses, the characteristics of Big Data are represented in requirements notations so that solution design can be created to meet the specifications [3]. Notwithstanding, it is also important that these data characteristics are defined along with the systems quality attributes (in the specification of quality requirements) as it is believed to be complementary set of properties. For example [8]: the system shall use a stream-processing engine with a latency of 0.5 2.0 seconds (e.g., Storm, S4, Spark or Samza) to process data in real-time between global earthquake sensors and the data centre. This requirement addresses both velocity (data characteristic) and performance (quality attribute); two commonly discussed issues in the context of Big Data systems.

Writing verifiable requirements: The need to specify verifiable requirements. In [12], it is explained that Big Data Analytics applications faces concept drift, which means that statistical properties of the target variable, which the model is trying to predict, change over time in

unforeseen ways, thus causing predictions to become less accurate as time passes. Therefore, one of RE problems for Big Data Analytics applications is to be able to define and specify verifiable (testable) requirements [12].

Intuitively, it appears that there are more challenges and issues related to the RE activities involving the development of Big Data applications than what might appear from our review. Further empirical studies are clearly needed to uncover more facts.

3.4.3 (Q3) What Solutions have been proposed in the domain of RE and Big Data Applications?

The technical solutions identified in our investigation are presented in Table 3.6 and discussed below. These solutions are organised into three groups: (a) Approaches, Methods and Models, (b) Architectures and Frameworks and, (c) tools.

Table 3.6: Overview of the solutions proposed in RE and Big Data Research

Solutions Proposed	Citation
Approaches, Methods and Models	
Big Data System Design method	Chen et al., [2]
Approach for handling non-functional requirements for Big Data projects in scrum	Sachdeva and Chung [7]
Approach for analysing and specifying Quality Requirements	Noorwali et al., [8]
RE Generic model based on I* and KAOS	Eridaputra et al., [14]
RE Artefact Model in the Context of Big Data Software Projects	Arruda and Madhajvi [15]
Architectures and Frameworks	
Descriptive Architecture for Big Data Requirements Elicitation	Lau et al., [6]
Requirements Specification framework for Big Data Collection	Al-Najran and Dahanayake [9]
NIST Interoperability Framework*	NIST [11]
Framework with security constraints	Youssef [13]
Tools	
Verification Tool	Bersanini et al., [5]
UML extension for privacy requirements analysis	Jutla et al., [10]

With reference to Table 3.6, we present an overview of the solutions identified in this study:

Methods, Models, and Approaches. In [2], a Big Data System design method is proposed - an attempt to systematically combine architecture design with data modelling approaches in development of Big Data Systems. Even though this method is not specific for RE but for system design, it incorporates a RE step for requirements analysis which is composed of the following activities: (i) identification of business goals, (ii) identification of constraints,

concerns and drivers, (iii) identification of quality attribute scenarios and, (iv) definition of Big Data architecture scenarios based on the quality attributes scenarios identified. Also, this method suggested a Big Data template for logging data information (e.g., data source quality, data variety, data volume, velocity, read/write frequency, time to live, queries, etc.). The resultant requirements should be used to drive the design of Big Data systems.

In [7], an approach composed of two processes for dealing with both privacy and performance requirements for IoT and Big Data projects in scrum is proposed. In the security side, the problems with dealing with security requirements is that they are commonly treated as soft goals and thereby there's no clear way of defining if they are met or not. In the performance side, the authors argue that the problem of handling performance requirements for IoT and Big Data applications is that it is treated as a qualitative measure rather than a quantitative one. To solve both problems, the use of clear user stories acceptance criteria in scrum is proposed. The authors argue that this approach helped to introduce the quality requirements such as security and performance in early stage of the software development process and help to define clear parameters for the measurement of both security and performance requirements.

In [8] an approach for analysing and specifying quality requirements for Big Data Applications is proposed. The main idea is to intersect a Big Data characteristic with a quality attribute (e.g., variety security). This approach incorporates three elements - Big Data characteristic, quality attributes, and quality requirement description and helps to ensure that the Big Data characteristics are addressed in the specification of quality requirements.

A Requirements specification generic model using i^* framework and KAOS approach was described in [14]. In this work, the authors tried to elicit generic requirements for Big Data based on the data characteristics (e.g., Volume demands improved storage capacity; Velocity demands Database tools with high performance, etc.). Then, the elicited requirements were modelled using i^* framework and the KAOS approach. The models resulting from i^* and KAOS tools can then be used as references in the modelling of both functional and quality requirements for Big Data applications. These models were applied to a case study conducted at the Indonesian government agency for development planning of West Java and, according to the authors, the results demonstrated that the models can be used to create valid software requirements specifications for Big Data applications.

In [15], a Requirements Engineering artefact model in the context of Big Data Software development projects is proposed. The model depicts the RE artefacts and inter-relationships involved in the development of Big Data Software applications. It is argued that this type of model can be used as a reference for the design of project-specific processes, software maintenance, and for supporting project decisions throughout the entire product life-cycle, currently bereft in the Big Data RE research.

Architectures and Frameworks. In [6], a conceptual descriptive architecture to help understand the user requirements and system characteristics of Big Data Analytics software is proposed. This architecture was developed as a high-level specification of how the numerous tools might work together in a Big Data Analytics platform. To develop this conceptual architecture, the authors applied sense-making models (e.g., iterative cognitive process that the human performs to build up a representation of an information space that is useful to achieve a goal) for Big Data analysis to help understand the cognitive complexity of Big Data Analytics as it is believed to consist of components that exploit both machine capability and human intelligence. In this work, the authors also presented two instantiations of the generic architecture of two use cases (social media and biomedical research domains) to provide examples of Big Data solutions related to situations in a specific organisation.

In [9] a requirements specification framework is proposed with the focus of identifying Big Data specific scenarios to be used in the data collection phase in the development of Big Data Analytics applications. In this framework, the scenario description governs the data collection process. Once the Big Data scenarios are elicited, they should be analysed with respect to: (i) the purpose (why, whereto, for when, for which reason); (ii) the sources (data provider, consumer, etc.); (iii) search patterns (determines which phrases and keywords correspond to the scenario at hand and must be contained within the data to be used); and (iv) the value (saving time by not collecting garbage but only needed data that is ready to use for more accurate real-time analysis). The authors claim that it helps to accelerate the analysis time by focusing on retrieving data from the source that meets the scenarios, thus improving the current processes of Big Data collection.

In [11] The NIST Big Data Interoperability Framework provides a discussion on security and privacy requirements with focus on the fundamental concepts needed to understand the

new paradigm for data applications, collectively known as Big Data, and the analytic processes collectively known as data science, and listed requirements extracted and summarised from 51 different use cases. These requirements are classified into seven different groups (e.g., data source requirements, data transformation requirements, etc.). Similarly, in [13] a framework based on Big Data Analytics in mobile cloud computing environments that applies security constraints and access control mechanisms that guarantee integrity, confidentiality and privacy in Big Data healthcare systems is presented.

Tools. In [5], a software verification tool called DICE Verification Tool (D-VerT), - is proposed with the aim to allow designers to evaluate the system design against safety properties such as reachability of undesired configurations of the system. For example, this tool checks if a given topology reaches an unwanted configuration (e.g., whether it allows for bad executions that do not conform to some non-functional requirements). The verification is performed on annotated UML models which contain all the necessary information related to a topology. This tool supports two different types of verification based on logical formalisms: bounded satisfiability checking and the reachability checking. The bounded satisfiability checking has a topology property as input and checks whether there is an execution that violates this property. In the reachability checking type of verification, the topology is defined through an array-based system that undergoes verification of a safety problem. This approach uses a set of system transitions, an initial configuration and, a formula that defines the set of unsafe states. The result of this analysis is either safe or unsafe.

In [10], the authors proposed privacy extensions to UML use cases diagrams to help software engineers to visualize privacy requirements as well as to design privacy into Big Data applications. This solution is implemented as MS Visio extension ribbon in Visual Studio. The authors argued that these extensions to UML help software engineers to visually and quickly model privacy requirements in the analysis phase of the RE process. As a proof of concept, a prototype was created to show the usefulness of the extension and how it can be used to model the privacy requirements for Big Data systems in the domain of healthcare.

3.5 Recommendations for Further Research

In the RE area involving Big Data applications, as stated in [3], a clearer understanding is needed, separating requirements for infrastructures, analytic tools and techniques, and end-user applications. Some papers in RE for Big Data applications describe either the challenges posed by the Big Data paradigm to Software Engineering (Section 3.4.2) or the quality attributes such a Big Data Software might address (e.g., security, performance, data consistency, etc.) (Section 3.4.1). Also, we note that these traditional quality attributes are orthogonal to the V-characteristics of Big Data. Thus, one of the research challenges is to be able to integrate these complementary set of attributes in the specification of system requirements. Moreover - from our analysis we observe that, thus far, little scientific research has focused on RE in the context of Big Data applications and, no research was found addressing RE methods, tools and processes, for negotiation, validation, prioritization and management in the context of Big Data.

Finally, we noticed that little empirical studies have been conducted in this topic (section 3.3). While some papers [8–10, 15] have proposed solutions, they lack validation just yet. Only five papers [2, 5–7, 14] actually have their proposals validated through empirical studies (e.g., case studies in industry). Therefore, it is important that more empirical studies in industry are performed to obtain an improved understanding of the RE activities in the development of Big Data applications. Also, empirical studies would add significantly to the meagre knowledge base on RE involving Big Data applications, which can improve processes and technologies and uncover more facts that could lead to further research in this area.

3.6 Threats to Validity

Concerning the threats to validity, the following threats were assessed.

3.6.1 Construct Validity

Regarding the search string used in this study, we used the terms we considered most suitable to make the string as comprehensive as possible to capture the relevant literature. We performed various searches using the identified terms (e.g., search strings with different combinations of terms) (section 3.2.2) to decide upon the final version. Thus, we anticipate that this threat can be considered contained.

3.6.2 Internal Validity

Two major implications to be discussed are: (i) there might be bias in paper selection and (ii) the fact that we conducted manual searches. These issues were addressed by defining the steps for selecting the potential papers and establishing the selection criteria (sections 3.2.3 and 3.2.4). In addition, with respect to the manual searches, it is important to note that they were performed only in a limited set of sources (e.g., specific journals and conference proceedings).

3.6.3 External Validity

This threat is not considered relevant in this study because unlike in a case study or a scientific experiment where environment scopes (e.g., projects) are bounded, the scope of literature review data (selected papers) is universal.

3.6.4 Conclusion Validity

All the conclusions drawn in this chapter are shown to have been rooted in specific core sections of it thus there is traceability.

3.7 Summary

This chapter describes the results of a systematic literature review on RE research involving Big Data applications. This review was conducted with the aim to answer the overall research

question defined in this study (*What are the early signs of the ways Big Data applications are treated in RE?* See section 3.2.1 where sub-questions Q1-Q3 are also described). The selection process used in this review was composed of three steps (section 3.2.4). At the end of the selection process, 14 papers were deemed relevant for this review (section 3.3).

Our findings are: (i) 11 papers discussed and proposed solutions (section 3.4.3) to address specific areas of the RE process for Big Data applications (e.g., elicitation, specification and analysis of Big Data requirements). These solutions vary from RE methods, models and approaches to frameworks and architectures (see Table 3.6). Moreover, some of the selected papers [3, 8, 12, 14] also discussed RE research challenges in the context of Big Data (section 3.4.2). From our analysis, we also noted the type of requirements and the activities in the RE process that are discussed in the papers selected for this study (section 3.4.1). While the findings may not be surprising to the esoteric few, the value of this chapter to the wider audience is in setting the current baseline.

3.8 Chapter Addendum

Requirements Engineering involving Big Data software applications is an emerging area. The SLR reported in this chapter covers only works published until June 2017. Therefore, it is expected that new contributions appeared from July 2017 to December 2019 (time of thesis submission). Thus, we re-executed the searches (both automatic and manual) in order to uncover new research items that could be relevant to this chapter. Following the same selection process, criteria and steps, we came up with a list of additional papers to be included in this SLR addendum. Figure 3.8 depicts in updated distribution of papers by contribution and type of research organised according to the RE activities they address. This distribution first appeared in Figure 3.3. Table 3.7 summarises the included papers and characterises them with respect to their research type, contributions, and year of publication.

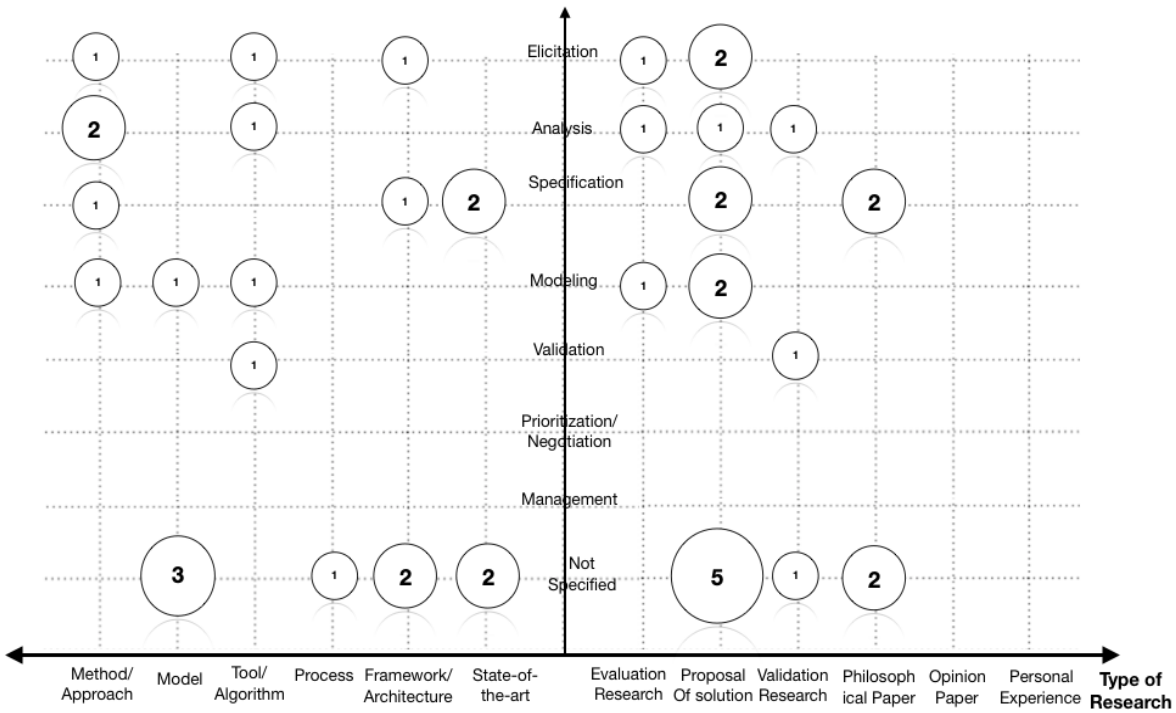


Figure 3.4: Updated distribution of Papers by contribution and type of research organised according to the RE activities they address

Table 3.7: Additions to the SLR results reported in this chapter

Paper Title	Research Type	Contribution Type	Year	Ref.
Ask the Right Questions: Requirements Engineering for the Execution of Big Data Projects	Proposal of Solution	Process Model	2017	[18]
A Requirements Engineering Model for Big Data Software	Proposal of Solution	Model	2017	[19]
User Requirements Based Service Identification for Big Data	Proposal of Solution	Algorithm	2017	[20]
Eliciting Big Data Requirement from Big Data Itself: A Task-Directed Approach	Proposal of Solution	Approach	2017	[21]
Quality profile-based cloud service selection for fulfilling Big Data processing requirements	Proposal of Solution	Approach	2017	[22]
A Collection of Software Engineering Challenges for Big Data System Development	Philosophical Paper	State-of-the-art	2018	[23]
A Validation Study of a Requirements Engineering Artefact Model for Big Data Software Development Projects	Proposal of Solution	Model	2019	[24]
QualiBD: A Tool for Modelling Quality Requirements for Big Data Applications	Proposal of Solution	Tool	2019	[25]

In this SLR study, we also investigated some of the RE challenges in the context of Big Data systems. Table 3.8 depicts a set of new RE challenges identified from the works included in this addendum.

Table 3.8: Additions to the RE Research Challenges discussed in this chapter

RE Challenges	Citation
Selection of services and software components to fulfill requirements for Big Data systems	[21, 22]
Unclear Requirements Specifications	[23]
Integration between hardware and software components	[23]
Emergence of new requirements from Big Data	[23]
Trade-offs between quality and performance given the complexity of Big Data software applications	[23]
Lack of RE specific approaches, tools and techniques for Big Data Systems	[21]

3.8.1 Summary

In this chapter addendum, we introduced new research contributions that were identified after this chapter has been published in 2018. The up to date numbers are as follows:

- 22 papers were reported in this chapter.
- Most of the reported papers were published in 2015 and 2017. The distribution is as follows: (i) 2013: one paper; (ii) 2014: three papers; (iii) 2015: five papers; (iv) 2016: three papers; (v) 2017: 7 papers; (vi) 2018: one paper; and (vii) 2019: two papers.
- 18 out of 22 proposed some sort of RE solution as follows: (i) Approaches, methods, and models: nine papers; (ii) Architecture and Frameworks: two papers; (iii) Tools: four papers; and (iv) Model Process: one paper.
- Eight RE challenges in creating/evolving Big Data applications.

Bibliography

- [1] B. Kitchenham and S. Charters, “Guidelines for performing Systematic Literature reviews in Software Engineering Version 2.3,” *Engineering*, vol. 45, no. EBSE Technical Report, p. 1051, 2007.
- [2] H.-M. Chen, R. Kazman, S. Haziyevev, and O. Hrytsay, “Big Data System Development: An Embedded Case Study with a Global Outsourcing Firm,” *2015 IEEE/ACM 1st International Workshop on Big Data Software Engineering*, pp. 44–50, 2015.
- [3] N. H. Madhavji, A. Miranskyy, and K. Kontogiannis, “Big Picture of Big Data Software Engineering: With Example Research Challenges,” *Proceedings - 1st International Workshop on Big Data Software Engineering, BIGDSE 2015*, pp. 11–14, 2015.
- [4] R. Narayanan, “Evolving and improving the requirements approach to big data projects: a roadmap to implementing big data projects.”
- [5] M. M. Bersani, F. Marconi, M. Rossi, and M. Erascu, “A Tool for Verification of Big-data Applications,” *Proceedings of the 2Nd International Workshop on Quality-Aware DevOps*, pp. 44–45, 2016.
- [6] F. Y.-T. Lydia Lau, N. Karacapilidis, and Abstract, “Requirements for Big Data Analytics Supporting Decision Making: A Sensemaking Perspective,” vol. 5, 2014.
- [7] V. Sachdeva and L. Chung, “Handling non-functional requirements for big data and IOT projects in Scrum,” *Proceedings of the 7th International Conference Confluence 2017 on Cloud Computing, Data Science and Engineering*, pp. 216–221, 2017.
- [8] I. Noorwali, D. Arruda, and N. H. Madhavji, “Understanding quality requirements in the context of big data systems,” *Proceedings of the 2nd International Workshop on BIG Data Software Engineering - BIGDSE '16*, no. 1, pp. 76–79, 2016.
- [9] N. Al-Najran and A. Dahanayake, “A Requirements Specification Framework for Big Data Collection and Capture,” *New Trends in Databases and Information Systems*, vol. 539, pp. 12–19, 2015.
- [10] D. N. Jutla, P. Bodorik, and S. Ali, “Engineering Privacy for Big Data Apps with the Unified Modeling Language,” *2013 IEEE International Congress on Big Data*, pp. 38–45, 2013.
- [11] NIST Big Data Public Working Group: Use Cases and Requirements Subgroup, “NIST Big Data Interoperability Framework: Volume 3, Use Cases and General Requirements,” vol. 3, p. 260, 2015. [Online]. Available: <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-3.pdf>

- [12] C. E. Otero and A. Peter, "Research directions for engineering big data analytics software," *IEEE Intelligent Systems*, vol. 30, no. 1, pp. 13–19, 2015.
- [13] A. E. Youssef, "A Framework for secure healthcare systems based on big data analytics in mobile cloud computing environments," *International Journal of Ambient Systems and Applications*, vol. 2, no. 2, pp. 1–11, 2014.
- [14] H. Eridaputra, B. Hendradjaya, and W. Danar Sunindyo, "Modeling the requirements for big data application using goal oriented approach," *2014 International Conference on Data and Software Engineering (ICODSE)*, pp. 1–6, 2014.
- [15] D. Arruda and N. H. Madhavji, "Towards a requirements engineering artefact model in the context of big data software development projects: Research in progress," in *2017 IEEE International Conference on Big Data (Big Data)*, Dec 2017, pp. 2314–2319.
- [16] R. Wieringa, N. Maiden, N. Mead, and C. Rolland, "Requirements engineering paper classification and evaluation criteria: A proposal and a discussion," *Requirements Engineering*, vol. 11, no. 1, pp. 102–107, 2006.
- [17] I. Sommerville, *Software Engineering*, 2009.
- [18] M. Volk, N. Jamous, and K. Turowski, "Ask the Right Questions : Requirements Engineering for the Execution of Big Data Projects Full Paper," in *Twenty-third Americas Conference on Information Systems*, 2017, pp. 1–10.
- [19] H. H. Altarturi, K. Ng, M. I. H. Ninggal, A. S. A. Nazri, and A. A. A. Ghani, "A requirement engineering model for big data software," in *2017 IEEE Conference on Big Data and Analytics (ICBDA)*, Nov 2017, pp. 111–117.
- [20] H. Wang and H. Zhang, "User requirements based service identification for big data," in *2017 IEEE International Conference on Web Services (ICWS)*, June 2017, pp. 800–807.
- [21] P. Wang, K. Tao, C. Gao, X. Ning, S. Gu, and B. Deng, "Eliciting big data requirement from big data itself: A task-directed approach," in *2017 6th International Workshop on Software Mining (SoftwareMining)*, Nov 2017, pp. 17–23.
- [22] M. A. Serhani, H. T. E. Kassabi, and I. Taleb, "Quality profile-based cloud service selection for fulfilling big data processing requirements," in *2017 IEEE 7th International Symposium on Cloud and Service Computing (SC2)*, Nov 2017, pp. 149–156.
- [23] O. Hummel, H. Eichelberger, A. Giloj, D. Werle, and K. Schmid, "A collection of software engineering challenges for big data system development," in *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, Aug 2018, pp. 362–369.
- [24] D. Arruda, N. H. Madhavji, and I. Noorwali, "A Validation Study of a Requirements Engineering Artefact Model for Big Data Software Development Projects," in *International Conference on Software Technologies (ICSOFT)*, July 2019, pp. 106–116.
- [25] D. Arruda and N. H. Madhavji, "QualiBD Assitant: A Tool for Modelling Quality Requirements for Big Data Systems," in *2019 IEEE International Conference on Big Data (Big Data)*, Dec 2019.

Chapter 4

Requirements Engineering Practices and Challenges in the Context of a Big Data Software Development Project: Insights from a Case Study

4.1 Introduction

On one hand, literature on Big Data mainly focuses on the development of algorithm and machine learning techniques to process large amounts of data [1], and extract value out of it [2]. On the other hand, on a smaller proportion, there is literature [3–11] that focus on understating the processes of engineering applications that operate upon Big Data - the so-called Big Data applications.

Just yet, there is not much “empirically grounded knowledge” on the complexities arising from engineering such applications. The engineering of Big Data is a quite troublesome task. For instance [3, 4, 6, 9], it is known that the massive *Volumes* of data require distributed and parallel processing that traditional database technologies are not designed for; the *Variety* of data demands specific structure modeling and data management; the *Velocity* of Big Data requires data processing speeds that vary from real-time to batch windows which would adapt to different requirements for the system; the *Veracity* poses challenges for data validation and

governance.

The complexities involved in engineering such applications have implications in all activities of the software engineering process. Thus, analogously to the research reported in our previous study [6] (see Chapter 3) - where the focus was to map what have been reported in the scientific literature in terms of types of requirements, activities, challenges, and solution proposals - in this chapter, then, with the focus on the RE field, we aimed at investigating (empirically) the practices and challenges in the context of Big Data software applications development projects.

To this end, we conducted an exploratory case study on a Big Data software development project in the *Oil & Gas* domain. The investigation reported in this chapter was driven by the following core points: (i) the way systems requirements are elicited, specified, analysed and prioritised in such projects; (ii) the sources for identification of Big Data related systems requirements and their proportion in relation to the approximate total number of requirements in the projects; (iii) the role of Big Data characteristics and technologies in the RE and systems design; and (iv) challenges faced throughout the RE process when engineering Big Data applications. Understanding the way systems requirements are elicited, specified, analysed, and prioritised would provide insights on the RE practices in such projects. The sources would provide an idea of where Big Data-related systems requirements are mostly likely to be identified. The role of Big Data characteristics and technologies would give an idea of the extent to which they are explicitly addressed in the solution design. Research challenges would uncover challenges and issues in creating and evolving Big Data software applications.

Questions represented by the aforementioned core points, do not have responses grounded in empirical theory to date [6]. Therefore, the results reported in this chapter have implications in academia as it is new concrete knowledge that adds to the current scarce RE knowledge base in the context of Big Data applications, thus, promoting further research in this area.

The rest of this chapter is organised as follows: Section 4.2 depicts the research methodology, goal, questions, and methods defined for this study. Section 4.3 described the case under analysis. Section 4.4 explores the results of this study. Section 4.5 provides some directions for further research. Section 4.6 discusses the threats to validity and their associated mitigation strategies, and finally, Section 4.7 summarises this chapter.

4.2 Research Design

According to Wohlin et al., [12] there are two types of research paradigms that have different approaches to empirical studies in Software Engineering: (i) exploratory and (ii) explanatory. Exploratory research is concerned with the analysis of a not well-explored phenomena with the aim to seek insights and new ideas that would promulgate new research opportunities, whereas explanatory research focuses on seeking explanations for a situation or a problem [13]. In this chapter, we describe an exploratory case study whose scope is depicted in Figure 4.1.

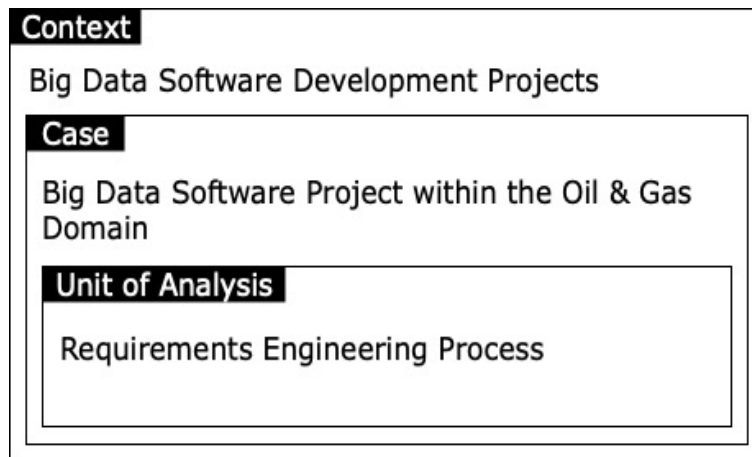


Figure 4.1: Scope of the exploratory case study depicting its context, case, and unit of analysis.

The next sections describe the research goal, research questions, and procedures for data collection and analysis followed in this study.

4.2.1 Research Goal

With the focus on the field of RE, we establish this chapter's research goal using the goal definition template - provided by the GQM (goal-question-metric) approach [14] - as described in Table 4.1.

Table 4.1: Definition of Research Goal

Analyse	RE process
For the Purpose of	Determining
With respect to	Practices and challenges
From the viewpoint of	Project's internal stakeholder
In the context of	Big Data software development projects

4.2.2 Research Questions

Based on the described research goal, we defined the following main research question to guide our investigation: **RQ - What are the RE practices and challenges identified in the Big Data software development project under analysis?** This question aims to determine the practices to elicit, specify, analyse, and prioritise systems' requirements in the project as well as identify the possible challenges encountered while practicing RE in such projects. Given the "broad" nature of this question, we thus decomposed it into the following constituent sub-questions focused on "how" and "what" (as depicted in Table 4.2).

Table 4.2: Decomposed Research Questions

Core Research Questions
Q1 - What are the sources for elicitation of Big Data-related software and non-software requirements? What is the proportion of the Big Data-related requirements in relation to the total number of requirements in the project?
Q2 - How are the systems requirements (specifically the Big Data-related ones) elicited, documented, analysed, and prioritised within the project?
Q3 - What is the role of Big Data technologies and characteristics in Requirements Engineering?
Q4 - What are the challenges in eliciting, documenting, and analysing systems' requirements while engineering Big Data software applications?

Although one may find complementary points to add to this core, in this investigation we believe that the described decomposed research questions cover a significant spectrum of the RE field involving Big Data applications.

4.2.3 Data Collection

The data gathering process was driven directly by the investigative research questions defined in this chapter. For such, we used the following data gathering tools:

(a) **Questionnaire:** we defined a semi-structured instrument composed of 26 questions distributed as follows: (i) Five background questions; (ii) One software development process-related question; and (iii) 20 requirements engineering focused questions. The reason being is that questionnaires are effective tools for gathering relevant and specific data in an organised manner [15], which in turn, aids in faster data analysis;

(b) **Communication with a project stakeholder:** A need for in-depth information or even clarifications would arise from the analysis of the gathered data. Whenever that was the case, follow-up conversations with a project representative were held.

4.2.4 Data Analysis

Coding and Categorisation [16, 17] techniques were used to analyse the data originated from open-ended type of questions. Coding provides a good way of indexing or categorizing pieces of text in order to establish a framework of thematic ideas, thus, facilitating the analysis of qualitative data [17]. During the analysis process, tags (e.g., challenge, practice, context, solution description, quality attribute, etc.) were create to group the answers according to a set of characteristics they represent. For instance, consider a piece of information that describes the challenges in eliciting Big Data requirements. This piece of text would receive two types of tags: challenge and elicitation. The first tag is used to characterise the nature of the information, and the second one is used to described where in the RE process that information occur. The coding and categorisation was done manually by one researcher (first author) and reviewed by another researcher (second author). That was possible because the analysis was performed on questionnaire data, which is more focused and to the point, thus, facilitating the overall analysis process. Important to notice that the data was gathered in both English and Portuguese languages. Figure 4.2 shows an example of a script extracted from an answer given to an open-ended question and their associated tags.

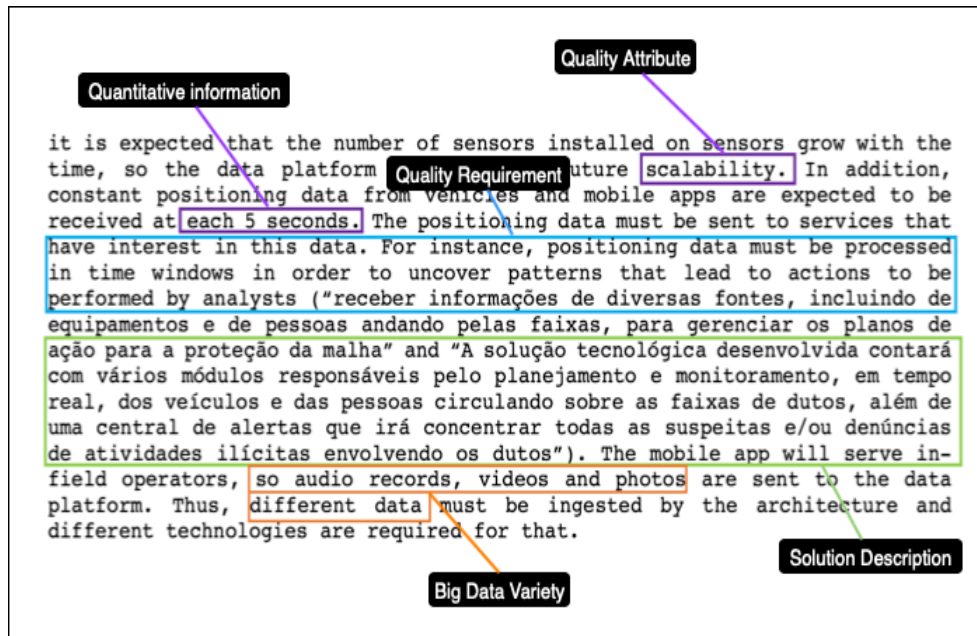


Figure 4.2: Example of tags used in coding and categorising the gathered data.

In the next section, we describe the (i) case explored in this chapter; (ii) context and motivation for the establishment of the project; (iii) Big Data solution, and (iv) organisation of the project and its development methodologies.

4.3 The Case

The case is concerned with a Big Data software development project in the *Oil&Gas* domain within a mid-size non-profit organization, that primarily conducts research and technology development with industrial partners and government institutions.

4.3.1 Context

From project documentation: “Cases of fuel theft in (*removed for privacy reasons*) pipelines have increased exponentially in recent years, with the states of (*removed for privacy reasons*) and (*removed for privacy reasons*) being the most affected ones. Data from Public Prosecution Services show that approximately 14.2 million liters of fuel are stolen annually from (*removed for privacy reasons*) oil pipelines, a subsidiary of (*removed for privacy reasons*) responsible

for fuel transportation and supply logistics. Investigations point to specialized gangs, covering a whole chain of crimes with a high degree of organization and sophistication, ranging from irregular pipeline drilling and tanker truck transportation to illegal refineries to resale. This type of criminal action has already become a national security issue. The biggest concern, however, is the risks to people and the environment in the event of explosions, fires and leaks caused by clandestine shunts. For instance, in December 2018, an attempted theft caused the leak of 60,000 liters of oil in (removed for privacy reasons). In April 2019, an attempted theft of a gas pipeline in the municipality of (removed for privacy reasons) caused a leak, injuring five people and causing the death of a nine-year-old child.”.

The described situation led the company to improve and intensify the pipeline network protection infrastructure it currently operates. For that purpose, it has established a partnership with the aforementioned non-profit organisation with the aim to develop a technological solution (described in the next subsection) composed of a web system - which is integrated with several other existing systems and information sources - and a mobile application to facilitate the activities of in-field agents.

4.3.2 The Big Data Software Solution

As previously mentioned, the project was responsible for the development of a mission-critical large scale Big Data solution. The solution is composed of a (i) web system, and (ii) mobile application. On one hand, the web system features modules responsible for the real-time planning and monitoring of vehicles and people circulating over the pipeline tracks. It also features an alert centre that concentrates all suspicions and allegations of illegal pipeline activities. On the other hand, the mobile app serves in-field agents not only in capturing audio records, videos, photos, positioning data, and alerts that are sent to the data platform but also in retrieving information from the back-end of the system with the aim to provide insights to support the operations of the in-field agents.

Given the possibility for high volumes of data, the system was designed with a microservice architecture and features scalable Big Data technologies (as described in Table 4.3). The following quality attributes drove the definition of system’s architecture: *scalability* (given the

expected volumes of data), *availability* (given the mission-critical nature of the system and the need to run 24x7), *completeness* (given the expected volumes of structured and unstructured data) resilience (given the mission-critical nature of the system), *maintainability* (given the complexity in this type of applications, easy maintainability is important), *performance* (given the nature of real-time data being injecting and retrived from the application), and *modularity* (given the progressive and structured development methodology).

Table 4.3: Technologies adopted according to the Big Data Pipeline

Collection and Transmission	Web Sockets
Storage and Management	MongoDB (for media, KML (geographic files), and ordinary data), Cassandra (for real-time streaming and positioning data from mobile apps), and H2 (for positioning data extracted from geographic files)
Processing	Flink (which will be replaced by a custom tailored solution currently under development)
Visualisation	Google Maps (for building KMLs), web-sockets, and leaflet (for visualising data on maps)

4.3.3 The Big Data Software Development Project

In this section, we describe some of the characteristics of the software development project such as team formation and their respective roles in the project, development methodology and process, and RE process.

Distribution of Team Members and Roles

The Big Data software project is composed of 10 members organised according to the following roles: three developers focused on the the development of the web application; three developers focused on the development of the mobile application; one of the developers also played the role of data engineer and software architect; two business analysts (a.k.a requirements analysts); and two project managers.

Software Development Methodology and RE Process

The project uses a mix software development methodology. Sprints (from agile) [18] are used for the planning of the development activities. The software development itself follows a logical sequence characterised by a progressive and structured development process, which in turn, enables the implementation and integration of features in an incrementally and evolutionarily manner. With regards to the RE process, it follows a very standard structure of activities as follows: Elicitation, Specification and Modelling, Analysis, and Prioritisation.

4.4 Results

This section discusses the results obtained for each research questions defined in Section 4.2.2.

4.4.1 (Q1) - What are the sources for elicitation of Big Data-related software and non-software requirements? What is the proportion of the Big Data-related requirements in relation to the total number of requirements in the project?

With this research question, we aimed at determining the source from which Big Data-related requirements are primarily identified.

Big Data-related requirements refer to the software and systems requirement intrinsically related to Big Data. In other words, those requirements that are directly related to Big Data itself or those requirements that trigger Big Data and analytics. For instance, a non-big data related requirements would look like “*the system shall support the registration of new users with either email address and social media credentials*”. This requirements does not relate with Big Data nor triggers any Big Data analytics in the back-end of the system. On the other hand, some examples of Big Data related requirements are [19, 20]: (i) Big Data processing requirements (e.g., *the system shall support real-time analytics*), (ii) Big Data consumer requirements (e.g., *the system shall display all information of user’s interest as layers in a georeferenced interface.*), (iii) Big Data Infrastructure Requirements (e.g., *the system shall support large distributed data storage*); and (iv) Big Data source requirements (e.g., *the data format of a given*

API must comply to a previously agreed specified interface).

Approximately 40% of the total number of requirements in the project is considered Big Data-related requirements. These requirements are identified from the internal and external sources to the project. Roughly 25% of the Big Data requirements are identified from external sources (e.g. end-users) whereas the remaining 75% are identified in-house (e.g., customer stakeholders, project stakeholders, contractual documentations, etc.). Finally, When it comes to the elicitation of Big Data requirements, roughly 65% are identified upfront while establishing the project whereas around 35% are identified later in downstream processes (e.g., design/coding/testing). We provide a more detailed analysis and discussion concerning this distribution in Section 4.4.5.

4.4.2 (Q2) - How are the systems requirements (specifically the Big Data-related ones) elicited, documented, analysed, and prioritised within the project?

In this section, we describe the results concerned with the “how” type of question. Following the RE activities performed in the project, we identified ten RE practices and their associated supporting tool (when needed) as described in Table 4.4.

Table 4.4: RE practices and Supporting Tools/Techniques

RE Activities	Practices	Tool/Technique Support
Elicitation	<ol style="list-style-type: none"> 1. Analysis of contractual documents. 2. Meetings with customer stakeholders. 3. Requirements conceptual models to support the elicitation process. 	—
Specification & Modelling	<ol style="list-style-type: none"> 4. Architecture, data flows, and conceptual data structure are specified through modelling. 5. User interface and data visualisation requirements are specified as mock-up screens. 6. Requirements (functional and non-functional) are specified through use cases 7. Big Data Characteristics are implicit assumed in the project's use cases and explicitly noted through sticky notes on screen mock-ups, data flows, and other models. 	<p><i>Google Draw</i> for documenting the architecture.</p> <p><i>Visual Paradigm</i> for data flow and data models.</p> <p><i>Balsamiq</i> for screen mock-ups.</p>
Analysis	<ol style="list-style-type: none"> 8. Requirements are analysed in order to determine the extent to which they address customers' expectations and needs. 9. Requirements analysis are also shown to project managers. 10. Updates on requirements, screen mock-ups, and models are made based on analysis results. 	Manual and Visual Inspection.
Prioritisation	<ol style="list-style-type: none"> 11. Prioritisation is driven by functionality and its importance to the end-user and customers. 	—

With reference to Table 4.4:

Elicitation. Requirements elicitation, also known as requirements discovery, is one of the crucial tasks of the requirements engineering process, enabling the discover of which requirements the users want to see incorporated into the system under development [21]. The requirements elicitation in the project under analysis was primarily driven by holding frequent meetings with customer stakeholders and representatives as well as performing the analysis of contractual documents. To support the requirements identification task, the project's business analysts defined a requirements model depicting the various system's components and their interactions with other systems, data sources, and users.

Analysis. The requirements analysis activity relates to the refinement of stakeholders needs into formal product specifications [22]. The analysis of systems requirements in the project

follows a very standard approach, where the requirements are analysed by the business analyst with the aim to determine whether the documented requirements reflect user's expectations. Then, the analysis is also presented to the project managers in order to keep them up to date with the latest version and correct state of the project. For such, manual and visual inspections are performed. If any changes are required to the front-end developers regarding the systems' user-interface, the mock-up screens and their associated requirements are also updated.

Specification and Modelling. Once the requirements are elicited, they are expressed in form of use cases for textual description. Additionally, some of the requirements in the project are documented in a graphical manner. For example, while modelling data flows, data model, and creating conceptual models, and architecture of the application under development, the Visual Paradigm - a UML CASE Tool supporting UML, SysML, and Business Process Modeling Notation - is used. As for the user-interface and presentation layer requirements, mock-up tools are used to create a prototype or a model that provides an idea about how the final product, once done, will be.

Prioritisation. Requirements prioritization is the activity that aids in identifying the subset of important requirements (e.g., the requirements that are susceptible to be incorporated in the system, due to their relevance from the onset), according to various criteria defined by the project [21, 23]. In the project under analysis, the systems requirements are mostly prioritised based on the importance (to the customer and end-users) of a given functionality present in the system rather than the existence of specific data characteristics or quality attributes within the system. Of course that one could argue that the data characteristics and quality attributes (expressed as quality requirements) defined for the system would be prioritised in the design of the solution as the system's architecture is highly dependable on those attributes. We further discuss the role of data characteristics and technologies in the RE process in the project in Section 4.4.3.

4.4.3 (Q3) - What is the role of Big Data technologies and characteristics in Requirements Engineering?

Several reported studies acknowledge the importance of addressing the Big Data characteristics (also known as “V” attributes) in the solution requirements [3, 5, 6, 9, 24] and system’s design [3, 4, 25]. Likewise, when it comes to the definition of the system’s architecture, the “V” characteristics are of extremely importance assisting in selecting the appropriated set of technologies to compose the architecture [4].

With this in mind, in this study (because RE is a cross-functional discipline and provides support to all phases in the software development process), we investigated the extent to which the “V” attributes and Big Data technologies are explicitly addressed in the project requirements and design documents. To this end, we asked questions such as *“Are Big Data technologies considered or identified at the stage of identifying system’s requirements or are these decided upon later in the process?”* and *“Are the Big Data characteristics expressed in requirements notations along side with the traditional quality attributes?”*

In this section, we first comment on whether the V characteristics of Big Data are expressed explicitly, in the requirements documentation. We then discuss the role of data characteristics and technologies in the definition of systems architecture and requirements satisfaction.

As for the explicit specification of Big Data characteristics in the project’s requirements descriptions within the project documentation, our analysis demonstrated that those characteristics are no explicit described in textual forms on the use cases defined for the project. Instead, they are implicitly understood. For example, in the project’s use cases there would be mentions such as (i) “large amounts of real-time data”, and (ii) “Audio records, videos, and photos will be sent to the data platform”. If we mapped the first occurrence to their corresponding “V” characteristics, then “large amounts” would refer to volume whereas “real-time streaming data” would refer to the velocity of Big Data. Following the same premise, “audio records, videos, and photos” would refer to the variety of Big Data.

Additionally, at some point, where graphical notations are used, notes on Big Data Characteristics are provided (e.g., a mock-up screen depicting the user-interface of the system that would display the results of the data being analysed in real-time, would have a “stick note” attached to it mentioned the V attribute. In this case, for example, “V” characteristic would be

Visualisation of Big Data).

As for the impact of V characteristics of Big Data in solution design and technological requirements satisfaction, in the project under analysis, some of the Big Data technologies were identified at the stage of identifying systems requirements during the elicitation process. That was possible given the defined goals for the project. For instance, during the definition of the project, it was already expected that the system would handle and store large amounts (*Volume*) of real-time positioning data (*Velocity*). Given this, Apache Cassandra was selected as database management system given its ability to support the fast ingestion of data and horizontal scalability. Following the same idea, Flink was chosen as the technology to handle the processing of real-time data. Moreover, thinking about the performance of the system, H2 was chosen to store positioning data extracted from geographic files, the reason being is that H2 provides the real-time processing of events and features cache domain info that supports real-time decision-making in real-time processing environments. That helps avoiding requesting data from APIs, which could be slow and hard to aggregate with real-time streams. Moreover, it is important to notice that the main “criteria” used in selecting the Big Data technologies - besides the intrinsic quality requirements of the application - was the better trade-off between cost and benefit in terms of ease maintenance and high levels of performance.

4.4.4 (Q4) - What are the challenges in eliciting, documenting, and analysing systems’ requirements while engineering Big Data software applications?

Analysis of gathered data led us to the identification of a set of **RE challenges** in the context of the studied Big Data application development project. The identified challenges are organised based on the activities in the RE process they relate to, and presented in Table 4.5 and described in the following paragraphs.

Choosing the best fit technology to meet users’ needs and structural quality of the application: During the system design phase, when it comes to selecting the appropriate technologies to address the Big Data envisaged requirements, it is important to consider how to select technologies and software resources to be used in the project as well as the extent to

Table 4.5: RE challenges in creating Big Data Applications

Requirements Engineering Challenges
Elicitation
1. Selection of appropriate Big Data Technologies to meet users' needs and requirements, and structural quality of the application.
Specification & Modelling
2. Lack of specific modelling tools for Big Data requirements. 3. Lack of specification techniques for Big Data specifying system's requirements.
Analysis
4. Lack of appropriate knowledge concerning Big Data architectures and technologies. 5. Lack of industrial patterns and specifications for Big Data architecture and requirements

which the existing technologies and frameworks help in addressing the Big Data requirements, both system and software. The large number of distributed systems frameworks and technologies available in the context of Big Data, introduces challenges and time constraints to the project that imply in short window for experimentation. The same holds true for the activity of selecting software resources (e.g., external services) [26, 27]. Those software resources or services often refer to a large variety of outsourceable functions to the cloud usually accessible through APIs. It can range from supply of data to the supply of analytical software tools [26], for instance.

Lack of specific modelling tools and specification techniques. As we described in early sections of the chapter, Big Data applications are complex solutions composed of several components. The current existing tools for modelling and specifying Big Data system requirements do not support some of the common concepts underlying Big Data (e.g., data characteristics, processing techniques, visualisation, services, data flow, etc.). This results in incomplete requirements specifications that introduce misconceptions over the Big Data-related requirements to be met by the system, thus, affecting negatively the project as whole.

Lack of industrial patterns or specifications for architecture and requirements. The lack of solution patterns for requirements, data, and architecture introduces important challenges. For instance, each modelled element must be explained to the audience prior to presenting the architecture. Otherwise, the architecture design won't be clearly understood by all project stakeholders.

Lack of in-depth knowledge about Big Data technologies and reference architectures.

The lack of adequate knowledge concerning Big Data technologies and references architecture pose significant challenges to engineering Big Data applications. For instance, in the context of the studied project, for enabling an intensive push of positioning data to the browser, at first, the project stakeholders did not know they had to rely on using *Web sockets* technology. This was communicated to them from a senior developer from other project within the organisation. The lack of knowledge limits the productivity of the team, contributes to the unclear and inconsistent requirements specifications, thus, resulting in re-work.

4.4.5 Discussion**RE Practices**

While the RE practices and support tools reported in Section 4.4.2 of this chapter might appear expected, the value of this study to the wider audience is in providing empirical evidence of “how” Big Data and its related requirements are handled in development of Big Data software applications. Moreover, one may argue whether specific RE approaches and tools are really necessary for Big Data software projects (as argued by [3–6]) since the results reported in this chapter depicts activities being performed using existing general purpose tools. That is due to the fact that the field of RE in the context of Big Data applications is still not well developed. Thus, practitioners must use the available tools (even if they are not ideal) to support their activities. For instance, we identified that due to the lack of modelling tools and industrial patterns for Big Data applications, the project members had to use Google Drawing (which is not a software engineering supporting tool) to graphically “model” the system’s architecture. It was even mentioned to us that project’s architect had to “*use boxes and lines to indicate services and data flow, respectively*”. This provides opportunites for work in this area of RE as described in Section 4.5.

Distribution of Requirements

Recall that in RQ1 reported in Section 4.4.1, we identified that around 35% of the system's requirements were identified in downstream processes (such as design, coding, implementation, testing, etc.)? The statement raises important questions regarding (i) the extent the architecture of the system is stable e resilient; (ii) the extent of re-work performed in later phases of the project; (iii) money loss; (iv) schedule overruns, to name a few.

When questioned about this intriguing situation, the project representative provided us with some context: This phenomena happened mainly due to the lack of deep knowledge about Big Data and Big Data technologies, and the fact that there are way too many available Big Data frameworks and technologies, which in turn, makes it difficult to map systems requirements with the technologies that would potentially address those requirements.

For instance, recall that in Section 4.3.2, one of the quality requirements that drove the definition of systems architecture is related to maintainability? In this context, conscious decisions have been made by adopting Flink as the real-time processing engine. That is because Flink has complex maintenance procedures (maintenance risks related to changing source code and complex deployment procedures). At this moment, the project is investigating and implementing a custom tailored solution to substitute Flink. Likewise, the design of some microservices have been shown to be troublesome. For instance, some of them could have been merged into on unique service, thus, decreasing complexity in system's design, implementation, and operation. Also, some problems with the deployment of solution occurred. For example, Docker incurred in a effort that were not accounted for. While Dock is not a complex technology, the learning curve to effectively master it requires a considerable amount time.

Finally, the unclear specifications of requirements or the lack of requirements identified upfront in the process, also led in wrong definition of data models, resulting in re-work. The project representative stated that data models will be reviewed when establishing a new contract for the implementation of future versions of the Big Data solution.

RE Challenges

In a previous study [6], reported in Chapter 3, we described a few RE challenges in creating and evolving Big Data applications. Because RE involving Big Data applications is an emerging area, we stated that, intuitively, there would be more challenges and issues related to the RE activities in the development of such applications than it has appeared from that study. The case study reported in this chapter, then, confirms our statement. We identified five more challenges faced while engineering applications that operate upon Big Data (see Section 4.4.4). Collectively, the challenges open up new venues for research exploration in the field of RE involving Big Data applications.

4.5 Directions for Further Research

The results reported in this chapter can lead to several opportunities for further research. These opportunities are mainly on the scope of the RE challenges described in Section 4.4.4, and can be focused either on specific RE processes and tools (e.g., conflict management, modelling tools, specification techniques and tools, etc.) or specific approaches and methods for requirements specifications, modelling, management, and requirements conflict, for instance.

Example research questions are: (i) How to systematically map and select the available Big Data technologies to fulfil system's requirements? (ii) How to specify Big Data quality requirements that incorporate both Big Data characteristics and quality attributes? (*this specific question is explored in chapter 6 of this thesis*); (iii) How to represent semantic properties and patterns of Big Data in requirements notations?; (iv) How to manage the complexities arising from engineering Big Data applications?

Finally, empirical studies are a great way to generate knowledge and contribute to the scientific knowledge base of the field under analysis. Therefore, additional empirical studies in industry (like the one reported in this chapter) must be conducted in order to obtain an improved understanding of the RE practices concerning Big Data software development projects, which in turn, can aid in process and technology improvement, and uncover more facts that could promote further research.

4.6 Threats to Validity

This section overviews the threats to validity identified in this study. We followed the guidelines proposed by Runeson and Host [13] which describe four main types of threats to validity for case studies within the software engineering field: reliability, construct, internal, and external validity. To this set of threats, we add conclusion validity [12]. Considering that in this research we did not aim at establishing any types of causal relationships, we did not consider the interval validity as a threat in this study [13] [12].

4.6.1 Reliability Validity

Reliability validity is concerned with the extent to which the data and the analysis are dependent on the specific researchers [13]. In other words, it is concerned with the replicability of the study.

Regarding the reliability of data gathering, we based our process on a semi-structured questionnaire. One could argue that questionnaires questions could introduce validity threats if the designed questions are not clear or are not aligned with the research's main objective. Thus, the instrument for data collection used in this study was created by the first author and validated thoroughly by the second author through several iterations. This helped to minimise any possible threats to reliability of the instrument used in this study.

As for the analysis of the data gathered, one researcher (the first author) was at the core throughout the data analysis process. One researcher (the third author), reviewed the results of the analysis drawn by the first author, thereby mitigating researcher bias.

4.6.2 Construct Validity

Construct Validity is concerned with the extent to which the operational measures represent what is being investigated in the study [13]. Two threats to construct validity are identified:

The quantitative information provided as part of the answer to Research Question 2 described in Section 4.4.2, were approximate numbers based on the experience of the project representative. They do not represent the exact numbers of requirements in the project. Thus,

this remains a threat in this study. However, it is worth mentioning that the project representative played the roles of developer, systems architect, and data engineering, thus, holding a significant amount of knowledge concern the project and its characteristics.

The fact that we used questionnaires as the instrument for data collection could introduce threats to construct validity if the defined questions were not interpreted in the same way by the researcher and the respondent [13]. In order to minimise this threat, as explained in Section 4.2.3, we had follow-up conversations with a project representative whenever unclear statements were identified. Moreover, the use of coding and categorising technique [16] (see Section 4.2.4) helped prevent the personal biases of the researchers and overcome the deficiencies intrinsic to the study, thereby mitigating researcher bias as well.

4.6.3 External Validity

External Validity is concerned with the generalisability of the results to other contexts [13]. As described by Wohlin et al., [12], normally, generalisability from case studies is weaker due to the lack of control, low replicability, and the fact that there is no population from which a statistically representative sample could be drawn [12]. In the context of the research reported in this chapter, the reported RE practices and challenges might differ from organisation to organisation or project to project. This implies that the disclosed results (see Section 4.4) are not immediately generalisable to other contexts. However, despite this obvious limitation, the results account for an important data-point for improving the scientific knowledge base in field of RE involving Big Data applications.

4.6.4 Conclusion Validity

The conclusion validity is concerned with the relationship between the treatment and the results. It concerns whether conclusions are traceable to the findings [12]. All the conclusions drawn in this chapter are shown to have been rooted in specific core sections of this chapter thus there is traceability. However, should any assumptions underlying the case study not be valid then the validity of the results and, consequently, of the conclusion drawn, would be

threatened.

4.7 Summary

In this chapter, we reported on the results of an exploratory case study conducted on a Big Data applications project within the Oil&Gas domain with the goal to determine the RE practices and challenges in such projects. This research provided preliminary but important results in the context of Big Data software development projects.

Ten RE practices were identified and organised according to the RE process (see Section 4.4.2). We also analysed the role Big Data characteristics and technologies play in defining systems requirements and architecture (see Section 4.4.3). Moreover, five challenges in creating Big Data applications were identified (see Section 4.4.4). Even though the goal of this study was not focused in defining causal-relationships, we could not fail to notice that these challenges, specifically, led to we-work and unstable definition of system's architecture in the project. Finally, these challenges present the research community with opportunities for further research in this area of RE concerning the development of Big Data applications.

Bibliography

- [1] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, “A survey of machine learning for big data processing,” *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 1, p. 67, 2016.
- [2] R. Wegener and V. Sinha, “The value of big data: How analytics differentiates winners,” 2013.
- [3] N. H. Madhavji, A. Miranskyy, and K. Kontogiannis, “Big Picture of Big Data Software Engineering: With Example Research Challenges,” *Proceedings - 1st International Workshop on Big Data Software Engineering, BIGDSE 2015*, pp. 11–14, 2015.
- [4] H.-M. Chen, R. Kazman, S. Haziyevev, and O. Hrytsay, “Big Data System Development: An Embedded Case Study with a Global Outsourcing Firm,” *2015 IEEE/ACM 1st International Workshop on Big Data Software Engineering*, pp. 44–50, 2015.
- [5] V. Dipti Kumar and P. Alencar, “Software engineering for big data projects: Domains, methodologies and gaps,” *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, pp. 2886–2895, 2016.
- [6] D. Arruda and N. H. Madhavji, “State of requirements engineering research in the context of big data applications,” in *Requirements Engineering: Foundation for Software Quality*, E. Kamsties, J. Horkoff, and F. Dalpiaz, Eds. Cham: Springer International Publishing, 2018, pp. 307–323.
- [7] R. N. Laigner, M. Kalinowski, S. Lifschitz, R. S. Monteiro, and D. D. Oliveira, “A Systematic Mapping of Software Engineering Approaches to Develop Big Data Systems,” no. July, 2018.
- [8] H. Eridaputra, B. Hendradjaya, and W. Danar Sunindyo, “Modeling the requirements for big data application using goal oriented approach,” *2014 International Conference on Data and Software Engineering (ICODSE)*, pp. 1–6, 2014.
- [9] I. Noorwali, D. Arruda, and N. H. Madhavji, “Understanding quality requirements in the context of big data systems,” *Proceedings of the 2nd International Workshop on BIG Data Software Engineering - BIGDSE '16*, pp. 76–79, 2016.
- [10] F. Y.-T. Lydia Lau, N. Karacapilidis, and Abstract, “Requirements for Big Data Analytics Supporting Decision Making: A Sensemaking Perspective,” vol. 5, 2014.
- [11] D. N. Jutla, P. Bodorik, and S. Ali, “Engineering Privacy for Big Data Apps with the Unified Modeling Language,” *2013 IEEE International Congress on Big Data*, pp. 38–45, 2013.

- [12] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson, B. Regnell, and A. Wessln, *Experimentation in Software Engineering*. Springer Publishing Company, Incorporated, 2012.
- [13] P. Runeson and M. Hst, “Guidelines for conducting and reporting case study research in software engineering,” *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2009.
- [14] V. R. B. G. Caldiera and H. D. Rombach, “The goal question metric approach,” *Encyclopedia of software engineering*, pp. 528–532, 1994.
- [15] M. N. Douglas R. Berdie, John F. Anderson, *Questionnaires: Design and Use*, 1986.
- [16] J. W. Creswell, *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage, 2013.
- [17] G. Gibbs, *Analyzing qualitative data, Qualitative research kit*, 2007.
- [18] (2019) Agile requirements core practices. [Online]. Available: <http://www.agilemodeling.com/essays/agileRequirementsBestPractices.html>
- [19] D. Arruda, N. H. Madhavji, and I. Noorwali, “A Validation Study of a Requirements Engineering Artefact Model for Big Data Software Development Projects,” in *International Conference on Software Technologies (ICSOFT)*, July 2019, pp. 106–116.
- [20] “NIST Big Data Interoperability Framework: Volume 3, Use Cases and General Requirements,” vol. 3, p. 260, 2015. [Online]. Available: <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-3.pdf>
- [21] J. M. Fernandes and R. J. Machado, *Requirements in Engineering Projects*, 1st ed. Switzerland: Springer International Publishing, 2016.
- [22] B. Berenbach, D. Paulish, J. Kazmeier, and A. Rudorfer, *Soft. Systems Requirements Eng. In Practice*, 2009.
- [23] I. Sommerville, *Software Engineering*, 2009.
- [24] 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*, May 2015, pp. 19–25.
- [25] C. E. Otero and A. Peter, “Research directions for engineering big data analytics software,” *IEEE Intelligent Systems*, vol. 30, no. 1, pp. 13–19, 2015.
- [26] H. Wang and H. Zhang, “User requirements based service identification for big data,” in *2017 IEEE International Conference on Web Services (ICWS)*, June 2017, pp. 800–807.
- [27] M. A. Serhani, H. T. E. Kassabi, and I. Taleb, “Quality profile-based cloud service selection for fulfilling big data processing requirements,” in *2017 IEEE 7th International Symposium on Cloud and Service Computing (SC2)*, Nov 2017, pp. 149–156.

Chapter 5

An Empirically Derived Requirements Engineering Artefact Model in the Context of Big Data Software Development Projects

5.1 Introduction

Predominantly, the current focus in the field of Big Data software is on data analytics and the development of algorithms and techniques to process and extract value from huge amounts of data [1]. In contrast, little research or industry practices focus on software applications and services that utilise the underlying Big Data to enhance the functionality and services provided to the end-users [2,3].

While scientific literature [1,4,5] and economic outlook [6,7] suggest that the field of Big Data is growing exponentially, there is no recognisable body of knowledge on the development of applications and services that utilise Big Data. Consequently, end-users are potentially missing out on the anticipated benefits of innovative applications and services that could provide enhanced results, experience, or value. This void is also reflected in the field of Requirements Engineering (RE) where current RE practices (such as elicitation, specification, analysis, etc.) [8] do not prescribe how to treat Big Data and the V characteristics in the development of

Big Data software applications. The current difficulties in the RE process for Big Data applications is compounded by the lack of suitable domain or artefact models, considered important in RE [9]. A solid foundation for creating sound applications is a thorough understanding of the domain and models that embody the various artefacts, activities, and relationships involved in the RE process [10, 11].

In order to ameliorate the current situation, we attempt to throw some light on different types of artefacts and inter-relationships involved in Big Data software development projects, with particular focus on Requirements Engineering. This is a foundational phase for every large software project. It deals with what the customer wants, and how the system should behave during usage [12], [8]. The detailed artefacts and inter-relationships are embodied in a model, called a Requirements Engineering Artefact Model (REAM) in the context of Big Data software projects (BD-REAM) [13]. This model was subsequently assessed for qualities such as accuracy, completeness, usefulness, and generalisability by ten practitioners from Big Data software projects in industry. This validation study is also described in this chapter including the resultant improved artefact model. Thus, the contribution of this chapter, firstly, is the improved artefact model. Further, this chapter creates a stronger baseline for the RE artefact model for Big Data software systems upon which can depend new applications development, RE technology development, and further empirical studies.

The remainder of the chapter is organised as follows: Section 5.2 describes the model creation process. Section 5.3 d presents the pre-validation version of the proposed artefact model. Section 5.4 presents the model validation procedure and the methodology used in the validation study. Section 5.5 describes the assessment results. Section 5.6 compares the old and the new versions of the artefact model as well as introduces the post-validation version of the RE artefact model in the context of Big Data Software Developments Projects. Section 5.7 describes threats to validity and the respective mitigation strategies. Section 5.8 summarises this chapter. Finally, Section 5.9 describes an addendum for this chapter.

5.2 Model Creation Process

This section describes the model creation process we followed in order to define the first version of the BD-REAM. For that, we used the 4-step process proposed by Berenbach [9]. To this process, we added one more step called “*Artefact Model Evaluation*”, and organised all the activities into three distinctive phases with its own inputs and outputs - as depicted in Figure 5.1.

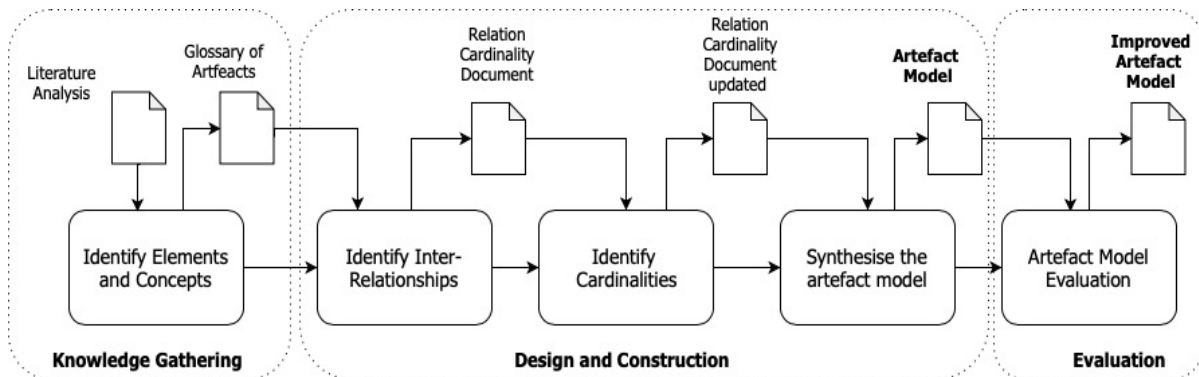


Figure 5.1: Model Creation Process

With reference to Figure 5.1, in the subsequent subsections, we briefly discuss each activity depicted in the model creation process.

Identification of Elements and Concepts

A Systematic Literature Review (SLR) (reported in chapter 3 of this thesis) was conducted on Requirements Engineering involving Big Data Applications [3]. In total, 311 papers were identified and, after methodical selection, 14 papers were deemed relevant to be used in our review. In addition to the results of the SLR, we also selected traditional software and RE literature [8, 9, 12]. The selected papers and traditional software and requirements engineering literature were then analysed and the model elements (artefacts) identified, from which a glossary of terms was created. However, it is important to notice that the main of the SLR study was not to identify RE artefacts but to understand and map the state of the art of RE involving Big Data software systems. The definition of each element was extracted either from the results of our SLR or traditional Requirements and Software Engineering literature.

Definition of the Artefact Relationships

Using the glossary of artefacts and interpreting the domain knowledge from the scientific literature, we created a table of artefacts and their inter-relationships referred as to the *Relation Cardinality Document*.

Definition of Cardinalities

Using the basic table of artefacts and inter-relationships (*Relation Cardinality Document*) and interpreting the domain knowledge from the scientific literature, a *Relation Cardinality Document* was updated.

Synthesising the Artefact Model

Using fragments of artefacts, their inter-relationships, and cardinality information, they were inter-connected iteratively, respecting RE domain knowledge, eventually resulting in the artefact model as shown in Figures 5.3.

Evaluation of the Artefact Model

In this step of our research, we performed the evaluation of the proposed artefact model. This step is composed of its own process and methodology presented and discussed in Section ?? of this chapter.

5.3 Pre-validation version of the RE Artefact Model in the context of Big Data Software Development Projects

The completion of the activities in the **Design and Construction** phase of the model creation process (depicted in Figure 5.1) resulted in the first version of the Big Data Requirements Engineering Artefact Model (BD-REAM) [13], referred as to, in this chapter, the pre-validation version of the BD-REAM.

The pre-validation version of the BD-REAM is composed of three basic elements:

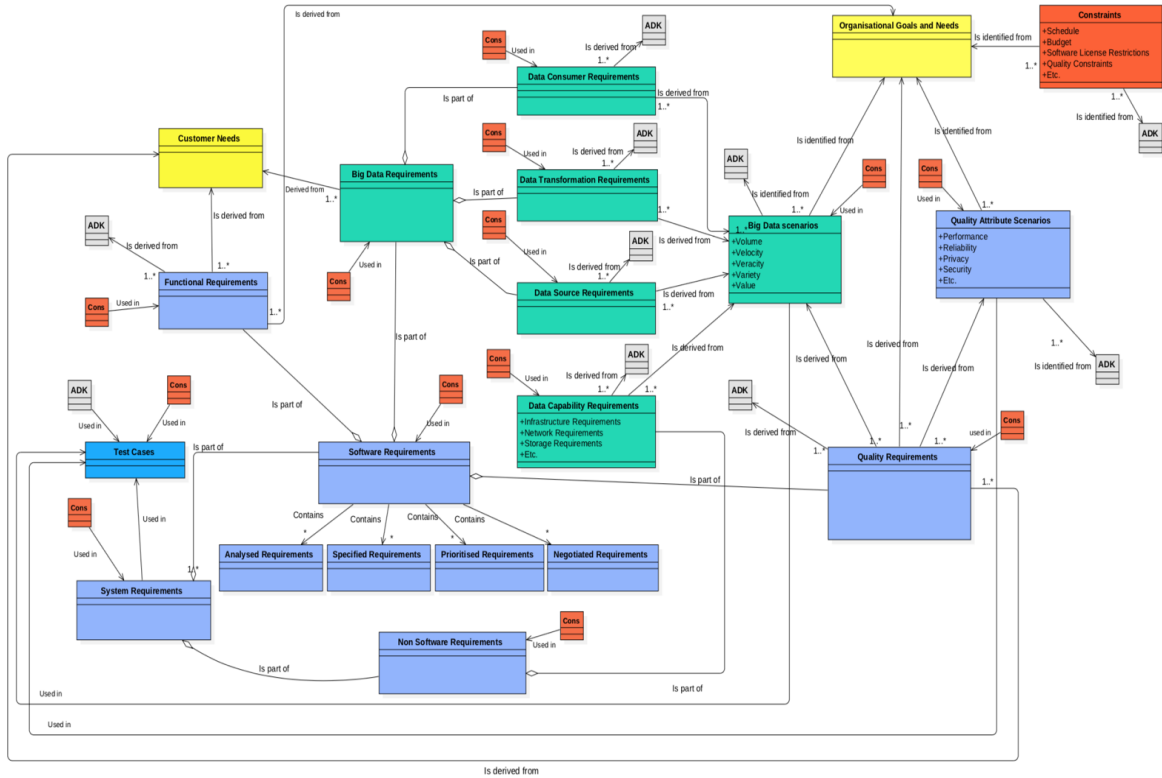
- Artefact: a rectangular shape (UML Class) identified with the name of the artefact it represents;
- Association: a line connecting two artefacts. Each association is labelled to indicate the type of relationship between the artefacts;
- Cardinality: it indicates quantity. If the cardinality is not expressed in the association line, it means that it has a value of 1.

The following relationships are represented in the model: *Is-derived-from*, *Is-identified-from*, *is-part-of*, *Contains*, and *Used-in*.

This version of the model is composed of 21 elements of which six are Big Data specific elements and numerous relationships. Example elements are [14]: Data-Capability Requirements (typically infrastructure related): *the system shall support legacy, large distributed data storage*; Data-Source Requirements (e.g., *the system shall support high-throughput data transmission between data sources and computing clusters*); Data-Transformation Requirements (typically processing related): *the system shall support batch and real-time analytics*; Data-Consumer Requirements (e.g., *the system shall support diverse output file formats for visualisation*).

Figure 5.3 depicts the pre-validation version of the requirements engineering artefact model in the context of Big Data system development projects.

Please note: To avoid repetition, in this subsection, we will not describe in details the entities in the model nor their respective inter-relationships and cardinalities. This information will be provided later in this chapter in Section 5.6 when we discuss the post-validation (improved and up to date) version of the BD-REAM.



Legend: ADK - Application Domain-knowledge | Cons - Constraints | \longrightarrow Association line | \longleftarrow Aggregation Line | \equiv Artefact

Figure 5.2: Pre-validation Big Data RE artefact model [13]

This model depicts 21 elements. The entities coloured in “green” are Big Data elements. The entities coloured in “purple” represent the traditional RE artefacts. The entities coloured in “yellow” represent the business needs artefacts. The entities coloured in “orange” and “grey” represent the project constraints and application domain knowledge, respectively. Finally, the entity coloured in “blue” represents the test cases artefact that is defined based on systems requirements. **If the cardinality is not expressed in the association line, it means that it is has a value of 1..**

5.4 Model Evaluation Study

In [15], Shaw describes several types of validation in software engineering research: (a) by analysis; (b) by experience; (c) by example; (d) by evaluation; (e) by persuasion; and (f) by blatant assertion. Shaw also explains that the validation type needs to be appropriate for the type of research contribution (e.g., validation by experience would be suitable for research results that have been used in practice by someone other than researcher).

For the descriptive model that we describe in this chapter, the appropriate validation procedure is evaluation: to assess whether the proposed model satisfactorily describes the phenom-

ena of interest, in our case, development of Big Data software applications.

For the purpose of validation, we created an instrument (questionnaire) for gathering data, composed of 15 questions organised as follows: (i) background questions; (ii) technical validation questions concerned with completeness and accuracy of the elements and relationships depicted in the proposed model; and (iii) validation questions concerned with usefulness and generalisability of the proposed model. On the one hand, the technical validation questions refer to the types of questions focused on the technical elements of the model. For example, “Is the naming structure technically sound?”, “Do the model express the common spectrum of Big Data requirements?”, “Is there any Big Data element missing in the model?”. On the other hand, the general validation questions refers to the participants’ opinions regarding the applicability, usefulness, and generalisability of the model in an industry setting.

The questions in the instrument had multiple-choice responses; used the 5-point Likert scale [16] (strongly agree to strongly disagree); and a few open-ended questions concerning the artefact model. Thirteen practitioners in Big Data software development projects were invited of which ten 10 agreed to participate in the study. Three declined due to business constraints.

5.4.1 Model Evaluation Process

This section depicts the qualitative research methodology [17] composed of a 4-phase research process as depicted in Figure 5.3.

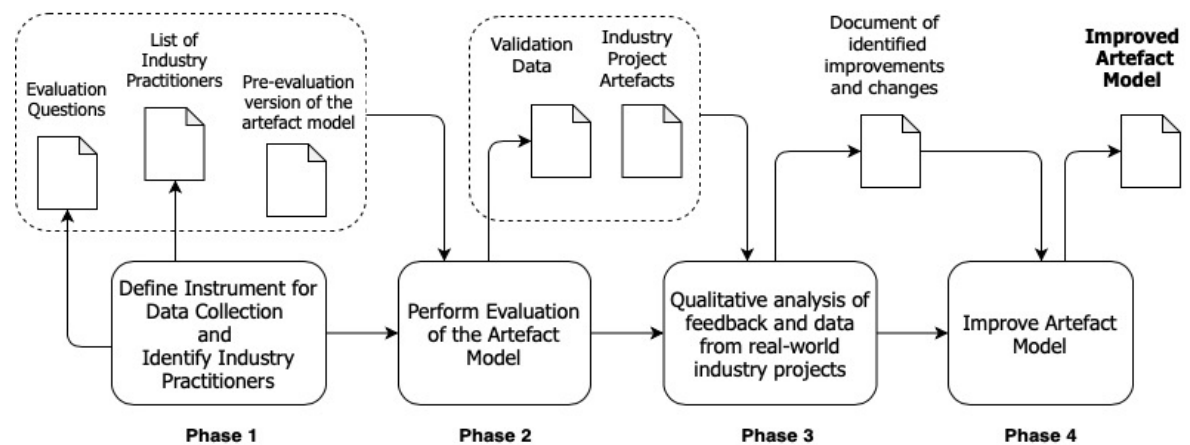


Figure 5.3: Model Validation Process

With reference to Figure 5.3:

- **Phase 1: Define Instrument for Data Collection and Identify Industry Practitioners.**
This phase consisted of defining an instrument for data collection to be used in the model evaluation study as well as identifying practitioners from Big Data software development projects. As outputs, we have an evaluation questionnaire (see appendix C) and a list of industry practitioners.
- **Phase 2: Perform Evaluation of the Artefact Model.** This phase consisted of the assessment of the pre-validation version of the artefact model. Having the evaluation questionnaire and the preliminary version of the model as an input, we invited practitioners from industry to participate in this study (convenience sampling). As output, we received filled questionnaires with feedback from the participant practitioners.
- **Phase 3: Qualitative Analysis of Feedback and Data from Real-world Industry Projects.** This phase consisted of the Qualitative analysis (thematic coding [17] of feedback from Phase 2) and data from industry projects (re. artefacts); Indexing/categorising text; Grouping artefact-types and information as per the RE reference model [18]: (i) business needs, (ii) requirements specifications, and (iii) systems specifications (see Section 5). Because we used a semi-structured questionnaire in the validation process, the data was captured in an organised and structured manner, facilitating the data analysis process. As output, we had a document of identified improvements to be made to the model.
- **Phase 4: Improve Artefact Model.** This phase consisted of improving the pre-validation version of the artefact model based on output from Phase 3, maintaining the model primitives as described in the model creation process [9]:(i) artefacts, (ii) relationships, and (iii) cardinalities. The output of this phase is the improved version of the artefact model.

5.4.2 Descriptive Statistics

Table 5.1 gives descriptive statistics of the participants. The subsections describe the results of the study.

Table 5.1: Descriptive statistics of the study participants.

Practitioner	Roles	Application Domains	Experience with Requirements and RE	Experience with Big Data
1	Business Analyst Developer Researcher	Marketing IT/Telecom	informal	5+ years
2	Requirements Analyst Developer Architect	IT/Telecom	1-4 years	3-5 years
3	Business Analyst Developer	IT/Telecom	1-4 years	1-2 years
4	Requirements Analyst Developer Architect Manager Consultant researcher	IT/Telecom Marketing	11-15 years	5+ years
5	Architect Developer Researcher	Marketing IT/Telecom Healthcare Defense/Military Commercial Cyber Security	1-4 years	5+ years
6	Requirements Analyst Developer Manager	Government Transport Manufacturing	1-4 years	1-2 years
7	Consultant Developer researcher	Geo-spatial data processing	1-4 years	1-2 years
8	Requirements Analyst Architect Developer	Government IT/Telecom	16+ years	3-5 years
9	Developer	Marketing IT/Telecom Geo-spatial Data Processing	1-4 years	3-5 years
10	Requirements Analyst	Quality Assurance IT/Telecom Transport Mobile Engineer	1-4 years	1-2 years

5.5 Evaluation Results

The following subsections discuss the validation results from specific angles: (i) accuracy and completeness of the model; and (ii) usefulness and generalisability of the model.

5.5.1 Accuracy and Completeness

The questions formulated to assess the accuracy and completeness of the model were divided into four Likert scale type of questions and one polar (yes-no) question followed by an open text-field.

Table 5.2 lists the four questions and practitioners responses. The responses fall predominantly within the “Strongly Agree” and “Agree” options for all the questions. When asked about the neutral choice made, practitioner #5 replied that some element names (e.g., data transformation requirements) could change depending on the project. Also, he indicated that not all projects follow the naming proposed by NIST [14], e.g., the term “data capability requirements” could be referred to as “platform requirements”. Likewise, practitioner #10 replied that “the relationships are okay and represent the way most of the applications are developed, but some other projects could have some different relationship labels”.

Following the Likert scale questions, we asked: Do you think any elements are missing from the proposed artefact model? Two practitioners (#2 and #9) answered “no”; whereas, the remaining eight participants answered “yes”. The suggestions from the “yes” respondents, were as follows: Practitioner #1 – non-functional requirements such as privacy and security should be depicted in the model. Practitioner #3 – the non-functional requirements related to the process (e.g., documentation quality and template patterns) could be introduced in the model. (We feel that the types and instances of non-functional requirements would likely differ from project to project). For example, some projects could have a catalogue of non-functional requirements focused on privacy and security whereas others could have a catalogue of non-functional requirements focused on performance and reliability. Thus, we decided not to include them explicitly in the model (for simplicity reasons). However, they can be considered as contained inside the “Non-functional Requirements Specifications” artefact, which is

represented in the model.

Practitioner **#4** – the “data analytics” type of requirements was missing. We clarified that these types of requirements were indeed represented in the model as “data transformation requirements” as classified by NIST (2015). Practitioners **#1**, **#3**, **#4** and **#10** – to include elements related to the artefacts for technological requirements for the project, e.g., those elicited concerning the data pipeline: data collection, storage, processing, visualisation, and management. (We agreed with the suggestion, thus adding the technological requirements related entities to the post-validation version of the model).

Practitioners **#5** and **#8** – to include a note or a specific element addressing the application type based on the nature of data processing, whether it would be batch or streaming. (The type of application based on the nature of data processing would play an important role in defining the systems requirements, however, it would not change the types of artefacts in the project. Adding the type of application as an entity would add complexity to the model. Thus, we decided to include an explanatory note linked to the entities denoting “Big Data Scenarios” and “Quality Attributes Scenario” since they would cover information regarding the type of application being dealt with in the project).

Finally, Practitioner **#7** – to better represent the entity denoting “Big Data scenarios”. Specifically, this label could be misleading because the scenarios are domain specific and do not describe only the data specific characteristics. (We agree with this recommendation. Thus, we added an explanatory note linked to the entities named “Big Data Scenarios” and “Quality Attributes scenarios”).

Improvements made to the model in response to the assessment, as well as the supporting rationale can be seen in Table 5.7.

Table 5.2: Results of the accuracy and completeness questions.

Questions	Strongly Agree	Agree	Neither agree nor disagree	Disagree	Strongly Disagree
1. To what extent do you agree that the schematic model reflects the type of RE artefacts in the development of Big Data applications in industry?	Practitioners 7 and 8	Practitioners 1, 2, 3, 4, 5, 6, 8, 9 and 10			
2. To what extent do you agree that the names of the artefacts depicted in the proposed artefact model are appropriate?	Practitioners 3, 4, 6, 7, and 9	Practitioners 1, 2, 8 and 10	Practitioner 5		
3. To what extent do you agree that the labels of the relationships in the artefact-model are appropriate?	Practitioners 3, 4, 6, 7, 9	Practitioners 1, 2, 5, 8.	Practitioner 10		
4. To what extent do you agree that the elements in the artefact model named: data-capability requirements, data-source requirements, data transformation requirements and data-consumer requirements represent the whole spectrum of the types of Big Data requirements?	Practitioners 7, 8 and 9	Practitioners 1, 2, 3, 4, 6 and 10		Practitioner 5	

5.5.2 Usefulness and Generalisibility

For assessing the usefulness of the artefact model, we asked the following question: To what extent do you agree that artefact model is useful in practice? Table 5.3 depicts that most of the participants agree or strongly agree that the model is useful in practice.

Table 5.3: Results of the usefulness question

Likert Items	Practitioner #
Strongly agree	3, 6, 7 and 8
Agree	2, 4, 9 and 10
Neither agree nor disagree	1 and 5
Disagree	–
Strongly disagree	–

Further, we asked the participants to give their opinion on the purposes the artefact-model would be useful for. Some example variety of answers we received are: Practitioner #2 – “to aid in requirements gathering and initial architecture design.” Also, “as a guiding template for customer and executive level presentations.” Practitioner #3 – “with a clear artefact model, it is

easier to go through all field/checklist that need to be considered in RE and in architecture design.” Practitioner #4 – “to support the development of specifications for Big Data applications, development of test cases based on the requirements, traceability of requirements through the development cycle” as well as for “getting a big picture view of the project and how it fits into the organisation.” Practitioner #6 – “used as a reference to support the review and elaboration of development processes and policies in companies that work with data-centric applications.” Practitioner #7 – “This work (the proposed artefact-model) is a first step in providing a solid set of artefacts for supporting practitioners to reason about RE in Big Data Apps.” Practitioner #8 – “the design of a Big Data application involves a series of requirements artefacts that, in my opinion, are captured by the proposed model.” Also, “support the specification, validation, and test of Big Data applications.” This view is also echoed by Practitioner #9. Practitioner #10 – “good start to help in the elicitation process. The requirements analyst could use it to guide in the interviews, focus groups and workshops with stakeholders in order to identify the most important or relevant requirements.” Finally, Practitioners #1 and #5 did not provide any opinion on usefulness.

Table 5.4 depicts a synthesis of categorised reasons and participants based on the analysis of total feedback received on usefulness.

Table 5.4: Reasons for usefulness of the model juxtaposed by participant groups

Reasons for usefulness of the model	Practitioner #
Reason 1: provide a big picture of requirements artefacts used/created in the project	4, 6, 7, 9 and 10
Reason 2: aid in requirements elicitation.	2, 3, 4, 6 and 10
Reason 3: aid in the definition of specific RE processes.	6, 9, and 10
Reason 4: aid in the specification, validation and testing of Big Data software applications.	4 and 8
Reason 5: aid in the architecture design; serving as template for executive presentations.	3 and 2

When asked: “*To what extent do you agree that the artefact model is generic enough to be used in different Big Data software development projects (with few modifications)?*”, most of the answers fell within the “agree” (six answers) and “strongly agree” (three answers) options (see Table 5.5). Thus, there is a consensus amongst the practitioners regarding the applicability of the artefact model in different projects, regardless of their unique characteristics.

Table 5.5: Results of the generalisability question

Likert Items	Practitioner #
Strongly agree	6, 9, and 10.
Agree	1, 2, 3, 5, 7, and 8.
Neither agree nor disagree	4
Disagree	–
Strongly disagree	–

5.5.3 Comparison between the proposed and improved versions of artefact model

In this section, we present and discuss the improvements made to the pre-validation version of the model in response to the feedback obtained in the validation study as well as present the post-validation version of the RE artefact model.

Table 5.6 shows that the model has changed drastically (in the total number of entities) – doubled – from 21 to 43 entities and tripled in terms of Big Data specific elements (from 6 to 18). Changes are due to missing elements in the pre-validation model (e.g., technological requirements, external interface requirements, data requirements) or implicit representation in the graphical nodes of the model (e.g., functional specifications contain functional requirements). Also, two new relationship types were added to the post-validation model (e.g., “assist-in” and “is-composed-of”). Additions, changes, and removals made to the original model are described in Table 5.7.

Table 5.6: Comparative statistics between the pre- and post-validation artefact models.

Likert Items	Pre-validation	Post-validation
Number of Elements	21	43
Number of Relationship types	5	6
Number of Big Data specific elements	6	18

Table 5.7: Model changes

Entities Added	Rationale
Technological Requirements Specifications	Big Data technologies play a critical role in storing, processing, and managing data. Early decisions in technology selection can help simplify development and aid in the definition of systems architecture.
System Architecture, Design Components, and Abstractions	These artefacts are influenced by requirements specifications and so their depiction in the artefact model renders the model more explicit.
External interface specifications	External interface specifications denote that the Big Data system will communicate with external components.
Data Requirements	Data Requirements are an inherent part of any Big Data system.
Relationships Added	Rationale
“Assist-in”	This relationship was added to represent the situation when one or more artefacts assist in the creation of one or more other artefacts (e.g., system requirements in the creation of system architecture).
“Is-composed-of”	This relationship denotes the “grouping” of artefacts (e.g., requirements specifications composed of functional and non-functional requirements).
Relationships Removed	Rationale
Is-identified-from	In improving the artefact model, this type of relationship was no longer needed.
Labels changed	Rationale
Data Capability Requirements (is changed to) Infrastructure Requirements	These labels (promoted by NIST [14]) were changed based on recommendations from the practitioners.
Data Transformation Requirements (is changed to) Data Processing Requirements	These labels (promoted by NIST [14]) were changed based on recommendations from the practitioners.

5.6 Post-validation version of the RE Artefact Model in the context of Big Data Software Development Projects

As a result of the validation study reported in this chapter, a new version of the BD-REAM was created (see Figure 5.4). The post-validation version of the artefact model depicts 43 entities (artefacts), six types of relationships, and several inter-connections. The entities are grouped into the following three categories extracted from the Requirements Engineering Reference Model [18]:

- **(1) Business needs artefacts:** These specify customer and strategic requirements, including product and business goals of the system under development [18]. In the post-validation version of the artefact model, seven entities fall in this group of artefacts (see pink-coloured nodes in Figure 5.4).
- **(2) Requirements specification artefacts:** These contain functional and non-functional

requirements. They are analysed and modelled from the customer and user perspectives and derived from (and justified by) the business needs [18]. In the post-validation version of the artefact model, 15 entities fall in this group of artefacts (see green-coloured nodes in Figure 5.4).

- **(3) Systems specification artefacts:** These contain a definition of the functional system concept; the required behaviour and its integration into the overall system and environment. It defines constraints on the design and realisation of the system [18]. In the post-validation version of the artefact model, 21 entities fall in this group of artefacts (see blue-coloured nodes in Figure 5.4).

The Big Data specific entities (artefacts) are: Big Data Requirements Specifications; Data Processing; Requirements Specifications; Data Consumer Requirements specifications; Data Source Requirements Specifications; Data Requirements Specifications; Big Data Scenarios; Technological Requirements Specifications; and their contained artefacts (e.g., Data requirements specifications contain data requirements and data modelling and linking details). These entities are depicted in the post-validation version of the artefact model in a rectangular shape with bold (darker) borders and integrated with the traditional entities (such as Systems Requirements Specifications, System Architecture, Design Components, and Abstractions, etc.) by the types of relationships depicted in the model and described in Table 5.8. Additionally, in Table 5.9, we provide definitions for the main entities depicted in the model presented in Figure 5.4.

Table 5.8: Descriptions of artefacts inter-relationships

Relationships	Descriptions
<i>Is-derived-from</i>	Two or more artefacts are said to be associated by “ <i>is-derived-from</i> ” relationship when from one artefact (e.g., Big Data scenarios) one or more artefacts can be derived and specified (e.g., Non-Software Requirements Specification <i>Is-derived-from</i> Big Data Scenarios)
<i>Is-part-of</i>	Two or more artefacts are said to be associated by a “ <i>is-part-of</i> ” relationship when one or more artefacts are part of one or more major artefacts (e.g., functional requirement <i>Is-part-of</i> software requirements);
<i>Is-composed-of</i>	Two or more artefacts are said to be associated by a “ <i>Is-composed-of</i> ” relationship when one or more artefacts are composed of one or more other artefacts (e.g., Software Requirements Specification <i>Is-composed-of</i> Functional Requirements Specifications)
<i>Contains</i>	Two or more artefacts are said to be associated by a “ <i>Contains</i> ” relationship when one or more artefacts have or hold information from another artefact within (e.g., software requirements <i>Contains</i> analysed requirements)
<i>Used-in</i>	Two or more artefacts are said to be associated by a “ <i>Used in</i> ” relationship when one or more artefacts can be used to guide in the definition of one or more artefacts (e.g., Constraints are <i>Used-in</i> Big Data Requirements Specifications)
<i>Assist-in</i>	Two or more artefacts are said to be associated by a “ <i>Assist-in</i> ” relationship when one or more artefacts assist in the definition of one or more artefacts (e.g., Software Requirements Specifications <i>Assist-in</i> the definition of Systems Architecture)

Table 5.9: Main Artefact Descriptions

Entity (Artefact)	Description
Business Case	Captures the reasoning for initiating a project. It is often presented in a well-structured written document. The logic of the business case is that, whenever resources such as money or effort are consumed, they should be in support of a specific business need [19].
Business Goals	Describe what a company expects to accomplish over a specific period of time. In software engineering, organisational goals drive the conception, creation, and evolution of software systems [20]. They are associated with the needs of the organization rather than the needs of the customers [9].
Business Models	Specifies the framework for finding a systematic way to uncover long-term value for an organisation while delivering value to customers and capturing value through monetisation strategies [21].
Business Plan	Provides a description of the business’s future, specifying what to do and how to do [22].
Customer and Stakeholders Needs	A customer can be internal or external to the organization [23]. Customers are stakeholders of the project and as such their ideas, needs and wishes are central to the project [8]. Customers needs describe their expectations regarding the product to be developed. It represent the views of those at the business or enterprise operations level that is, of users, acquirers, customers, and other stakeholders as they relate to the problem (or opportunity), as a set of requirements for a solution that can provide the services needed by the stakeholders in a defined environment [24].
Project Definition	Specifies project relevant information. It must fully document objectives and deliverables. Must be aligned to business objectives, address the needs of stakeholders and customers, and properly set the project team’s expectations [25].
Big Data Scenarios	Scenarios capture the system, as viewed from the outside (e.g., by a user, business, using specific examples) [26]. Big Data Scenarios are scenarios that incorporate Big Data characteristics in their descriptions (e.g., volume, velocity, variety, veracity, etc.)
Quality Attribute Scenarios	Scenarios that describe the usage of the software with respect to the quality attributes it might address (e.g., performance, privacy, etc.). A quality attribute scenario helps to derive quality-attribute-specific requirements applicable to the system [27].

Application Knowledge and Models	Domain and	Valid knowledge used to refer to an area of human endeavour, an autonomous computer activity, or other specialized discipline [28].
Constraints		Specify the restrictions or dictates the actions of the project team [29] (e.g., Scope, schedule, budget, quality, resources, limited software licenses, etc.) [26].
Technological Requirements	Re-quirements Specifica-tions	Specifies the technological requirements for the Big Data systems. Usually, the specification follows the Big Data analytics pipeline: Data Collection , Data Storage , Data Processing , Data Visualization, and Data Management
Integration Plan		specifies the process of incorporating smaller sub-systems into one larger system to ensure they all work together [30]
Non-software Requirements	Re-quirements Specifica-tions	In the context of this research, it represents all requirements specifications that are not related to the software itself (e.g., infrastructure requirements, project requirements, etc.).
Data Modelling and Linking Details		Specifies an abstract model that organizes elements of data and standardizes how they relate to one another and to the properties of real-world entities [31].
Data Requirements		Specifies directives or consensual agreements that define the content and/or structure that constitute high quality data instances and values [32].
Data Source Requirements		Refer to the set of requirements the system should address to support or deal with the different characteristics of the data sources (e.g., data size, file formats, rate of growth, at rest or in motion, etc.) [14].
Infrastructure Requirements	Re-quirements	Specifies Big Data infrastructure details (e.g., need to support legacy and advanced software packages, legacy and advanced computing platforms, data storage and elastic data transmission, hardware, networking [14]).
External Interface Requirements		Specifies hardware, software, and database elements with which a system or component must interface [33]
System Architecture and Design components and abstractions		Specifies the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system [27].
Software Requirements	Re-quirements Specifica-tions	Prescriptive statement to be enforced by the software to be developed and formulated in terms of phenomena shared between the software and the environment [28].
System Requirements	Specifica-tions	Specify all the requirements necessary for build the whole system and that includes hardware requirements, infrastructure requirements, project requirements, software requirements, for example.
Functional Requirements	Re-quirements	Specifies a function that a system or system component must be able to perform [23]. Describes what the system should do, how the system should react to particular inputs, and how the system should behave in particular situations [8].
Non-functional Requirements	Re-quirements	Specifies quality-related properties (e.g., performance, security, etc.) that the functional effects of the software should have [8].
Big Data Software Requirements	Re-quirements Specifica-tions	In the context of the BD-REAM, it is the artefact that contains the Big Data specific software requirements (e.g., Big Data processing requirements).
Data Processing Requirements	Re-quirements	Specifies the types of requirement that relate to data analytics, data fusion and data processing requirements [14]
Data Consumer Requirements	Re-quirements	Specifies the set of requirements related to the presentation of the processed results of Big Data to the users (e.g., processed results in text, table, visual, and other formats) [14].

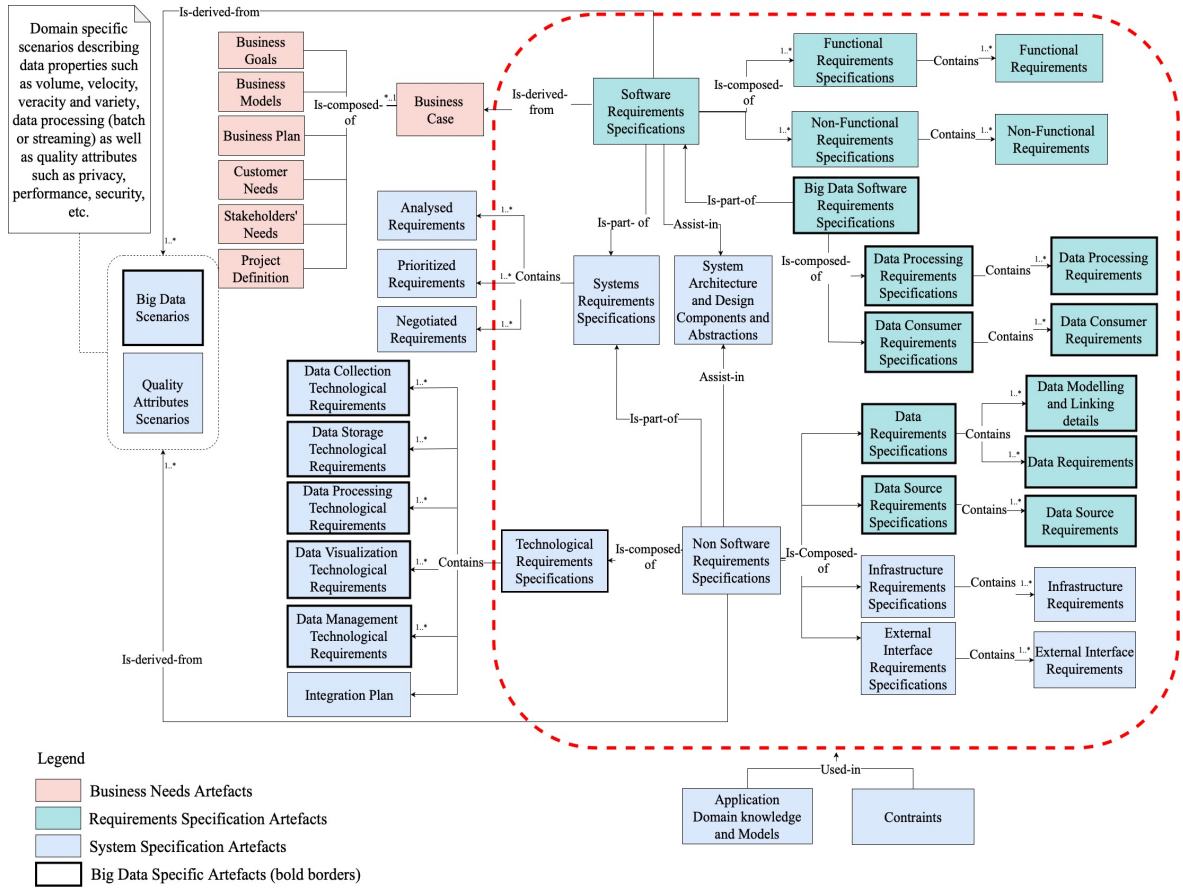


Figure 5.4: Graphical Representation of the Post-validation version of the Big Data RE artefact model (BD-REAM).

This model depicts nodes of three types: (i) Business Needs, (ii) Requirements Specification, and (iii) Systems Specification. The rectangles with heavy border-lines are Big Data elements. The two blue rectangles labelled ADM and Constraints at the bottom of the figure are Used in every rectangle encapsulated inside the red boundary. They have been factored out to simplify the diagram. **If the cardinality is not expressed in the association line, it means that it is has a value of 1.** For a tabular representation of the artefact model with all cardinalities expressed please refer to Table 5.11

5.7 Threats to Validity

We use Runeson and Hosts [34] guidelines to discuss the threats to validity and limitations of this research, and our approaches to mitigate them.

5.7.1 Construct Validity

Construct validity is concerned with the extent the studied constructs represent their real-life meanings [34]. Given the large number of artefacts and relationships (i.e., constructs) in the artefact model, construct validity takes heightened importance. One threat to construct validity is in the model assessment. It is possible that the participants misunderstood our intent. To mitigate this threat, we provided the artefact model along with a definition of the elements and relationships in the instrument. We also briefed the model individually prior to assessment and were available for clarification during the study. There were no clarification incidents.

5.7.2 Internal Validity

Threats to internal validity are concerned with confounding factors that may have influenced causal relationships in the study. Because our study does not involve causal relationships, this threat does not arise in the study.

5.7.3 External Validity

External validity is concerned with generalisability of the artefact model. Of course, with ten participants, we cannot claim strong generalisability across a large body of Big Data software development projects. However, the varied sources from which we have constructed the model (i.e., literature, expert opinion, and Big Data projects) give a first solid basis for applicability of the model in other projects. Regardless, the user is recommended to exercise caution when using the model in a real-life project.

5.7.4 Reliability

Reliability is concerned with whether the study can be repeated by other researchers and lead to the same results. This threat does exist for several reasons. For example, the participants background and experience would likely differ in another study and hence may induce variation in the results. Also, the questions in the instrument may be interpreted with variability. This threat was mitigated by using guidelines for instrument creation [35]. Also, the instrument was reviewed by all the three authors independently and any differences were resolved in consensus meetings over several iterations to ensure clarity and correctness. Another threat to reliability can result from the researchers subjective interpretation of the gathered data, leading to a biased artefact model. We addressed this threat by ensuring that all the artefact model elements are rooted in the scientific literature and data from actual Big Data projects. Also, we used thematic coding, an established process for qualitative research.

5.7.5 Selection Bias

Selection bias is a possible threat in this study due to the use of convenience sampling for selecting participants. Such bias can skew the resultant artefact model. However, practitioner knowledge and experience from diverse real-world Big Data projects helps to mitigate this threat.

5.7.6 Experience bias

Experience bias exists towards early period (1-4 years) of the participants in Big Data systems in industry. This threat remains at the early stage of the field of Big Data, but we hope that participants in the future studies on the artefact-model will have gained further experience to minimise this type of bias.

5.7.7 Conclusion Validity

Threats to conclusion validity are concerned with whether conclusions are traceable to the findings [34]. This threat is considered contained since all the conclusions presented are shown to have been rooted in specific sections of this chapter.

5.8 Summary

Whereas much attention has been given to analytics concerning Big Data, little amount of attention has been invested in the development of software applications and services leveraging Big Data. This situation is also reflected in the field of RE where domain models, processes, methods, techniques and tools have not yet embraced Big Data in a significant way. To ameliorate this situation, in 2017, we had created a preliminary RE artefact model to aid the development of Big Data software applications [13].

In this chapter, we describe how we have taken the early result to the next level by having the model validated by ten third-party practitioners from diverse Big Data software development projects. Specifically, the model was validated on its qualities such as: accuracy, completeness, usefulness, and generalisability (see Section 4). This chapter gives details of the validation study, such as descriptive statistics of the study participants and application domains in industry (see Subsection 4.1); data gathered and analysed (see Subsection 4.2); and the resultant, improved, artefact model (see Figure 2, Section 5). The validation results indicate that the model captures the key RE artefacts and relationships of a Big Data software development project, currently lacking in the literature. The validation results also confirm consensus amongst the study participants regarding the usefulness and applicability of the model in practice (see Table 5, section 4).

5.9 Chapter Addendum

In this chapter addendum, we describe some additional items:

- In section 5.4.1 of this chapter, we described the model evaluation process composed of four phases. Phase 3 consisted of the Qualitative analysis - thematic coding [17] - of feedback from Phase 2 and data from industry projects (requirements artefacts). In this addendum, We then, present some of the descriptive information (see Table 5.10) concerning the Big Data industry projects and examples of artefacts that were used to guide the model improvement process.
- In addition to the graphical representation of the Big Data Requirements Engineering Artefact Model (BD-REAM), we also represent it in a tabular form as described in Table 5.11.

Table 5.10: Descriptions of artefacts inter-relationships

<p>Project 1: Big Data Grapes</p> <p>Description: The Big Data Grapes project aims to develop and demonstrate powerful data processing technologies that will increase the efficiency of companies that need to take important business decisions dependent on access to vast and complex amounts of data. To catalyse the creation of a data ecosystem and economy that will increase the competitive advantage of companies that serve with IT solutions these sectors. It specifically tries to help companies across the grapevine-powered value chain ride the Big Data wave, supporting business decisions with real time and cross-stream analysis of very large, diverse and multimodal data sources.</p> <p>Example of Artefacts of this project: D2.1: Use Cases & Technical Requirements Specification; D2.3: BigDataGrapes Software Stack Design; D3.1: Data Modelling and Linking Components; D3.2: Data Ingestion and Integration Components; D6.1: Integrated Software Stack and APIs. The complete list of artefact can be seen at: http://www.bigdatagrapes.eu/deliverables</p>
<p>Project 2: Big Data Stack</p> <p>Description: The Big Data Stack project aim to deliver an infrastructure management system for the holistic management of computing, storage and networking resources, encompassing techniques for runtime adaptations of all BigDataStack operations.</p> <p>Example of Artefacts of this project: D2.4 - A conceptual model and reference architecture in BigDataStack; D2.2 - Requirements & State of the Art Analysis II; D5.1 - Dimensioning, Modelling And Interaction Services Of BigDataStack. The complete list of artefact can be seen at: https://bigdatastack.eu/deliverables</p>
<p>Project 3: Big Data Ocean</p>

Description: The BigDataOcean project strives to capitalize on modern technological innovations, utilizing them to revolutionize the way maritime-related industries work. The maritime sector, which is quite traditional and slow-moving, and has historically been unorganized and fragmented, is ripe for the introduction of innovations such as the big-data-driven economy, interrelated data streams from diverse sectors and languages, and cross-technology innovations that deliver data in several different formats (such as structured and unstructured, or real-time and in batches). These innovations will enable the creation of an entirely new value-chain, which will lead to great economic, societal, and environmental impact.

Example of Artefacts of this project: D2.1 Analysis Report on Big Data Components, Tools and Methodologies; D4.1 BigDataOcean Technology Requirements and User Stories; D4.4 Big-DataOcean Platform Architecture, Components Design and APIs v3.00; D4.5 BigDataOcean Final Platform Architecture, Components Design and APIs; D7.3 BigDataOcean Business Cases; D3.3 BigDataOcean Cross- Sector Semantics,Analytics and Business Intelligence Algorithms. The complete list of artefact can be seen at: <http://www.bigdataocean.eu/site/deliverables/>

Project 4: Big Data Special

Description:The SPECIAL project is motivated from the need for a simplified personal data management that complies with the General Data Protection Regulation (GDPR). The SPECIAL project addresses the contradiction between Big Data innovation and privacy-aware data protection by proposing a technical solution that makes both of these goals realistic. SPECIAL allows citizens and organisations to share more data, while guaranteeing data protection compliance, thus enabling both trust and the creation of valuable new insights from shared data.

Example of Artefacts of this project: D1.1 Use case scenarios V1 (M5); D1.2 Legal requirements for a privacy enhancing Big Data V1 (M6); D1.4 Technical requirements V1 (M8); D3.5 Scalability and Robustness testing report V2 (M27). The complete list of artefact can be seen at: <https://www.specialprivacy.eu/publications/public-deliverables>

Project 5: Big Data Europe

Description:Big Data Europe will undertake the foundational work for enabling European companies to build innovative multilingual products and services based on semantically interoperable, large-scale, multi-lingual data assets and knowledge, available under a variety of licenses and business models.

Example of Artefacts of this project: 3.2: Technical Requirements Specifications Big Data Integrator Architectural Design I; 3.3:Big Data Integrator Deployment and Component Interface Specification; 3.5: Big Data Platform Requirements, Architecture and Usage; 5.2: Domain-Specific Big Data Integrator Instances I. The complete list of artefact can be seen at: <https://www.big-data-europe.eu/results/>

Table 5.11: Tabular Representation of the Post-validation version of the BD-REAM

Artefact	Relation	Cardinality	Artefact
Business Case	<i>Is-composed-of</i>	1..*	Business Goals
			Business Models
			Business Plan
			Consumer Needs
			Stakeholders Needs
			Project Definition
Software Requirements Specifications	<i>Is-Composed-of</i>	1..*	Functional Requirements Specifications
			Non-functional requirements Specifications
			Big Data Requirements Specifications
Big Data Requirements Specifications	<i>Is-Composed-of</i>	1..*	Big Data Processing Requirements Specifications
			Big Data Consumer Requirements Specifications
Non-software Requirements Specifications	<i>Is-Composed-of</i>	1..*	Data Requirements Specifications
			Big Data Source Requirements Specifications
			Infrastructure Requirements Specifications
			External Interface Requirements Specifications
Systems Requirements Specifications	<i>Contains</i>	1..*	Analysed Requirements
			Prioritised Requirements
			Negotiated Requirements
Data Requirements Specifications	<i>Contains</i>	1..*	Data Requirements
			Data Modelling and Linking Details
Infrastructure Requirements Specifications	<i>Contains</i>	1..*	Infrastructure Requirements
External Interface Requirements Specifications	<i>Contains</i>	1..*	External Interface Requirements
Technological Requirements Specifications	<i>Contains</i>	1..*	Data Collection Technological Requirements
			Data Storage Technological Requirements
			Data Processing Technological Requirements
			Data Visualization Technological Requirements
			Data Management Technological Requirements
			Integration Plan
Functional Requirements Specifications	<i>Contains</i>	1..*	Functional Requirements
Non-functional Requirements Specifications	<i>Contains</i>	1..*	Non-Functional Requirements
Big Data Processing Requirements Specifications	<i>Contains</i>	1..*	Big Data Processing Requirements
Big Data Consumer Requirements Specifications	<i>Contains</i>	1..*	Big Data Consumer Requirements
Software Requirements Specifications	<i>Is-Part-Of</i>	1..1	Systems Requirements Specifications
Non-Software Requirements Specifications	<i>Is-Part-Of</i>	1..1	Systems Requirements Specifications
Software Requirements Specifications	<i>Assist-In</i>	1..1	Systems Architecture and Design Components Abstractions
Non-Software Requirements Specifications	<i>Assist-In</i>	1..1	Systems Architecture and Design Components and Abstractions
Non-Software Requirements Specifications	<i>Is-derived-from</i>	1..*	Big Data Scenarios
Non-Software Requirements Specifications	<i>Is-derived-from</i>	1..*	Quality Attributes Scenarios
Software Requirements Specifications	<i>Is-derived-from</i>	1..*	Big Data Scenarios
Software Requirements Specifications	<i>Is-derived-from</i>	1..*	Quality Attributes Scenarios
Application Domain Knowledge	<i>Used-in</i>	1..1	All entities within the red boundary (see Figure 5.4)
Constraints	<i>Used-in</i>	1..1	All entities within the red boundary (see Figure 5.4)

Bibliography

- [1] V. Dipti Kumar and P. Alencar, “Software engineering for big data projects: Domains, methodologies and gaps,” *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, pp. 2886–2895, 2016.
- [2] N. H. Madhavji, A. Miranskyy, and K. Kontogiannis, “Big Picture of Big Data Software Engineering: With Example Research Challenges,” *Proceedings - 1st International Workshop on Big Data Software Engineering, BIGDSE 2015*, pp. 11–14, 2015.
- [3] D. Arruda and N. H. Madhavji, “State of requirements engineering research in the context of big data applications,” in *Requirements Engineering: Foundation for Software Quality*, E. Kamsties, J. Horkoff, and F. Dalpiaz, Eds. Cham: Springer International Publishing, 2018, pp. 307–323.
- [4] 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*, May 2015, pp. 19–25.
- [5] R. N. Laigner, M. Kalinowski, S. Lifschitz, R. S. Monteiro, and D. D. Oliveira, “A Systematic Mapping of Software Engineering Approaches to Develop Big Data Systems,” no. July, 2018.
- [6] V. D. Nadkarni, A. (2016) Worldwide big data technology and services forecast, 2016 - 2020. [Online]. Available: <https://www.marketresearch.com/IDC-v2477/Worldwide-Big-Data-Technology-Services-10510864/>
- [7] R. . Davenport. T.H., Bean. (2019) Big companies are embracing analytics, but most still dont have a data-driven culture. [Online]. Available: <https://hbr.org/2018/02/big-companies-are-embracing-analytics-but-most-still-dont-have-a-data-driven-culture>
- [8] I. Sommerville, *Software Engineering*, 2009.
- [9] B. Berenbach, D. Paulish, J. Kazmeier, and A. Rudorfer, *Soft. Systems Requirements Eng. In Practice*, 2009.
- [10] B. Penzenstadler, D. M. Fernandez, and J. Eckhardt, “Understanding the impact of artefact-based RE - Design of a replication study,” *International Symposium on Empirical Software Engineering and Measurement*, pp. 267–270, 2013.
- [11] M. R. I. Nekvi and N. H. Madhavji, “Impediments to Regulatory Compliance of Requirements in Contractual Systems Engineering Projects,” *ACM Transactions on Management Information Systems*, vol. 5, no. 3, pp. 1–35, 2014. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=2666081.2629432>

- [12] R. S. Pressman, *Software engineering: a practitioners approach*, 2001.
- [13] D. Arruda and N. H. Madhavji, "Towards a Requirements Engineering Artefact Model in the context of Big Data Software Development Projects," *Proceedings of the IEEE International Conference on Big Data*, pp. 2232–2237, 2017.
- [14] "NIST Big Data Interoperability Framework: Volume 3, Use Cases and General Requirements," vol. 3, p. 260, 2015. [Online]. Available: <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.1500-3.pdf>
- [15] M. Shaw, "Writing Good Software Engineering Research Papers," in *International conference on software engineering (ICSE'03)*, vol. 6, 2003.
- [16] R. Likert, *A technique for the measurement of attitudes.*, 1932.
- [17] G. Gibbs, *Analyzing qualitative data, Qualitative research kit*, 2007.
- [18] E. Geisberger, M. Broy, B. Berenbach, J. Kazmeier, D. Paulish, and A. Rudorfer, "Requirements Engineering Reference Model (REM)," Tech. Rep.
- [19] Wikipedia. (2019) Business case. [Online]. Available: https://en.wikipedia.org/wiki/Business_case
- [20] P. Clements and L. Bass, "Relating business goals to architecturally significant requirements for software systems," SEI - Research, Technology, and System Solutions Program, Tech. Rep., 2010.
- [21] G. Cuofano. (2019) What is a business model? 30 successful types of business models you need to know. [Online]. Available: <https://fourweekmba.com/what-is-a-business-model/>
- [22] Entrepreneur. (2019) Business plans: A step-by-step guide. [Online]. Available: <https://www.entrepreneur.com/article/247574>
- [23] "Ieee standard computer dictionary: A compilation of ieee standard computer glossaries," *IEEE Std 610*, pp. 1–217, Jan 1991.
- [24] A. G. to the Systems Engineering Body of knowledge. (2019) Stakeholder needs and requirements. [Online]. Available: https://www.sebokwiki.org/wiki/Stakeholder_Needs_and_Requirements
- [25] D. Finnefrock. (2006) Project management artifacts. [Online]. Available: <https://www.projectmanagement.com/wikis/233503/Artifacts>
- [26] J. M. Siegelaub, "Six (yes six!) constraints: an enhanced model for project controll," in *PMI Global Congress*, 2007.
- [27] Wikipedia. (2019) Systems architecture. [Online]. Available: https://en.wikipedia.org/wiki/Systems_architecture
- [28] A. v. Lamsweerde, *Requirements Engineering: from system goals to UML models to software specifications*, 2009.
- [29] (2017) Project management tips: Guidance for real life situations. [Online]. Available: <http://pmtips.net/blog-new/defining-project-constraints>
- [30] J. Fischer. (2019) 5 questions to think about when planning system integration. [Online]. Available: <https://www.ariasolutions.com/5-questions-think-planning-system-integration/>

- [31] Wikipedia. (2019) Data model. [Online]. Available: https://en.wikipedia.org/wiki/Data_model.
- [32] (2019) Create data requirements. [Online]. Available: http://semwebquality.org/mediawiki/index.php?title=Create_Data_Requirements
- [33] H. Richard, "Software System Engineering: A Tutorial," *Computer*, vol. 35, pp. 68–73, 2002.
- [34] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2009.
- [35] M. N. Douglas R. Berdie, John F. Anderson, *Questionnaires: Design and Use*, 1986.

Chapter 6

An Approach for Modelling Quality Requirements for Big Data Applications

6.1 Introduction

Requirements Engineering (RE) is considered to be of foundational importance in software development [1]. RE provides ways for understanding the needs and desires of the customers and users, for assessing the feasibility of a project, for negotiating solutions, for analysing and specifying the requirements of the proposed solution, for prioritising among the requirements for implementation, for validating the requirements, and for managing the requirements throughout the systems life-cycle [2].

Now that Big Data is on the scene, one can argue that, due to added complexity, the role of RE is even more critical in the creation and evolution of Big Data-oriented applications. These applications, like traditional applications, serve customers and end-user needs except that we expect improved, even different, experience from the system as it leverages the underlying Big Data to provide responses. However, as yet there is no recognisable body of RE knowledge concerning the development of such hybrid applications. Moreover, quality demands in the design of such applications are higher than in traditional software applications [3] design due to not only the specific data characteristics (also known as the V characteristics: volume, velocity,

veracity, and variety) and the exponential increase of data sets and data rates but also due to the fact that Big Data applications are complex solutions composed of dynamic components such as distributed computation nodes, networks, databases, middleware, and business intelligence layer [4]. Not surprisingly, some researchers have discussed quality attributes such as performance, security, privacy, scalability, portability, and reliability [5] in the context of Big Data applications whereas some others have highlighted challenges posed by Big Data in the development of software applications, e.g.: data inconsistency, inadequate resources, scalability constraints [6]; security risks and predicting threat sources in real-time [7]; transparency and individual consent [8] dynamic changes in requirements [9], verifiability of Big Data systems requirements [10] to name a few. Quality demands are represented in form of quality requirements (also known as non-functional requirements) which can be specified in natural language and modelled using diagrams and visualisations techniques.

Problem Statement and Principal Idea. The Big Data characteristics pose serious challenges to achieving high system quality standards for security, performance, scalability, privacy and other quality requirements [11]. However, while requirements engineering (RE) has long been recognised as critical for downstream development of computer systems [12], the entire field is basically passive about how to deal with characteristics of data in the RE process in the development of Big Data software applications [11]. This raises the question as to the extent to which the Big Data challenges are addressed in the solution design. Thus, in this chapter, we describe a goal-oriented approach for modelling quality requirements for Big Data applications that incorporates both Big Data characteristics (e.g., volume, velocity, veracity, and variety) and traditional systems quality attributes (e.g., scalability, performance, security, etc.) (see Section 6.3). The proposed approach is composed of five systematic steps organized into two phases: (i) Pre-modelling which consists of analysing requirements statements from several requirements sources (e.g., stakeholders, workshops, scenarios, etc.) with the aim to extract requirements relevant information (e.g., goals, quality attributes, data characteristics, etc.); and the (ii) modelling phase which consists of using the extracted information to build the goal models.

As proof of concept, we utilised requirements descriptions from real-world Big Data software development projects to demonstrate the feasibility of the proposed approach (see section

6.5). Our feasibility analysis indicates that it is possible to specify - through modelling - quality requirements that integrate both Big Data characteristics and traditional systems quality attributes, aiding in more complete requirements specifications which, in turn, may assist in creating quality Big Data software applications.

Significance of Research and Contributions. Requirements specifications are critical for downstream software development [12]. Proper requirements representation facilitates communication of requirements and translation into systems design [13]. However, concrete ways of specifying quality requirements for Big Data software applications are still lacking [9, 11]. The proposed approach is unique and, thus, should add to current RE knowledge base in the context of Big Data applications.

The contributions of this chapter are fivefold: (i) systematic process for specifying quality requirements for Big Data applications; (ii) a template for logging requirements information; (iii) checklists; (iv) a Big Data goal-oriented requirements language; and (v) a prototype tool that implements the proposed Big Data requirements language;

Chapter Structure. Section 6.2 discusses the related work. Section 6.3 described the QualiBD approach. Section 6.4 described the prototype tool. Section 6.5 presents the feasibility analysis of the proposed QualiBD approach. Finally, Section 6.6 summarises the chapter and provides recommendations for further work.

6.2 Related Work

Scientific research aimed at understanding the elicitation, specification, modelling, analysis, prioritisation and management of Big Data system requirements (both functional and non-functional) is still in its early stages. However, there has been an emerging effort to further contribute to the process of engineering Big Data Software Applications.

Some researchers have focused on discussing the characteristics and requirements of Big Data Analytics applications focusing on data acquisition, preservation, pre-processing, processing and visualization [14] as well as requirements engineering challenges in the context of Big Data systems [4, 10, 15, 16]; whereas some other researchers have focused on defining

RE collaboration process models [17]; data-mining approaches for semi-automatic requirements elicitation for data analytics [18]; user requirements based service identification [19]; RE artefact-models for Big Data software development projects [20, 21]; and processes for handling both privacy and performance requirements in user stories for Big Data projects in scrum [22].

Alternatively, some research has focused on the specification and modelling of Big Data systems requirements. For example, in [11], it is introduced the idea of intersecting a Big Data characteristic (e.g., volume, velocity, etc.) with a quality attribute (e.g., security, performance) in the requirements description, thus, guaranteeing that Big Data characteristics are addressed in the specification of quality requirements.

In [23], while no specific domain language was proposed, the authors defined four general requirements for Big Data systems (e.g., huge databases capacity, fine database performance, quality and structure of the data, and privacy and security) and modelled them using *i** and KAOS, which are general-purpose modelling languages. However, we found that the four generic requirements defined in this paper do not represent the spectrum of general Big Data systems requirements. Big Data systems are complex systems composed of distributed computation nodes, networks, processing models, analytics models, middleware, and business intelligence layers [4, 15]. Their requirements go beyond database related quality requirements. Moreover, the authors claimed that the resultant models can then be used as references in the modelling of both generic functional and quality requirements for Big Data software systems. Still, there is no empirical evidence regarding the usefulness and generalisability of the modelled requirements.

In [24], privacy extensions to UML use cases diagrams to help software engineers to visualize privacy requirements as well as to design privacy into Big Data applications is proposed. This solution is implemented as MS Visio extension ribbon in Visual Studio. The authors argued that these extensions to UML help software engineers to visually and quickly model privacy requirements in the analysis phase of the RE process. As a proof of concept, a prototype was created to show the usefulness of the extension and how it can be used to model the privacy requirements for Big Data systems in the domain of healthcare. However, while the tool proposed in [24] provides a step towards incorporating privacy annotations in the UML

use case diagrams, it does not provide information about the requirements themselves in the context of Big Data.

6.2.1 Discussion

Finally, the approach proposed in this chapter and described in the following sections differs from the works described in this related work analysis in four ways:

- We provide a systematic way for specifying Big Data quality requirements that incorporate Big Data characteristics and traditional systems quality attributes, resulting in more complete requirement specifications.
- The proposed approach also enables the identification of possible solution alternatives to address the identified quality requirements.
- We propose a Big Data domain specific goal-oriented modelling language that allows the representation of systems quality requirements easily in a graphical manner.
- We created a prototype tool that implements the proposed requirements language realising the use of the proposed approach.

6.3 The QualiBD Approach

In this section, we describe QualiBD, an approach that enables the specification, through modelling, of Big Data quality requirements that incorporate traditional systems quality attributes (such as latency, scalability, and confidentiality) and Big Data characteristics (such as volume, velocity, variety, and veracity) on the same requirement representation.

The QualiBD approach is composed of a systematic process, requirement logging template, checklist, Big Data requirements language, and prototype tool. Also, it is organised into two phases: (i) pre-modelling (one focuses on the identification and specification (in natural language) of systems quality requirements), and (ii) modelling (one focuses on modelling the encoded quality requirement from the pre-modelling phase). The requirements language

proposed as part of the QualiBD Approach was built upon the concepts of the Softgoal interdependency (SIG) graph of the NFR framework [25]. It was intentionally designed to be simple and easy-to-use. Thus, supporting the basic elements necessary to address the main goal of this research which is to specify quality requirements for Big Data applications that incorporate both data characteristics and systems quality attributes in the same requirement representation.

6.3.1 Concepts and Assumptions

In this subsection, we describe the concepts and assumptions underlying the QualiBD approach.

Goals. Goals are statements of intentions and desired outcomes of a system under consideration [26]. Goals can be considered as functional and non-functional (usually representing functional and non-functional requirements, respectively). In the QualiBD approach, we use goals as the main guiding concept in defining specifications of Big Data quality requirements and their associated solution alternatives.

Permutations and Permutations Attributes. Permutation is the conjunction of one or more Big Data characteristic (e.g., volume, velocity, etc.) with one or more quality attributes (e.g., performance, scalability, privacy, etc.) [11]. Examples of permutations are: Velocity x Latency; Volume x Scalability; Veracity x Security x Performance. In such permutations are attributes with specific values. Example values are: Data format (e.g., unstructured data, structured data, and Semi-structured data) and quantitative information (response time of 1.5 seconds, latency of 0.5-2.0 seconds, and throughput of 1TB of data per 30 minutes). The purpose of such a permutation in RE is to capture both Big Data attribute (s) and traditional quality attribute (s) in one requirement. It is possible that a Big Data characteristic may intersect with more than one quality attribute (e.g., veracity x security x performance), as multiple quality attributes can impinge on a given system component. This facilitates capturing of more complex conditions of quality/data.

Assumptions. The QualiBD approach was designed underlying the following assumptions:

- The approach requires the user (e.g., requirements analyst, business analyst, solutions

architect, etc.) to have knowledge about the application domain, Big Data and technologies.

- Big Data characteristics are treated as attributes that along with traditional quality attributes pose constraints to the operation of the system.
- The Big Data system project is characterised by at least one Big Data characteristic and a set of quality attributes to be addressed in the design of the solution, allowing for the creation of attributes permutations, the main concept introduced in this approach.

In section 6.3.2, we first describe the overall process of the QualiBD approach. Then, in Section 6.3.4, we describe our its proposed goal-oriented requirements language.

6.3.2 Overall process of the QualiBD Approach

The QualiBD approach, goal-oriented, is composed of five steps organised into two phases: (1) pre-modelling phase, which comprises a systematic process for reasoning about Big Data quality requirements that incorporate both data characteristics and quality attributes in order to determine the candidate solutions to fulfil a given requirement; and (2) modelling phase, which consists of an easy-to-use goal-oriented modelling language to support the graphical representation of the identified requirements. Figure 6.1 depicts the process.

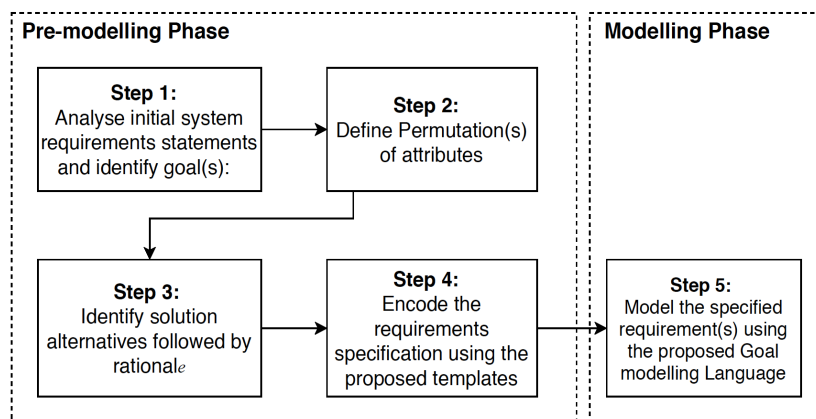


Figure 6.1: Process overview of the proposed approach

With reference to figure 6.1:

- Step 1: Analyse initial system requirements statements and identify goal(s): consists of the analysis of initial systems behavioural descriptions from different sources (e.g., scenarios, systems vision, domain properties, interviews, etc. [2]) in order to identify requirements statements and their associated goals, Big Data characteristics and quality attributes.
- Step 2: Define permutations of attributes: consists of defining the permutations based on the defined goal and its associated Big Data characteristic (s) and quality attribute (s). Referred to in this chapter as Data Characteristic and Quality attributes permutation. While identifying the permutations, it is important to make sure that the permutations relate to the requirement description and goal identified in Step 1.
- Step 3: Identify solution alternatives followed by rationale: brainstorming and definition of possible solution alternatives based on the analysis of the requirement description, its associated goal (s), and permutations. A rationale should follow each identified solution alternative.
- Step 4: Encode the requirements specification using the proposed templates: consists of formalising the information analysed so far using the proposed requirements specification template (as depicted in Figure 3) that will later be used as input in the modelling phase.
- Step 5: Model the specified requirements using the goal-oriented modelling language: this step consists of translating the encoded Big Data quality requirements (from step 4 of the pre-modelling phase) into goal model elements.

6.3.3 Checking for consistency while transitioning between phases

In order to minimise errors while identifying and logging the requirements information (e.g., goal, permutations, permutation attributes, etc.) needed for use in the modelling phase, we propose a checklist to be used within the process described in Section 6.3.2. In RE, checklists are powerful tools that assist in understanding various RE sub-processes and activities. In eliciting

and documenting systems requirements, checklists can be used to measure the completeness, accuracy, and efficiency of such requirements specifications [27]. In Figure 6.2, we describe the checklist for checking the completeness and accuracy of the encoded quality requirements. This checklist is to be used at the end of Step 4 in the Pre-modelling Phase.

Checklist Items	Status
Goal	
Is the goal clearly stated in the requirement template?	<input type="checkbox"/>
Is the identified goal related to the analysed requirement description?	<input type="checkbox"/>
Big Data Characteristics and Quality Attributes	
Are all the identified Big Data characteristics stated in the requirement template?	<input type="checkbox"/>
Are all the identified quality attributes stated in the requirement template?	<input type="checkbox"/>
Are all the identified Big Data characteristic and quality attributes related to the state requirement description?	<input type="checkbox"/>
Permutations	
Are all the possible permutations stated in the requirement template?	<input type="checkbox"/>
Are all the identified permutations between one or more Big Data characteristics and one or more quality attributes?	<input type="checkbox"/>
Permutation Attributes	
Are all the identified permutation attributes stated in the requirements template?	<input type="checkbox"/>
Are the identified permutation attributes related to the identified permutation (s)?	<input type="checkbox"/>
Solutions	
Are all the identified solution alternatives stated in the requirements template?	<input type="checkbox"/>
Are the rationale to the identified solution alternative clearly stated in the requirements template?	<input type="checkbox"/>

Figure 6.2: Checklist for the completeness and accuracy of the encoded quality requirement

6.3.4 The Big Data Goal-oriented Requirements Language

In the following subsequent sections, we describe the proposed Big Data goal-oriented requirements language (part of the QualiBD Approach) to translate the encoded Big Data quality requirements (from step 4 of the pre-modelling phase) into model elements.

The Model Elements

The proposed Big Data goal-oriented requirements language consists of the following modelling elements: Goal, NFR Soft-goal, Big Data Characteristic, Permutation, Permutation Attributes, Operationalising Soft-goal, Claim Soft-goal, Association Link, Permutation Link, Operationalisation Link, Argumentation Link, Decomposition Links, and Contribution Links. The aforementioned elements are further described in Figure 6.3.
















Model Elements	Description	Notation
Nodes		
Goal	Used to represent the overall requirement goal.	
NFR Soft-goal	Used to represent the identified quality attribute (s).	
Big Data Characteristic	Used to represent the identified Big Data characteristic (s).	
Permutation	Used to represent the identified permutation (s).	
Operationalising Soft-Goal	Used to represent identified solution alternative (s).	
Claim Soft-goal	A short statement used to support the identified solution alternative (s)	
Attributes		
Permutation Attributes	Used to represent the attributes (with specific values) used to further characterise the identified permutations.	
Refinements		
Association Link	Used to express associations amongst model elements	
Permutation Link	Used to express permutations amongst model elements.	
Decomposition Links	Used to express decompositions in the model. There are three types of decompositions: 'AND', 'OR', and 'Equal'. A 'AND' decomposition means that in order for the parent node be satisfied all child nodes must also be satisfied. A 'OR' decomposition means that in order for the parent node be satisfied at least one child node must be satisfied. A 'Equal' decomposition means that means that the parent is satisficeable if and only if the child is.	  
Argumentation Link	Used to connect a Claim Soft-goals to a Decomposition Links.	
Contribution Links	Used to express the Contribution Links. There are two types of contributions: (i) Positive (green arrow); and (ii) Negative (red arrow).	 

Figure 6.3: Model Elements Description and Graphical Notation

The modelling elements described in Figure 6.3, interact to one another in the following ways:

- Goals are associated with one or more NFR Soft-goals and one or more Big Data Characteristics.
- Refinements from Goals to NFR Soft-goals and Big Data Characteristics are done by an Association Link.
- NFR Soft-goals and Big Data Characteristics are refined into a Permutation Container by a Permutation Link.
- Permutation Containers can have zero or more Permutation Attributes. They are used to further characterise the defined permutations.
- Refinement links are also used to connect the lower level (Operationalising Soft-goals) nodes that represent solution alternatives to their parent node (Permutation Container or refined NFR Soft-goal) through what we call a Decomposition link.
- Decomposition Links can also be used to decompose Operationalising Soft-goals into more concrete Operationalising Soft-goals. For instance, an Operationalising Soft-goal labeled encryption at rest can be refined by a Decomposition Link into more concrete Operationalising Soft-goals such as (i) FFE File and Folder Encryption and (ii) VTE - Vormetric Transparent Encryption.
- When possible, Operationalising Soft-goals should be accompanied by a Claim Soft-goal, i.e., a short statement provided to support the solution alternative proposed in the model (connected through an Argumentation Link).
- Once Operationalising Soft-goals are identified, then contribution links can be used to express the contribution of one Operationalising Soft-goal to other nodes (such as NFR Soft-goals or Permutation Containers).

How are the modelling elements graphically organised?

The modelling elements described in the previous section are graphically represented in levels of abstraction as follows (see Figure 6.4 for reference):

- Goal Level: The Goal element is represented.
- Big Data Characteristic and Quality Attribute Level: We represent the Goals associated Big Data characteristics and NFR Soft-goals elements that allows for the definition of permutations of data characteristic and quality attributes.
- Permutation Level: all the possible permutations identified are expressed in this level.
- Operationalisation Level: Operationalising Soft-goal refinements that represent requirements solutions are expressed in this level. They are derived either from an NFR Soft-goal (when there is no permutation with a Big Data characteristics) or from the Permutation Container (when there is a permutation of a Big Data characteristic with a quality attribute).

Illustrative Example

In this section, we describe a scenario requirement and its corresponding modelled quality requirement to illustrate the usage of the QualiBD approach.

Consider the following scenario (adapted from [11]): *“Project A aims to develop a Big Data-based earthquake real-time monitoring application. The solution should be able to distinguish natural from induced seismicity and measure the impact of detected seismicity via real-time ground motion measurements. The software application shall use a stream-processing engine with a latency of 0.5 - 2.0 seconds to respond to data in real-time between global earthquake sensors and the data centre. The application shall be able to deal with unstructured data (such as sensor and program logs). It is expected that the monitoring results would be displayed in a real-time dashboard that allows for different types of data visualisation...”*

From the analysis of this information, we derive the following requirement information:

- **Requirement Description:** The system shall use a stream-processing engine with a latency of 0.5 - 2.0 seconds to respond to data in real-time between global earthquake sensors and the data centre.
- **Goal:** To process sensor data in real-time.

- **Big Data characteristic:** Velocity.
- **Quality Attribute:** Latency.
- **Permutation:** Velocity x Latency.
- **Permutation Attributes:** Streaming of unstructured data; Latency of 0.5-2.0 seconds.

Based on the derived requirements information, one identifies solution alternatives (Operationalising Soft-goals) and their corresponding rationale (Claim Soft-goals), e.g.: (i) Samza, (ii) Storm, and (iii) Spark. This information would then be used to model the requirement, as depicted in Figure 6.4.

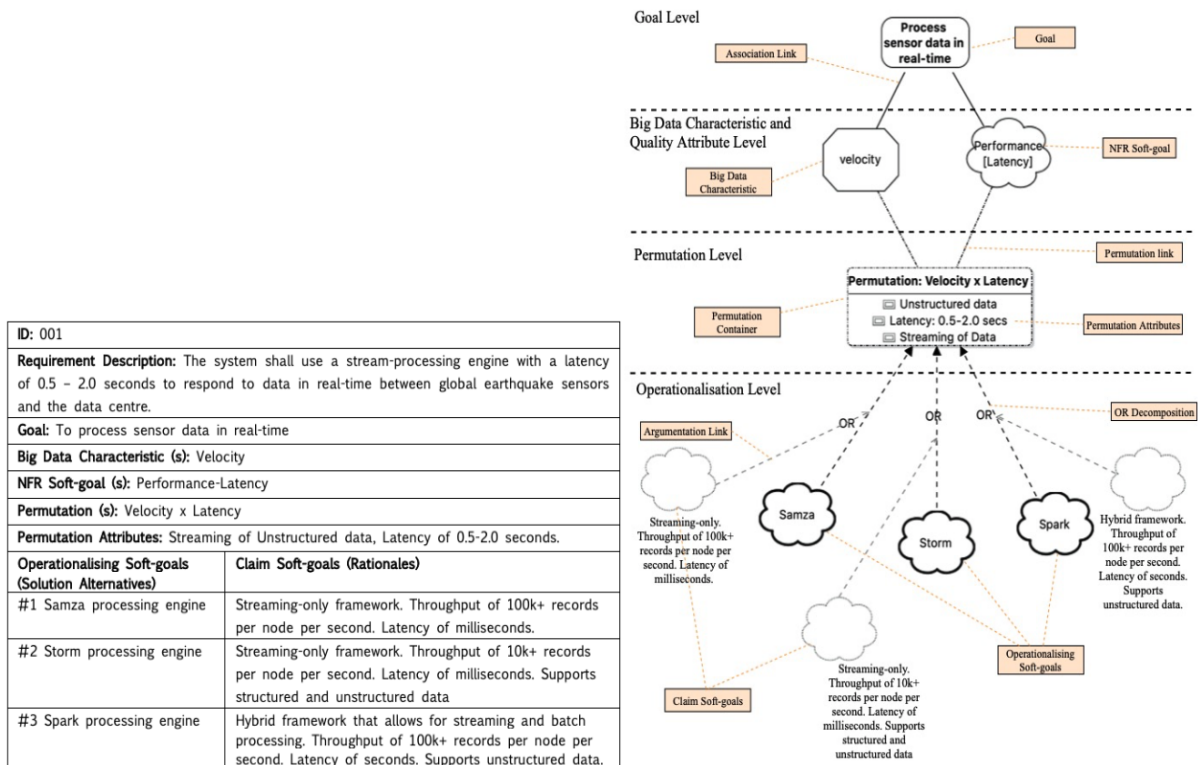


Figure 6.4: Example of a Big Data quality requirement modelled using the proposed QualiBD approach.

Depicted in this Figure: On the left, the requirement specified in natural language and encoded using the template proposed in this chapter. On the right, the same quality requirement modelled using the described Big Data goal-oriented modelling language organised by levels of description.

Considerations for using the QualiBD Approach

- The approach can be used to model either individual requirements or multiple requirements if the aim is to create a catalogue of quality requirements for the project.
- The modelling of multiple requirements on the same diagram could result in a large diagram with several connections that, in turn, would add complexity to the final specification.
- To avoid repetition of elements in the diagram - when modelling multiple quality requirements - one Big Data Characteristic and one NFR Soft-goal can be associated to one or more parent nodes (Goal).
- Identified solutions alternatives (Operationalising Soft-goals) are better defined if accompanied by a rationale (Claim Soft-goal).
- The modelling of requirements with multiple permutations allows for the analysis of the impact of one operationalisation on different parenting nodes (permutations).
- Please note that, in this chapter we use scenarios to illustrate our approach. However, the necessary modelling information (extracted using the process proposed in the pre-modelling phase in Section 6.3.2) can be identified from several sources (e.g., use-cases, interviews with stakeholders, workshops, etc.) at the discretion of the project.

6.4 Tool Support

In this section, we describe QualiBD, a modelling tool that implements the described goal-oriented requirements language (see Section 6.3.4). We first present features supported by the proposed modelling tool, and then we briefly discuss the technologies and frameworks used in the tool definition.

6.4.1 Tool Features

The modelling tool facilitates graphical modelling of Big Data application requirements in the style of WYSIWYG paradigm [28]. This includes drag and drop features and editing of entities and relationships in the model. Automatic labelling of a permutation containers (based on parent node labelling) simplifies model creation. In addition, currently, there are rudimentary analysis capabilities such as raising caution when there are: missing or duplicate relations in the model; missing labels; and permutation containers without permutation attributes.

6.4.2 Tool Implementation

The implementation of the QualiBD tool consisted of two phases: (i) modelling and code generation; and (ii) graphical editor definition. For the former, we used the Eclipse Modelling Framework (EMF), a modeling framework for building tools and applications based on a structured data model [29]. For the graphical modelling definition, we used Sirius [30], an Eclipse project that enables the creation of graphical modelling workbenches by leveraging the Eclipse Modelling technologies. Figure 6.5 depicts the frameworks used in this chapter and shows how they interact with one another.

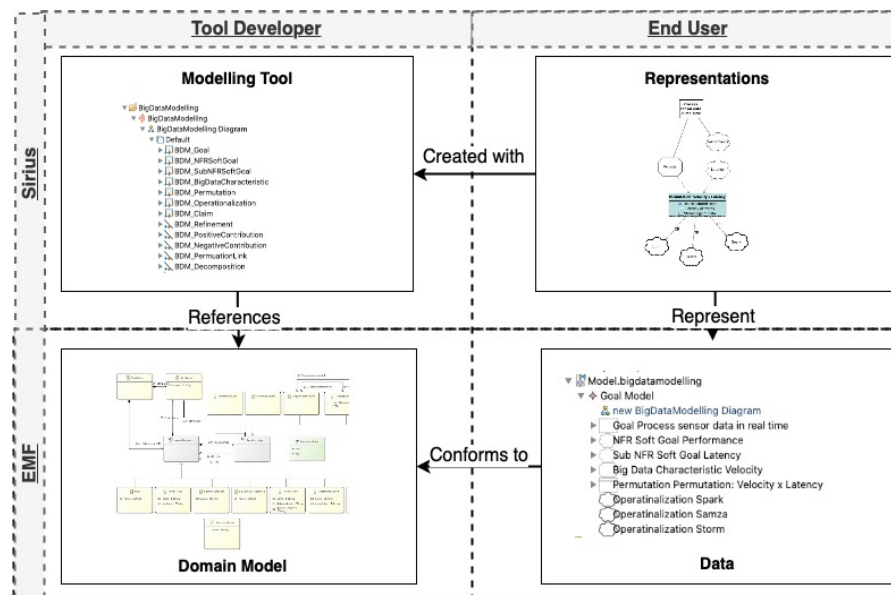


Figure 6.5: Overview of the frameworks used in the prototype tool creation

With reference to Figure 6.5: On the EMF side, we define the domain model (*lower left quadrant of the figure*) and create a concrete instance of that model that is dynamically interpreted using the runtime system within the Eclipse IDE environment (*lower right quadrant of the figure*). On the Sirius side, we design the modelling tool (*top left quadrant of the figure*) by defining all modelling elements, behaviour attributes, java services, validation expressions, navigation tools, and the graphical attributes of the model. The modelling tool references the domain model defined in EMF. The Graphical representation (*top right quadrant of the figure*) of the model is created using the defined modelling tool. The graphical representation is the modelled quality requirement in the QualiBD tool. It represents the data (*lower right quadrant of the figure*) that is the concrete instance of the defined domain model. Finally, the data conforms with the domain model defined in EMF.

The graphical user interface of the QualiBD tool is depicted in Figure 6.6. Further details on the implementation of the QualiBD tool can be found in Appendix D.

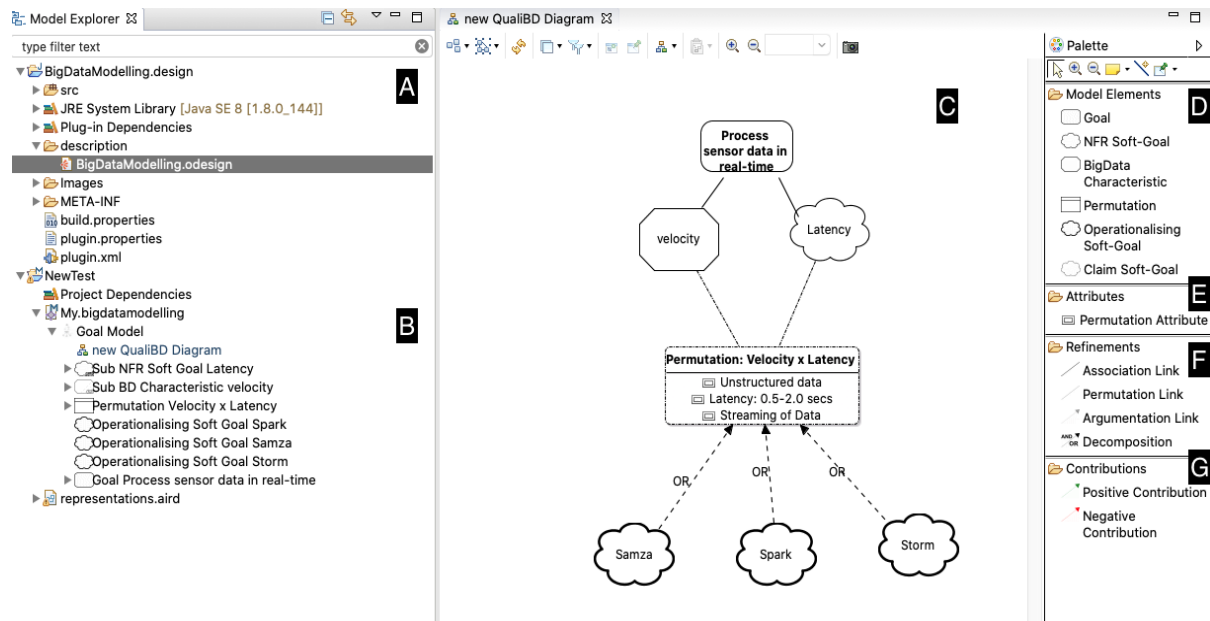


Figure 6.6: QualiBD Tool Graphical User-Interface

This figure depicts: A) Sirius Design project where the nodes, tools, and behaviour attributes of the modelling tool are defined; B) Eclipse project that creates a concrete instance of the defined Ecore domain-model; C) Tool canvas where models can be created, edited and deleted; D) Portion of the palette tool that allows end-users to create instances of model elements; E) Portion of the palette tool that allows end-users to add permutation attributes to permutation containers; F) and G) Portions of the palette tool that allow end-users to define the types of refinements (relations) supported by the QualiBD tool.

6.5 Validation Case Studies

In [31], Shaw describes several types of validation in software engineering research: (a) by analysis; (b) by experience; (c) by example; (d) by evaluation; (e) by persuasion; and (f) by blatant assertion. Shaw also explains that the type of validation needs to be appropriate for the type of research contribution. For instance, validation by experience would be suitable for research results that have been used in practice by someone other than researcher. Likewise, for the specification approach and modelling tool that we describe in this chapter, one appropriate validation procedure is by example which aims to exemplify the use of the proposed approach in a “slice of life” example based on real system projects [31]. We then used fragments of scenarios and requirements adapted from real-world Big Data applications development projects to demonstrate the feasibility of the QualiBD approach and described in the subsequent subsections.

6.5.1 Case 1

Consider the following short scenario adapted from [32]: *“Project A aims to develop and deliver a complete high-performant stack of technologies addressing the emerging needs of data operations and applications. The stack must be based on an infrastructure management system that drives decisions according to data aspects thus being fully scalable, runtime adaptable and performant for Big Data operations and data-intensive applications. The distributed storage of the platform must scale according to the usage in terms of volumes of data. The systems shall be able to store structured and non-structured data captured from different data sources...”*

Based on the scenario, one derives the requirement description and its associated Goal, Big Data characteristic (s), and Quality Attribute (s). Based on the derived information, one defines the possible permutations of Big Data characteristic (s) and Quality Attribute (s) and their corresponding Permutation Attributes. Then, solution alternatives (Operationalising Soft-goals) and their corresponding rationales (Claim Soft-goals) are identified as depicted in Figure 6.7. The logged requirement information would then be used as an input to model the requirement using the QualiBD Tool, as depicted in Figure 6.8.

ID: 001		Checklist Items	Status
Requirement Description: The distributed storage of the platform must scale according to the usage in terms of volumes of data. The systems shall be able to store structured and non-structured information captured from different data sources.		Goal	
Goal: Storage must scale according to the volume of data		Is the goal clearly stated in the requirement template?	<input checked="" type="checkbox"/>
Big Data Characteristic (s): Volume		Is the identified goal related to the analysed requirement description?	<input checked="" type="checkbox"/>
NFR Soft-goal (s): Scalability		Big Data Characteristics and Quality Attributes	
Permutation (s): Volume x Scalability		Are all the identified Big Data characteristics stated in the requirement template?	<input checked="" type="checkbox"/>
Permutation Attributes: Unstructured data, Structured data		Are all the identified quality attributes stated in the requirement template?	<input checked="" type="checkbox"/>
Operationalising Soft-goals (Solution Alternatives)	Claim Soft-goals (Rationales)	Are all the identified Big Data characteristic and quality attributes related to the state requirement description?	<input checked="" type="checkbox"/>
#1 Cassandra	Supports massive amounts of structured and unstructured data. Scale with minimal increase of administrative work (Linear Scalability, horizontally and vertically). High reliability. Easy to use. Supports integration with Hadoop ecosystem. Data consistency can be a problem with Cassandra.	Permutations	
#2 MongoDB	Offers high insert rates. Supports structured and unstructured data. Easier to update the data. Requires learning how to use the syntax. Supports integration with Hadoop ecosystem. Not built for transactional data. Best suit real-time analytics. Scales horizontally by adding new machines.	Are all the possible permutations stated in the requirement template?	<input checked="" type="checkbox"/>
#3 HBase	Supports linear and modular scalability. Supports integration with Hadoop ecosystem as it runs on top of the Hadoop distributed file system. Offers high reliability and schema flexibility.	Are all the identified permutations between one or more Big Data characteristics and one or more quality attributes?	<input checked="" type="checkbox"/>
		Permutation Attributes	
		Are all the identified permutation attributes stated in the requirements template?	<input checked="" type="checkbox"/>
		Are the identified permutation attributes related to the identified permutation (s)?	<input checked="" type="checkbox"/>
		Solutions	
		Are all the identified solution alternatives stated in the requirements template?	<input checked="" type="checkbox"/>
		Are the rationale to the identified solution alternative clearly stated in the requirements template?	<input checked="" type="checkbox"/>

Figure 6.7: Big Data quality requirement encoded using the QualiBD Approach and its corresponding checklist document (Case 1).

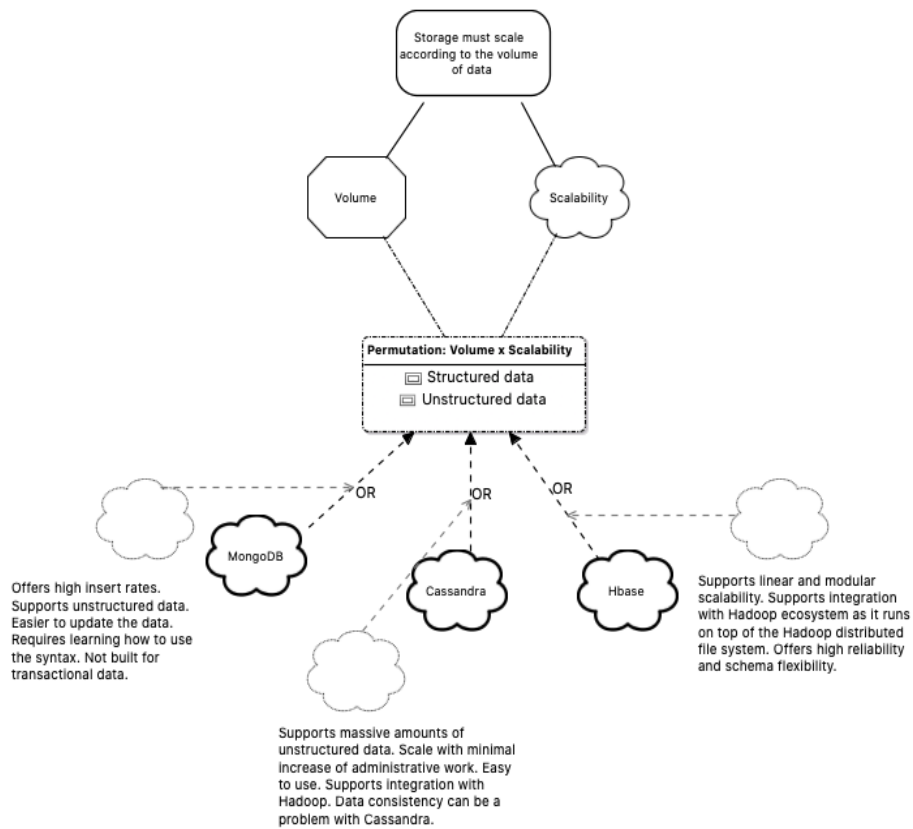


Figure 6.8: Big Data quality requirement modelled using the QualiBD Tool (Case 1).

6.5.2 Case 2

Consider the following scenario adapted from [33]:

“Project B aims to develop technology that supports the acquisition of user consent that caters for privacy-aware, security workflows to provide a dashboard with feedback and control features. The idea is to allow citizens and organisations to share more data, while guaranteeing data protection compliance, thus enabling both trust and creation of valuable new insights from shared data. The solution must be performant and scalable, this means, it must be capable of handling a vast amount of data, while keeping response times within a reasonable time range. Acceptable response time would be between 0.5 and 1 seconds. Additionally, mechanisms will be implemented to limit the amount of data displayed. This contributes to the usability of the dashboard, and indirectly, to the performance of the solution in terms of response time. The system must also guarantee a high level of confidentiality (security) since it is used to access its users sensitive personal data...”

From this scenario, one derives the requirements descriptions and their associated Goal (s), Big Data characteristic (s), and Quality Attribute (s). Based on the derived information, one defines the possible permutations of Big Data characteristic (s) and Quality Attribute (s) and their corresponding Permutation Attributes. Then, solution alternatives (Operationalising Soft-goals) and their corresponding rationales (Claim Soft-goals) are identified.

In the context of the aforementioned scenario, several requirements could be identified, from which two are described in this section. The first quality requirement (depicted in Figures 6.9 for the specification in natural language and 6.11 for the modelled requirement) is characterised by the following attributes: response time, usability and volume of unstructured data. The second quality requirement (depicted in Figures 6.10 for the specification in natural language and 6.12 for the modelled requirement) is characterised by the following attributes: confidentiality and volume of unstructured data.

Further analysis performed on the described scenario led us to the identification of three different types permutations for the first quality requirements (Volume x Response Time [application]; Volume x Response Time [storage]; and Volume x Usability [application]) and one permutation (Volume x Confidentiality) for the second quality requirement.

ID: 001		Checklist Items		Status
Requirement Description: The solution must be capable of handling a vast amount of data, while keeping response times within a reasonable time range.		Goal		
Goal: Support storage of large amounts of unstructured data while keeping application's low response time and certain level of usability		Is the goal clearly stated in the requirement template?		<input checked="" type="checkbox"/>
Big Data Characteristic (s): Volume		Is the identified goal related to the analysed requirement description?		<input checked="" type="checkbox"/>
NFR Soft-goal (s): Response Time (for storage and application); Usability		Big Data Characteristics and Quality Attributes		
Permutation (s): Volume x Response Time; Volume x Usability		Are all the identified Big Data characteristics stated in the requirement template?		<input checked="" type="checkbox"/>
Permutation Attributes: Unstructured data, response time between 0.5-1.0 seconds, user-friendly presentation of stored data.		Are all the identified quality attributes stated in the requirement template?		<input checked="" type="checkbox"/>
Operationalising Soft-goals (Solution Alternatives)		Permutations		
Claim Soft-goals (Rationales)		Are all the possible permutations stated in the requirement template?		<input checked="" type="checkbox"/>
#1 Cassandra (storage)	Supports massive amounts of unstructured data. Scale with minimal increase of administrative work. Easy to use.	Are all the identified permutations between one or more Big Data characteristics and one or more quality attributes?		<input checked="" type="checkbox"/>
#2 MongoDB (storage)	Offers high insert rates. Supports unstructured data. Easier to update the data. Requires learning how to use the syntax. Not built for transactional data.	Permutation Attributes		
#3 Asynchronous execution for applications layer (application)	The use of asynchronous execution environment would allow of techniques like lazy loading to optimize response times on a fine-grained level.	Are all the identified permutation attributes stated in the requirements template?		<input checked="" type="checkbox"/>
#4 Display limited amount of data (application usability)	The display of limited amount of data would help achieving a user-friendly presentation of the data by the dashboard.	Are the identified permutation attributes related to the identified permutation (s)?		<input checked="" type="checkbox"/>
		Solutions		
		Are all the identified solution alternatives stated in the requirements template?		<input checked="" type="checkbox"/>
		Are the rationale to the identified solution alternative clearly stated in the requirements template?		<input checked="" type="checkbox"/>

Figure 6.9: Big Data quality requirement encoded using the QualiBD Approach and its corresponding checklist document (Case 2 -Requirement 1)

ID: 002		Checklist Items		Status
Requirement Description: The solution must be capable of handling a vast amount of unstructured data. It must also guarantee a high level of confidentiality (security) since it is used to access users' sensitive personal data.		Goal		
Goal: Store large amounts of unstructured data while keeping high level of confidentiality.		Is the goal clearly stated in the requirement template?		<input checked="" type="checkbox"/>
Big Data Characteristic (s): Volume		Is the identified goal related to the analysed requirement description?		<input checked="" type="checkbox"/>
NFR Soft-goal (s): Confidentiality		Big Data Characteristics and Quality Attributes		
Permutation (s): Volume x Confidentiality		Are all the identified Big Data characteristics stated in the requirement template?		<input checked="" type="checkbox"/>
Permutation Attributes: Unstructured data, data at rest, data in motion.		Are all the identified quality attributes stated in the requirement template?		<input checked="" type="checkbox"/>
Operationalising Soft-goals (Solution Alternatives)		Are all the identified Big Data characteristic and quality attributes related to the state requirement description?		<input checked="" type="checkbox"/>
Claim Soft-goals (Rationales)		Permutations		
#1 In-transit encryption: SSL, SSH, and AES.	Use state-of-the-art encryption techniques when transmitting chunks of data between the data center and application. SSL is for securing internet connections between websites. SSL uses asymmetric cryptography to initiate the communication. SSH is for running commands via remote access. SSH uses symmetric encryption, asymmetric encryption and hashing in order to secure transmission of information. AES supports encryption of data (structured and unstructured) at rest and in-transit.	Are all the possible permutations stated in the requirement template?		<input checked="" type="checkbox"/>
#2 Delete data after session	Data retrieved by subjects could be deleted after every session, thus, contributing for the confidentiality of the processed and presented information.	Are all the identified permutations between one or more Big Data characteristics and one or more quality attributes?		<input checked="" type="checkbox"/>
#3 Deploy the dashboard within the controller's domain	The highest degree of security can be achieved by deploying the dashboard within the controller's domain, this way, the data subject's personal data remains in its entirety within the controller's infrastructure.	Permutation Attributes		
#4 Encrypt data retrieved by data subjects at rest: FFE and VTE.	FFE (File and Folder Encryption) and VTE (VTE - Vormetric Transparent Encryption). Both Support encryption of unstructured data at rest. VTE offers transparent data protection. It is scalable and easy to use. Meets most common compliance standards	Are all the identified permutation attributes stated in the requirements template?		<input checked="" type="checkbox"/>
		Are the identified permutation attributes related to the identified permutation (s)?		<input checked="" type="checkbox"/>
		Solutions		
		Are all the identified solution alternatives stated in the requirements template?		<input checked="" type="checkbox"/>
		Are the rationale to the identified solution alternative clearly stated in the requirements template?		<input checked="" type="checkbox"/>

Figure 6.10: Big Data quality requirement encoded using the QualiBD Approach and its corresponding checklist document (Requirement 2 - Case 2).

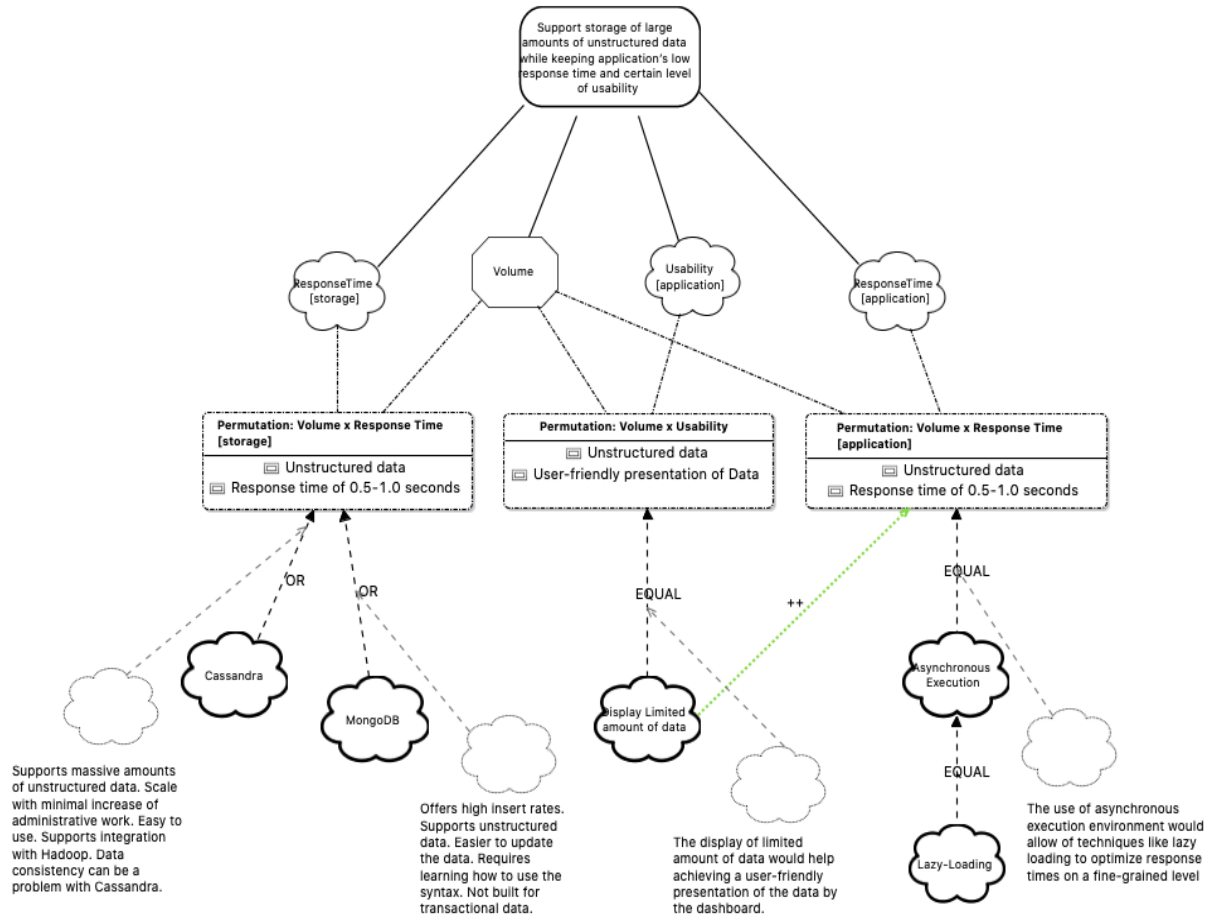


Figure 6.11: Big Data quality requirement modelled using the QualiBD Tool (Case 2 - Requirement 1).

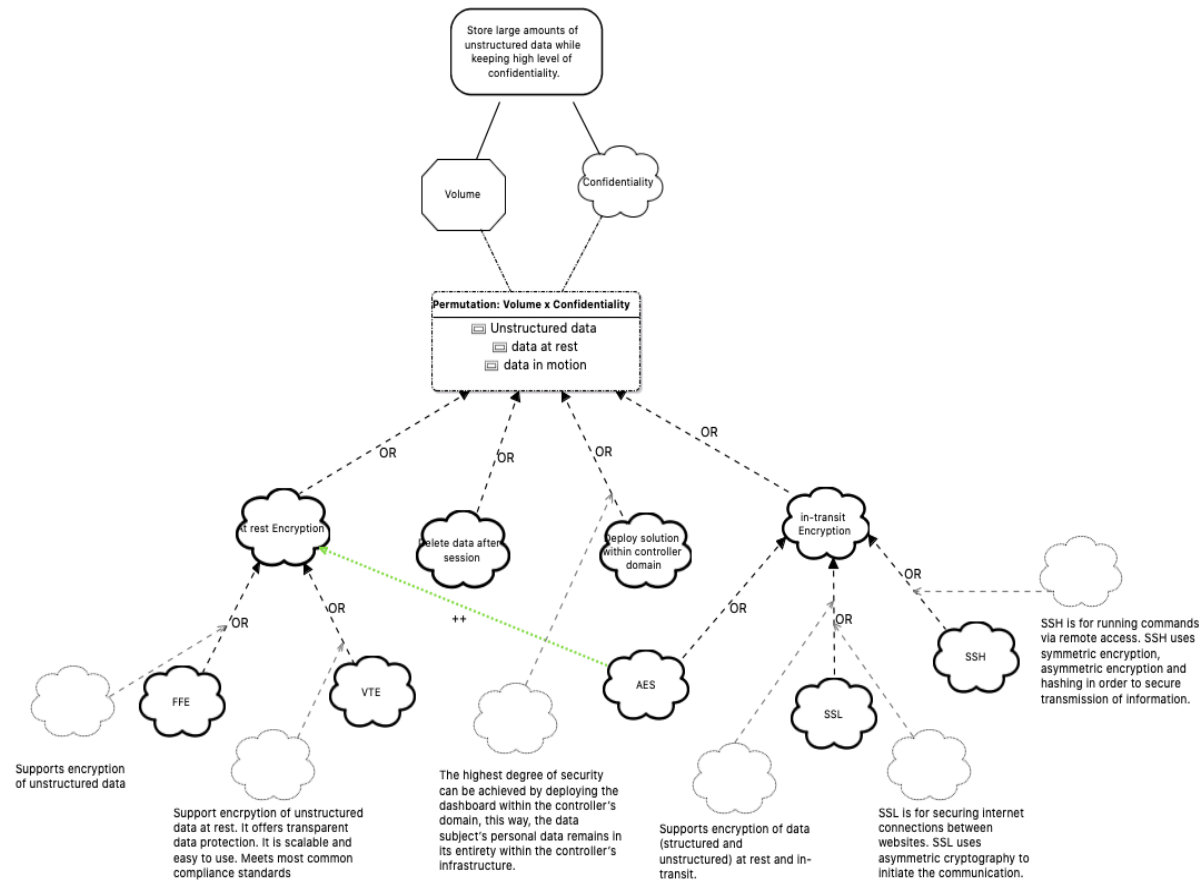


Figure 6.12: Big Data quality requirement modelled using the QualiBD Tool (Case 2 - Requirement 2).

6.6 Summary

The basic premise of this emerging work is that traditional software applications can leverage Big Data to enhance user experience with the systems responses. Currently, there is dearth of work done in such hybrid systems. This chapter attempts to address this issue and focuses on the what and the how of requirements for such applications. We described the QualiBD approach (see Section 6.3) for modelling quality requirements for Big Data software applications. The proposed approach was built upon the concepts of the Softgoal interdependency (SIG) graph of the NFR framework [25]. This allows us to bring existing theory and analysis techniques to the domain of RE involving Big Data applications, thus, adding scientific rigour to the approach.

The QualiBD approach is composed of a systematic process, checklist, requirements information logging template, a requirements language, and a modelling tool. It is organised into two phases: (i) pre-modelling (one focuses on the identification and specification (in natural language) of systems quality requirements), and (ii) modelling (one focuses on modelling the encoded quality requirement from the pre-modelling phase).

To determine the feasibility of the proposed approach as a proof of concept - we have used a set of systems scenarios descriptions extracted from real-world Big Data applications projects structured as two case studies (see Section 4). Our feasibility analysis demonstrates that it is possible to model Big Data quality requirements that integrate both Big Data characteristics and traditional systems quality attributes, thus, aiding in more complete requirements specifications.

Bibliography

- [1] B. Nuseibeh and S. Easterbrook, “Requirements engineering: A roadmap,” in *Proceedings of the Conference on The Future of Software Engineering*, ser. ICSE '00. New York, NY, USA: ACM, 2000, pp. 35–46.
- [2] R. S. Pressman, *Software engineering: a practitioners approach*, 2001.
- [3] A. S. et. al., “High-availability monitoring and big data: Using java clustering and caching technologies,” in *Proceedings of the ICALEPCS*, 2013.
- [4] N. H. Madhavji, A. Miranskyy, and K. Kontogiannis, “Big Picture of Big Data Software Engineering: With Example Research Challenges,” *Proceedings - 1st International Workshop on Big Data Software Engineering, BIGDSE 2015*, pp. 11–14, 2015.
- [5] N. Sawant and H. Shah, *Big Data Application Architecture Q&A: A Problem - Solution Approach*, 1st ed. Berkely, CA, USA: Apress, 2013.
- [6] A. Kadadi, R. Agrawal, C. Nyamful, and R. Atiq, “Challenges of data integration and interoperability in big data,” in *2014 IEEE International Conference on Big Data (Big Data)*, Oct 2014, pp. 38–40.
- [7] H. Kupwade Patil and R. Seshadri, “Big data security and privacy issues in healthcare,” in *2014 IEEE International Congress on Big Data*, June 2014, pp. 762–765.
- [8] M. Jensen, “Challenges of privacy protection in big data analytics,” in *2013 IEEE International Congress on Big Data*, June 2013, pp. 235–238.
- [9] D. Arruda and N. H. Madhavji, “State of requirements engineering research in the context of big data applications,” in *Requirements Engineering: Foundation for Software Quality*, E. Kamsties, J. Horkoff, and F. Dalpiaz, Eds. Cham: Springer International Publishing, 2018, pp. 307–323.
- [10] C. E. Otero and A. Peter, “Research directions for engineering big data analytics software,” *IEEE Intelligent Systems*, vol. 30, no. 1, pp. 13–19, 2015.
- [11] I. Noorwali, D. Arruda, and N. H. Madhavji, “Understanding quality requirements in the context of big data systems,” *Proceedings of the 2nd International Workshop on BIG Data Software Engineering - BIGDSE '16*, pp. 76–79, 2016.
- [12] M. Broy, “Requirements engineering as a key to holistic software quality,” in *Computer and Information Sciences – ISCIS 2006*, A. Levi, E. Savaş, H. Yenigün, S. Balcısoy, and Y. Saygın, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 24–34.
- [13] P. A. Laplante, *Requirements Engineering for Software and Systems*, 2nd ed. Boston, MA, USA: Auerbach Publications, 2013.

- [14] J. Al-Jaroodi and N. Mohamed, "Characteristics and requirements of big data analytics applications," in *2016 IEEE 2nd International Conference on Collaboration and Internet Computing (CIC)*, Nov 2016, pp. 426–432.
- [15] H.-M. Chen, R. Kazman, S. Haziyevev, and O. Hrytsay, "Big Data System Development: An Embedded Case Study with a Global Outsourcing Firm," *2015 IEEE/ACM 1st International Workshop on Big Data Software Engineering*, pp. 44–50, 2015.
- [16] 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*, May 2015, pp. 19–25.
- [17] H. H. Altarturi, K. Ng, M. I. H. Ninggal, A. S. A. Nazri, and A. A. A. Ghani, "A requirement engineering model for big data software," in *2017 IEEE Conference on Big Data and Analytics (ICBDA)*, Nov 2017, pp. 111–117.
- [18] P. Wang, K. Tao, C. Gao, X. Ning, S. Gu, and B. Deng, "Eliciting big data requirement from big data itself: A task-directed approach," in *2017 6th International Workshop on Software Mining (SoftwareMining)*, Nov 2017, pp. 17–23.
- [19] H. Wang and H. Zhang, "User requirements based service identification for big data," in *2017 IEEE International Conference on Web Services (ICWS)*, June 2017, pp. 800–807.
- [20] D. Arruda and N. H. Madhavji, "Towards a Requirements Engineering Artefact Model in the context of Big Data Software Development Projects," *Proceedings of the IEEE International Conference on Big Data*, pp. 2232–2237, 2017.
- [21] D. Arruda, N. H. Madhavji, and I. Noorwali, "A Validation Study of a Requirements Engineering Artefact Model for Big Data Software Development Projects," in *International Conference on Software Technologies (ICSOFT)*, July 2019, pp. 106–116.
- [22] V. Sachdeva and L. Chung, "Handling non-functional requirements for big data and IOT projects in Scrum," *Proceedings of the 7th International Conference Confluence 2017 on Cloud Computing, Data Science and Engineering*, pp. 216–221, 2017.
- [23] H. Eridaputra, B. Hendradjaya, and W. Danar Sunindyo, "Modeling the requirements for big data application using goal oriented approach," *2014 International Conference on Data and Software Engineering (ICODSE)*, pp. 1–6, 2014.
- [24] D. N. Jutla, P. Bodorik, and S. Ali, "Engineering Privacy for Big Data Apps with the Unified Modeling Language," *2013 IEEE International Congress on Big Data*, pp. 38–45, 2013.
- [25] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, *The NFR Framework in Action*. Boston, MA: Springer US, 2000, pp. 15–45.
- [26] S. Anwer and N. Ikram, "Goal oriented requirement engineering: A critical study of techniques," in *2006 13th Asia Pacific Software Engineering Conference (APSEC'06)*, Dec 2006, pp. 121–130.
- [27] H. Suleiman, A. Adepetu, E. Arnautovic, and D. Svetinovic, "Comprehensive integrated checklists for requirements engineering and software project management," in *2013 International Conference on Information Science and Applications (ICISA)*, June 2013, pp. 1–4.
- [28] C. Wiecha and S. Boies, "Generating user interfaces: Principles and use of it style rules," in *Proceedings of the 3rd Annual ACM SIGGRAPH Symposium on User Interface Software and Technology*, ser. UIST '90. New York, NY, USA: ACM, 1990, pp. 21–30. [Online]. Available: <http://doi.acm.org/10.1145/97924.97927>

- [29] D. Steinberg, F. Budinsky, and E. Merks, *EMF: Eclipse Modeling Framework*, ser. Eclipse (Addison-Wesley). Addison-Wesley, 2009.
- [30] (2019) Eclipse sirius official documentation. [Online]. Available: <https://www.eclipse.org/sirius/doc/>
- [31] M. Shaw, “Writing Good Software Engineering Research Papers,” in *International conference on software engineering (ICSE’03)*, vol. 6, 2003.
- [32] Big Data Stack Project. (2018) D2.1 state of the art and requirements analysis. [Online]. Available: <https://bigdatastack.eu/deliverables>
- [33] Big Data Special Project. (2017) Transparency dashboard and control panel - technical requirements v1. [Online]. Available: <https://www.specialprivacy.eu/publications/public-deliverables>

Chapter 7

Implications and Discussion

This chapter describes the implications and provides a discussion of the research reported in this thesis. In Section 7.1, we describe the implications of the results of the chapters of this thesis. Then, in Section 7.2, we discuss important complementary points concerning some of the research contributions.

7.1 Implications

The research presented in this thesis has various implications for industrial practice, academic research, and tool support. We briefly explain them as follows:

Industrial practice

- The proposed Big Data Artefact Model (BD-REAM) can aid in the: (i) definition of project-specific RE processes; (ii) requirements elicitation; (iii) architecture design, serving as template for executive presentations; (iv) specification, validation and testing of Big Data software applications; and (v) organisation of RE projects by providing a well-defined structure of RE artefacts and relationships.
- The QualiBD approach would enable practitioners to decisively construct quality re-

quirements using Big Data characteristics.

- The QualiBD tool would help in the specification of Big Data quality requirements through modelling which, in turn, may aid in creating quality Big Data software applications.
- The resultant modelled quality requirements would allow practitioners to easily visualise the identified solution alternatives to those requirements, that in turn, would aid in the decision making process of the project and architectural design.

Academic Research

- Researchers can explore the identified RE challenges in creating and evolving Big Data software applications with the aim to derive new research results addressing these challenges.
- The resultant Big Data RE Artefact model - with detailed elements and inter-relationships - is new knowledge that adds significantly to the current RE knowledge base involving the development of Big Data software applications.
- The proposed QualiBD Approach is novel and unique, thus, should also add to RE theory.
- Researchers can further validate the proposed Big Data RE Artefact model by performing further empirical studies in industrial settings (and in different application domains).

Tool Support

- The proposed Big Data RE Artefact model (with detailed elements and inter-relationships) can aid in creating traceability tools linking the artefacts.

7.2 Discussion

Some of the chapters reported in this thesis have solution proposals as their core contributions. In Chapter 5, for instance, we described a RE artefact model in the context of Big Data software development projects. In Chapter 6, we proposed a systematic approach and a tool for modelling quality requirements for Big Data applications. In this section, we assess these core contributions, focusing mainly on their limitations and potential for improvement and applicability to other project settings.

7.2.1 Big Data Requirements Engineering Artefact Model - BD-REAM

Applying the Artefact Model to Agile Projects

Because the BD-REAM is process agnostic and was defined in higher level of abstraction, the artefact model can be easily refactored to serve projects that operate on an agile environment. For instance, in agile projects, requirements are often expressed as user stories [1]. If you take a close look at the BD-REAM, we have several entities (referred as to artefacts) that represent the types of requirements within a Big Data project. Most types of requirements, regardless the software development methodology, will occur in any project (e.g., systems architectural requirements, infrastructure requirements, functional requirements, non-functional requirements, etc.), whether they are well documented or not. In other words, the BD-REAM has a strong focus on “what” and not on “how”. Thus, instead having an entity titled “Big Data Processing Requirements Specification”, we could have (in an agile environments), “Big Data Processing User Stories”. Instead of having “Business Case”, we would have “product vision statement”, and so on. Then, other agile-related artefacts (such as product backlog, spring backlog, product road-map) that have direct connection with RE would be added in the model, and new relationships would be created, thus, resulting in an agile version of the Big Data RE artefact model. This would be a much simpler process than defining the agile Big Data RE artefact model from scratch.

Defining Specific RE Process

In Chapter 5, Section 5.5.2, we stated that one of the advantages of having an RE artefact model defined for the project is that it enables the definition and tailoring of specific RE processes. For that purpose, one may add, for example, artefacts that represent completion status, decision gates, checklists, and activities [2, 3]. The idea is that, in early phases of the project or during the definition of the project, a draft of the artefact model is created and used to define the product life-cycle processes. As the project and product mature, the artefact model and defined processes are constantly updated. Berench [2] proposes a set of activities that could be followed in order to define RE specific processes from an artefact model as depicted in Figure 7.2.1 (for reference only).

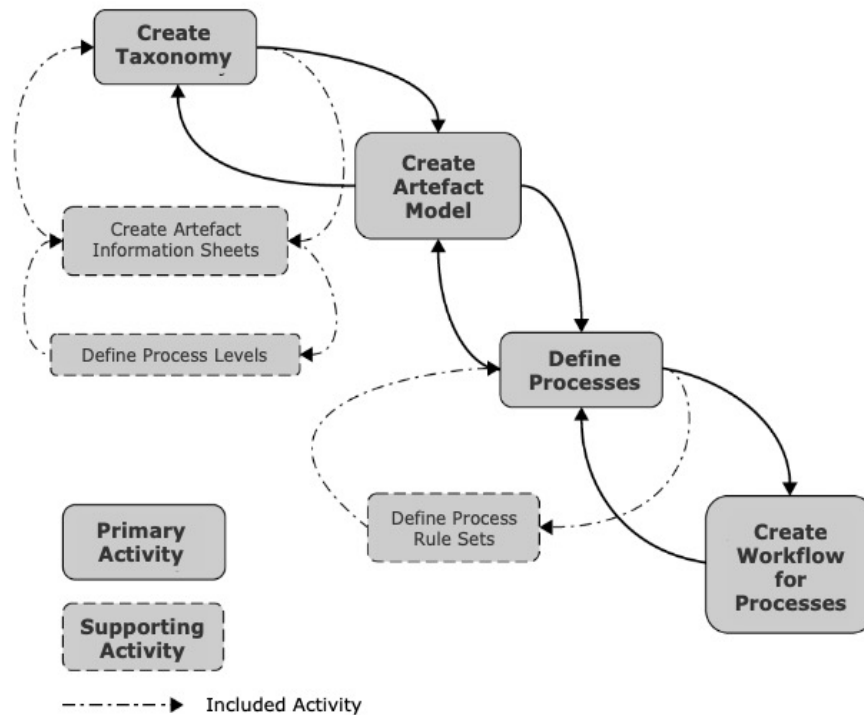


Figure 7.1: RE Activities for Process Creation (Adapted from [2])

Analysing the Cost for Adopting the Artefact Model in Industry

While we did not perform any cost analysis of the adoption of the artefact model in industry projects, we believe that the costs of adopting the model would be higher in cases where there is already an established project, given that some re-work would be necessary to reorganise the project artefacts according to the model's structure. In cases where the project is in the definition stage, the artefact model would be a valuable (and low cost) tool for guiding the requirements elicitation process and providing a well defined structure of artefacts that would aid in the overall organisation of the RE process within the project. Finally, Berenbach [2] states that, in his experience, while the upfront costs of creating a model may appear high, it was actually a very fast and cost-effective activity. Additionally, having project stakeholders think about downstream artefacts, quality gates, and approval checklists can result in significant improvements in the project [2].

Generalising the Artefact Model

One obvious limitation of artefact models is that one size does not fit all [2]. Although the model proposed in Chapter 5 has been internationally validated by ten practitioners working on ten different Big Data software projects, and further improved based on their feedback and analysis of project data from seven additional Big Data software projects, it is still not widely generalisable. For instance, an entity (artefact) depicted in the model, could be mandatory on a large project with a more traditional software development process, but optional on a medium-sized project, and not used at all on a small project [2]. However, given the validation results and model improvement process, we are optimistic that the artefact model proposed in this thesis would fit a wide range of projects with few modifications as described in Chapter 5, Section 5.5.2. Regardless, we recognise the need for additional empirical studies to further determine the generalisability of the model (see recommendations for future work in Chapter 8, Section 8.2).

7.2.2 The QualiBD Approach and Tool

Refining the Approach

The pre-modelling phase (see Chapter 6, Section 6.3.3) of the proposed approach currently provides a relatively high-level guidance as to how to ensure that the necessary requirements information to be used as input in the modelling phase is complete and accurate. At the moment, the approach handles this problem by providing a comprehensive checklist to be used after or while logging the required information (e.g., goals, data characteristics, quality attributes, etc.). In the future, automatic ways (e.g., natural language processing, machine learning techniques, etc.) for identifying the aforementioned information should be implemented along with more sophisticated techniques for checking the completeness and accuracy of the logged information.

Identifying Solution Alternatives

The identification of operationalising Soft-goals (used in the proposed QualiBD approach to represent solution alternatives to fulfil a given requirement) depend very much on the domain knowledge of the person performing the requirements specifications. Because of that, the quality and the extent to which the identified solution alternatives will in fact fulfil the requirement might differ depending on the person performing the activity. It is important to note that this is a “weak” point in most of the goal-oriented requirements engineering approaches. For instance, the NFR framework [4] (framework QualiBD approach is based upon), recognises that there is a gap between the NFR soft-goals and their associated solutions alternatives. Chung et al., [4] further explains that in order to bridge this gap, one must perform analysis and deal with a variety of factors (e.g, ambiguities, priorities, organisational needs, domain knowledge, and technologies, to name a few). In other words, comprehensive domain knowledge is key for the successful application of the approach. Nevertheless, in the future, we envision this process being supported by a semi-automatic tool (with an ontology library and automatic reasoning, for instance) that would help adding rigour to the approach as a whole.

Why identifying solution alternatives while modelling quality requirements?

“The complexity of a software system is determined partly by its functionality i. e., what the system does - and partly by global requirements on its development or operational costs, performance, reliability, maintainability, portability, robustness and the like. These non-functional requirements (or NFRs) play a critical role during system development, serving as selection criteria for choosing among myriads of alternative designs and ultimate implementations. Errors of omission or commission in laying down and taking properly into account such requirements are generally acknowledged to be among the most expensive and difficult to correct once a software system has been implemented”(Chung et al., [4]).

Stakeholders benefitting with the approach

As discussed in Chapter 6, the proposed approach supports the refinement of elements until an operationalisation level is achieved. Operationalisation level elements can be represented by several types of elements (representing design solutions, implementation ideas, patterns, etc.), and serve as selection criteria for choosing among myriads of alternative designs, technologies and ultimate implementations [4].

At a glance, the proposed approach can directly benefit two types of internal stakeholders in the project: software architects, and developers. For software architects, the resultant graph facilitates systems design through refinement lower levels (operationalising soft-goals) when defining design solutions, often related to technological requirements. Likewise, for developers, the resultant graph facilitates the implementation phase through refinement lower levels (operationalising soft-goals) when implementation elements are defined.

Using the Tool

Once the pre-modelling phase is completed and all required information is logged, one can use it to model the requirement through the QualiBD Tool. Currently, this process is done manually, which adds time to the process. In the future, we expect that the tool will have a feature to read structured requirements information documents (using the template provided

with our approach) to partially generate the requirements models, thus, speeding up the process and mitigating possible user induced errors.

7.3 Summary

In this section, we described the implications that the results reported in this thesis have on industrial practice, academic research, and tool support. Then, focusing on limitations and the possibility of improvement and applicability to other project settings, we assessed the solutions (e.g., the Big Data RE artefact model, and the QualiBD approach and tool) proposed in this thesis. Although there are some definite limitations, the results of the research presented in this thesis provide empirical evidence and help to form the foundation of RE involving Big Data applications currently not thoroughly explored in the scientific literature.

Bibliography

- [1] (2019) Agile requirements core practices. [Online]. Available: <http://www.agilemodeling.com/essays/agileRequirementsBestPractices.html>
- [2] B. Berenbach, D. Paulish, J. Kazmeier, and A. Rudorfer, *Soft. Systems Requirements Eng. In Practice*, 2009.
- [3] E. Geisberger, M. Broy, B. Berenbach, J. Kazmeier, D. Paulish, and A. Rudorfer, “Requirements Engineering Reference Model (REM),” Tech. Rep.
- [4] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos, *The NFR Framework in Action*. Boston, MA: Springer US, 2000, pp. 15–45.

Chapter 8

Conclusions and Future Work

In this section, we present the conclusions and future work of this thesis. Subsection 8.1 presents our conclusions drawn based on the reflections of the empirical studies reported in this thesis. Subsection 8.2 overviews directions for possible future work.

8.1 Conclusions

Practising Requirements Engineering (RE) is a complex and challenging task. It involves stakeholders with diverse backgrounds and levels of knowledge, different application and technology domains, it is expensive and error-prone, to name a few [1]. Recently, the emergence of software applications operating upon Big Data (the so-called Big Data applications) has introduced further complexities in the RE process. That is due to the fact that Big Data applications are complex solutions composed of dynamic components such as distributed computation nodes, networks, databases, middleware, and business intelligence layers [2], and in part, due to the lack of clarity in the RE literature and practices on how to treat Big Data and the “V” characteristics (e.g., volume, velocity, variety, veracity, etc.) in the development of Big Data applications.

As previously stated in this thesis, most of the focus in the field of Big Data software is on data analytics and the development of algorithms and techniques to process and extract value

from huge amounts of data [3]. In contrast, little research or industry practices focus on software applications and services that utilise the underlying Big Data to enhance the functionality and services provided to the end-users [2, 4]. In order to ameliorate the current situation, in this thesis, we investigated a number of issues (such as RE research, challenges, practices, domain models, and requirements specification approaches) in RE involving Big Data software applications.

In chapter 3, we reported on the results of Systematic Literature Review (SLR) conducted with the aim to setting the current baseline in RE research involving Big Data Applications. We ask three key questions represented by the following core points:

- (RQ1) activities in the RE process, types of requirements, and application domains;
- (RQ2) RE challenges;
- (RQ3) RE solutions.

The key findings relate to the following:

- (RQ1) RE activities, types of requirements (see Table 3.5), and application domains targeted by the current state of the RE research involving Big Data applications (see section 3.4.1)
- (RQ2) Eight RE challenges in creating/evolving Big Data software applications (see Section 3.4.2 and Table 3.8 for details);
- (RQ3) Eighteen RE solutions identified (e.g., models, algorithms, tools, and processes)(See Tables 3.6 and 3.7 for details).

The SLR results demonstrates that there has been little scientific research aimed at understanding the RE in the development of Big Data applications. An important observation, and conclusion, made is that, currently, there is not a significant amount of research addressing RE methods, tools, and processes for elicitation, negotiation, analysis, validation, prioritization and management of requirements in the context of Big Data application development projects. This, presents the scientific community with opportunities to conduct further research in this topic (see Sections 3.5 and 3.7 for details).

In chapter 4, we described an exploratory case study on a large-scale Big Data application development project in Oil&Gas domain within a non-profit organisation with the aim to understand the current RE practices and challenges in such projects. We defined four research questions represented by the following core points:

- (RQ1) Sources and Proportion of Big Data Requirements;
- (RQ2) RE practices and supporting tools for eliciting, documenting, analysing, and prioritising systems requirements;
- (RQ3) The role of Big Data Characteristics and Technologies;
- (RQ4) RE challenges in creating Big Data applications.

The key findings relate to the following:

- (RQ1) 40% of the system's requirements are considered Big Data-related from which 75% are identified from internal sources;
- (RQ2) 11 RE practices for elicitation, specification and modelling, analysis, and prioritisation of requirements;
- (RQ3) Big Data characteristics and technologies support the definition of system's architecture;
- (RQ4) Five challenges in eliciting, documenting, and analysing Big Data related requirements were identified.

The results of the reported case study demonstrates that there is a lack of RE supporting tools, RE patterns, and specific processes to support the engineering of Big Data applications. Despite this limitation, practitioners try to adapt existing SE tools and methods to the needs of such unique projects. However, this is not ideal. Thus, several research opportunities are envisaged and described in Section 8.2.

In chapter 5, we attempted to shed some light on different types of artefacts and inter-relationships involved in Big Data applications development projects, with particular focus on

Requirements Engineering. This type of model can be used as a reference for the design of project-specific processes [1, 5], software maintenance [1], and for supporting project decisions throughout the entire product life-cycle [5]. The investigation was centered around the following core points (CP):

- (CP1) Types of RE artefacts existing in Big Data applications development projects.
- (CP2) Inter-relationships that exist amongst the several types of RE artefacts in Big Data applications development projects.

The key findings relate to the following:

- (CP1) 43 different types of RE artefacts identified from which 18 are Big Data related.
- (CP2) Six different types of inter-relationships and several cardinality characteristics were identified.

Based on the analysed artefacts and their inter-relationships, we introduced a preliminary version of a Big Data Requirements Engineering Artefact Model - BDREAM (see Figure 5.2 for details). The described BDREAM was validated internationally by 10 practitioners from 10 different Big Data applications development projects in industry (see Subsection 5.4.1 for details). Following the validation study, and the analysis of data (artefacts) from Big Data applications projects in industry (see Table 5.10), we created an improved version of the BDREAM (see Figure 5.4 for details). The BDREAM depicts artefacts grouped into three groups [6]: (i) Business Needs artefacts; (ii) Requirements Specification artefacts; and (iii) Systems Specification artefacts (see Section 5.6 for details). Based on the validation results, we conclude that the proposed BDREAM captures the key RE artefacts and relationships of a Big Data applications development project, currently lacking in the scientific literature. The validation results also confirm consensus amongst the study participants regarding the usefulness and applicability of the model in practice (see Table 5.4, Subsection 5.5.2).

Finally, in chapter 6, we explored requirements specifications in the context of Big Data software applications. In particular, we investigated ways of specifying Big Data quality requirements that integrate Big Data characteristics (such as volume, velocity, variety, and ve-

racity) and systems' quality attributes (such as scalability, performance, security, privacy, etc) in the same requirement specification.

As a result of our investigation, we described an approach for specifying quality requirements for Big Data applications. The proposed approach is composed of a systematic process, requirements logging templates, checklists, Big Data goal-oriented requirements language, and a supporting tool. Our feasibility analysis (shown through the cases studies on three real-world Big Data software projects, presented in Section 6.5) demonstrates that it is possible to specify - through modelling - quality requirements that integrate both big data characteristics and traditional systems' quality attributes, aiding in more complete requirements specifications which, in turn, may assist in creating quality Big Data software applications.

8.2 Future Work

The empirical studies presented in this thesis provide important but preliminary knowledge on RE involving Big Data applications development projects. Given the exploratory nature of these studies, they opened up new avenues of scientific knowledge rather than confirming any previous hypothesis or theories. Thus, there are several opportunities for future work. In the following paragraphs, we describe these opportunities organised by the major contributions made in this thesis.

- *Empirical knowledge on RE practices in real-world Big Data Software Systems Projects*: it is important that additional empirical studies in industry are performed to obtain an improved understanding of the RE activities in the development of Big Data applications. Empirical studies would add significantly to the meagre knowledge base on RE involving Big Data applications, which can improve processes and technologies and uncover more facts that could lead to further research in this area.
- *The Big Data RE Artefact Model (BD-REAM)*: (i) enhancement of the model embracing new application domains, such as IoT (internet of things); (ii) empirical studies of the application of the model in Big Data projects to further assess the models adaptability

and generalisability; and (iii) cost analysis of adopting the artefact model in industry projects.

- *QualiBD Approach*: (i) perform empirical evaluations within different Big Data applications domain projects in order to evaluate the generalisability of the proposed approach; (ii) expand our feasibility analysis in order to determine whether the modelling representing the permutation among multiple Big Data characteristics and various quality attributes is viable; and (iii) define ways to semi-automate the steps involved in the proposed approach (e.g., transformation of textual requirement descriptions into model elements).
- *Big Data Requirements Modelling Language*: (i) formal definition of the requirements modelling language which, in turn, should help in the verification of the models generated using the QualiBD Tool.
- *QualiBD Tool*: (i) tool enhancement and use in practical projects; (ii) automatic generation of goal models from textual description of requirements; and (iii) scalability tests (controlled experiments) in order to evaluate the proposed requirements language with respect to the total number of goal elements supported by our modelling tool. [7]

Bibliography

- [1] B. Berenbach, D. Paulish, J. Kazmeier, and A. Rudorfer, *Soft. Systems Requirements Eng. In Practice*, 2009.
- [2] N. H. Madhavji, A. Miranskyy, and K. Kontogiannis, “Big Picture of Big Data Software Engineering: With Example Research Challenges,” *Proceedings - 1st International Workshop on Big Data Software Engineering, BIGDSE 2015*, pp. 11–14, 2015.
- [3] V. Dipti Kumar and P. Alencar, “Software engineering for big data projects: Domains, methodologies and gaps,” *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, pp. 2886–2895, 2016.
- [4] D. Arruda and N. H. Madhavji, “State of requirements engineering research in the context of big data applications,” in *Requirements Engineering: Foundation for Software Quality*, E. Kamsties, J. Horkoff, and F. Dalpiaz, Eds. Cham: Springer International Publishing, 2018, pp. 307–323.
- [5] B. Penzenstadler, D. M. Fernandez, and J. Eckhardt, “Understanding the impact of artefact-based RE - Design of a replication study,” *International Symposium on Empirical Software Engineering and Measurement*, pp. 267–270, 2013.
- [6] E. Geisberger, M. Broy, B. Berenbach, J. Kazmeier, D. Paulish, and A. Rudorfer, “Requirements Engineering Reference Model (REM),” Tech. Rep.
- [7] H. Suleiman, A. Adepetu, E. Arnautovic, and D. Svetinovic, “Comprehensive integrated checklists for requirements engineering and software project management,” in *2013 International Conference on Information Science and Applications (ICISA)*, June 2013, pp. 1–4.

Appendix A

Instrument for Data Collection: Case Study Questionnaire

In chapter 4, we reported on the results of an exploratory case study conducted on a large scale Big Data application development project within the *Oil&Gas* domain. In this appendix, we describe the instrument for data collection defined for this investigation. The semi-structured questionnaire (**presented in the next page**) is composed of 26 questions organised into (a) background, and (b) RE related questions. The later was designed according the the activities in the RE process (e.g., Elicitation, Specification and Modelling, Analysis, and Prioritisation).

Instrument for Data Collection

Created by Darlan Arruda

Reviewed by Nazim H. Madhavji

Last version date: Nov 12nd, 2019.

Background Questions

About the companies and projects

1. Number of employees in the company
2. What is the core business of the company?
3. What was the size of the project (in terms of team members)?
4. How is it distributed?
of developers:
of requirements analysts/business analysts:
of data scientists:
of sw architect:
of data engineers:
of testers:
5. What was your role in that project?

About the software development process

6. What development methodologies are used in the project (e.g., agile, spiral, waterfall, RUP, Iterative, prototype, mix of methodologies, etc.)?

Requirements Engineering Related Questions

Overall Requirements related Questions

7. What are the types of big data-related requirements (e.g., infrastructure/platform requirements, data source requirements, data analytics requirements, data processing, technological requirements, data requirements, etc.) involved in your project?
8. Typically, what proportion of all identified requirements are big data-related requirements? (answer to the best of your ability)
 1. 0-20%
 2. 21-40%
 3. 41-60%
 4. 61-80%
 5. 80%+

Requirements Elicitation related Questions

9. To what extent do you identify the big data-related requirements from external sources (answer to the best of your ability)?
 - a) Not at all (0%)
 - b) To a small extent (up to 25%)
 - c) To a moderate extent (from 26% to 50%)
 - d) To a great extent (from 51% to 75%)
 - e) To a very great extent (>75%)

10. To what extent are the big data-related requirements gathered from internally sources (answer to the best of your ability)?
 - a) Not at all (0%)
 - b) To a small extent (up to 25%)
 - c) To a moderate extent (from 26% to 50%)
 - d) To a great extent (from 51% to 75%)
 - e) To a very great extent (>75%)
11. Are big data technologies (e.g., tools for processing data, platforms, etc.) considered or identified at the stage of identifying systems requirements or are these decided upon later in the process?
12. Please indicate the % of requirements identified upfront in the RE process and the % of requirements identified downstream during design/coding/testing.
13. What, if any, are the challenges encountered in **identifying** big data-related requirements?

Requirements Specification related Questions

14. Do you specify (document) the software requirements? If so, do you also document Big data-related requirements in your project and, if so, please describe their format, standards followed (if any), support tools used, etc.?
15. With respect to non-functional requirements (i.e., those describing system qualities, e.g., performance, reliability, usability, etc.), do you document the characteristics of Big Data (e.g, velocity, volume, variety, etc.) along with system quality attributes in the same requirement description?
16. Please kindly give examples of such big data-related requirements.
17. What challenges, if any, are encountered in specifying (documenting) big data related requirements? Note; whereas Q13 is focused on “identifying”, here the focus is on “**specifying**” (or documenting) the requirements.

Requirements Modelling related Questions

18. Do you “model” the identified Big Data requirements (e.g., UML or other notations)? If so, please comment on the tools and modelling techniques used for this purpose.
19. What, if any, are the challenges encountered in modelling the big data-related requirements?

Requirements Analysis related Questions

20. What, if any, kind of analysis is done on the documented requirements?
21. Which methods or standard, if any, are followed for analysing the requirements?
22. What, if any, are the challenges encountered in analysing the big data related requirements?

Requirements Prioritisation related Questions

23. What factors are considered in prioritising big data-related requirements?
24. Which method, technique, tools, or standard, if any, are followed for prioritizing the requirements?
25. What, if any, are the challenges encountered in prioritizing big data related requirements?

Architecturally Significant Requirements

26. To what extent are Big Data requirements architecturally significant?

Appendix B

Instrument for Data Collection: Artefact Model Validation

This document depicts an Artefact-model (a model showing artefacts and interlinkages) in the field of Requirements Engineering (RE) in conjunction with Big Data software applications. Literature indicates that among the many uses of the artefact model are: (i) support in the definition of domain specific RE models, (ii) system life-cycle processes and, (iii) artefact centred processes.

- The depicted model was created from extensive analysis of the scientific literature and has been evaluated internationally by ten practitioners working in Big Data Software Projects. It is shown and explained in Section 2 of this document.
- The purpose of this document is to further validate this model with perspectives from the field of practice such that an improved model would be shared in the public domain for others to use in both research and practice.
- You are approached because of your background and expertise in the fields of RE, software development, Big Data, and related topics. Your input would help at this formative stage to create a foundationally strong model that both practitioners and researchers can depend upon.

This document is composed of the following sections:

- Section 1: Background Questions;
- Section 2: Overview of the Artefact model;
- Section 3: Technical Validation Questions;
- Section 4: General Validation Questions; and
- Section 5: Suggestions for Improvement.

Section 1: Background QuestionsName (optional): Email address (optional): Affiliation (optional):

Please note that we'd be glad to send you the final report for free if we have your contact details. The contact details, if provided, will NOT be used for any other purpose than to send you the final report. You need *not* provide us with any contact details but your input would be greatly appreciated.

1. Type of organisation you have worked in for 6 months or more (please choose one or more):

Industry

Governmental organization

Academic institution

Other (Please specify): **2. Key roles played in your career (please choose one or more):**

Manager (project/product/release/process/etc.)

Requirements Analyst/Engineer

Business Analyst

Developer

Architect

Quality assurance/engineer

Quality control (inspection/testing/internal audits/etc.)

Researcher

College/University Professor

Consultant

Customer relationship and product marketing roles

Other (Please, specify): **3. Number of years of work experience with Requirements Engineering (RE):**

None 1-4 years 5-10 years 11-15 years 16+ years

4. Number of years of work experience at senior levels in the organisation:

None 1-4 years 5-10 years 11-15 years 16+ years

5. Number of years of work experience in the Big Data field

None 1-2 years 3-5 years 5+ years

6. Please indicate your years of experience with the following RE projects:

	None	1-2 years	3-5 years	5+ years
RE in industry -- traditional (non-Big Data) applications development.				
RE in academia (research and/or teaching) -- traditional (non-Big Data) applications development.				
RE in industry – Big Data Analytics projects (e.g., in the development and use of tools for data analytics, machine learning algorithms, etc.).				
RE in academia (research and/or teaching) – Big Data Analytics projects (e.g., in the development and use of tools for data analytics, machine learning algorithms, etc.).				
RE in industry -- Big Data-centric applications development (NOT Data Analytics projects)				
RE in academia (research and/or teaching) -- Big Data applications development (NOT Data Analytics projects)				

7. In which application domain(s) do you work or have you worked? Please choose all that apply.

Healthcare

Biomedical Research

Government

Marketing

IT/Telecom

Astronomy and Physics

Environmental and Polar Science

Defense/Military

Commercial

Social Media

Retail

Tourism

Transport

Geospatial Data Processing/Geographic Information Systems

Manufacturing

Cyber Physical Systems

Agriculture

Banking and Financial Industry

Aviation Industry

National Security

Other (please indicate):

Section 2: Overview of the Artefact model

Extracted from:

D. Arruda and N. H. Madhavji, "Towards a Requirements Engineering Artefact Model in the Context of Big Data Software Development Projects: Research in Progress," 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 2017, pp. 2314-2319. doi: 10.1109/BigData.2017.8258185

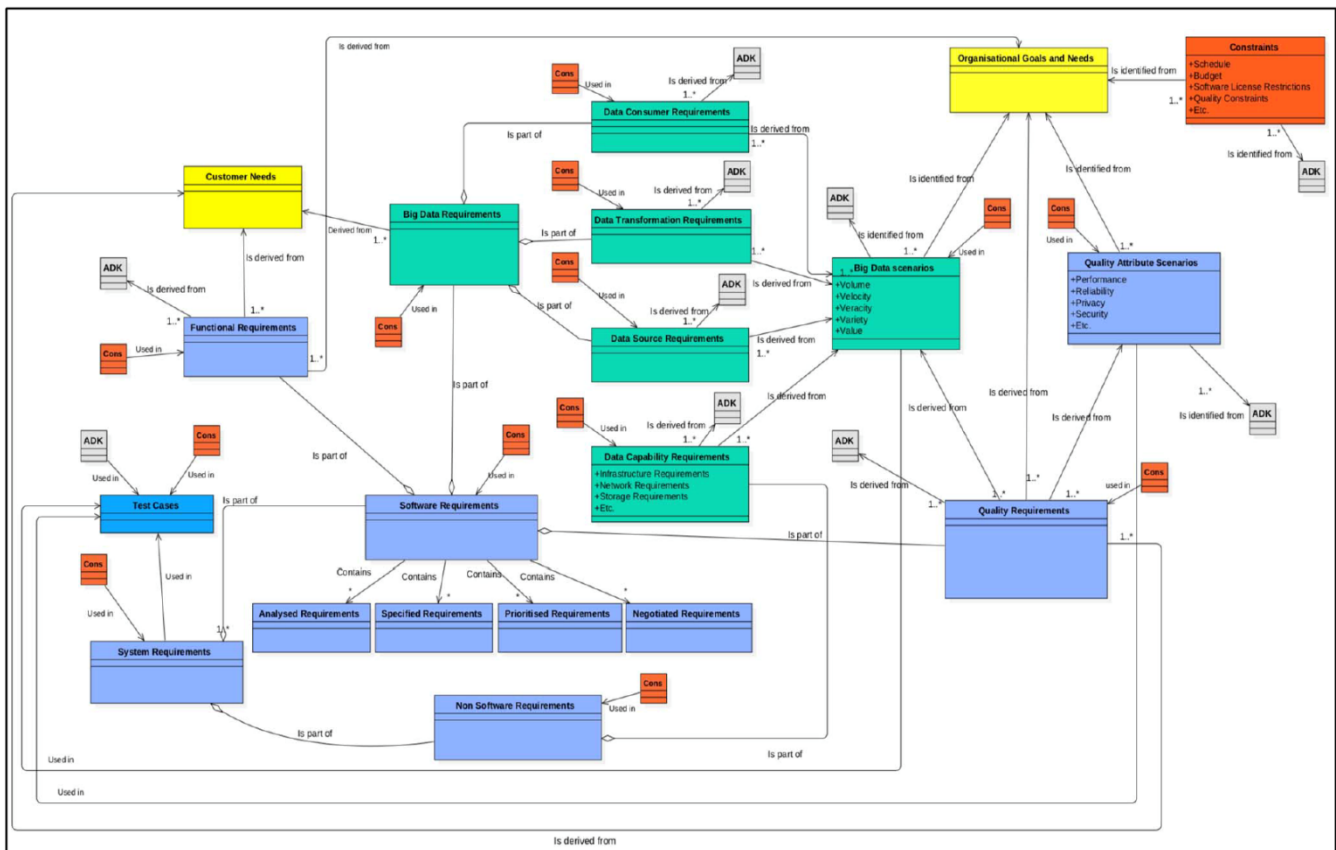
The RE artefact model depicted below, in the context of Big Data software development projects, comprises **21 elements and numerous relationships** – identified from the scientific literature. The following relationships are represented in the model:

- (i) **Is-derived-from** represents the relationship between the artefacts when from one artefact (e.g., Big Data scenarios) one or more artefacts can be derived and specified (e.g., quality requirements are derived from Big Data scenarios);
- (ii) **Is-identified-from** represents the relationships when from one artefact (e.g., organisational goals) one or more artefacts (e.g., Big Data Scenarios, Constraints and Concerns, etc.) are identified;
- (iii) **Is-part-of** relationship represents aggregation and it is illustrated when one or more artefacts are part of one or more major artefacts (e.g., functional requirement is part of software requirements);
- (iv) **Contains** relationship is used when one or more artefacts have or hold information from another artefact within (e.g., software requirements contain analysed requirements); and
- (v) **Used-in** relationship means that one artefact can be used to guide in the definition of other artefacts (e.g., project constraints are used in Big Data scenarios).

The following Big Data elements are represented in the model:

- **Big Data Scenarios:** They incorporate Big Data characteristics in their descriptions (e.g., volume, velocity, variety, veracity, etc.).

- **Data-Capability Requirements:** They deal with infrastructure issues such as: the need to support legacy and advanced software packages, legacy and advanced computing platforms, data storage and elastic data transmission, and hardware.
- **Data-Source Requirements:** They refer to the requirements that deal with different characteristics of data sources (e.g., data size, file formats, rate of growth, at rest or in motion, etc.) [
- **Data-Transformation Requirements:** They refer to the requirements that relate to data analytics, data fusion and data processing.
- **Data-Consumer Requirements:** They refer to the requirements that relate to the presentation of the processed results of Big Data to the users (e.g., processed results in text, table, visual, and other formats).



Section 3: Technical Validation Questions

8. To what extent do you agree that the schematic model in Section 2 reflects the type of RE artefacts in the development of Big Data applications in industry?

Strongly agree

Agree
Neither agree nor disagree
Disagree
Strongly disagree

Please give a short rationale for your opinion:

9. Do you think that there are any elements that are missing from the schematic model depicted in section 2?

Yes. Please list the missing elements and give a short rational for each element identified.

Rationale:

No.
No opinion.

10. To what extent do you agree that the names of the artefacts depicted in the model in Section 2 are appropriate?

Strongly agree
Agree
Neither agree nor disagree
Disagree
Strongly disagree

Please give a short rationale for your opinion:

11. To what extent do you agree that the labels of the relationships in the model depicted in Section 2 are appropriate?

- Strongly agree
- Agree
- Neither agree nor disagree
- Disagree
- Strongly disagree

Please give a short rationale for your opinion:

12. With reference to the artefact model in Section 2, to what extent do you agree that the elements in the model named: data-capability requirements, data-source requirements, data-transformation requirements and data-consumer requirements – represent the whole spectrum of the types of big data requirements?

- Strongly agree
- Agree
- Neither agree nor disagree
- Disagree
- Strongly disagree

Please give a short rationale for your opinion:

Section 4: General Validation Questions

13. To what extent do you agree that artefact model is useful for practice?

- Strongly agree
- Agree
- Neither agree nor disagree
- Disagree
- Strongly disagree

Please give a short rationale for your opinion:

14. For what purposes is the model considered useful?

- 1.
- 2.
- 3.
- 4.

15. To what extent do you agree that the artefact model is generic enough to be applicable to different types of Big Data projects, possibly with few modifications?

Strongly agree

Agree

Neither agree nor disagree

Disagree

Strongly disagree

Please give a short rationale for your opinion:

Improvement Suggestions

Please kindly provide any other recommendations below for improving of the artefact model.

Appendix C

QualiBD Tool: End-user Interface and Features

In this appendix, we describe the graphical user-interface and some of the features supported by the QualiBD Tool.

- The modelling tool facilitates graphical modelling of Big Data application requirements in the style of WYSIWYG (*what you see is what you get*) paradigm. This includes drag and drop features, and editing of entities and relationships in the model.
- Automatic labelling of a permutation containers (based on parent node labelling) simplifies model creation.
- Rudimentary analysis capabilities such as raising caution when there are: (i) missing or duplicate relations in the model; (ii) missing labels; and (iii) permutation containers without permutation attributes.

The following Figures depicts the graphical user-interface and the validation features of the QualiBD Tool.

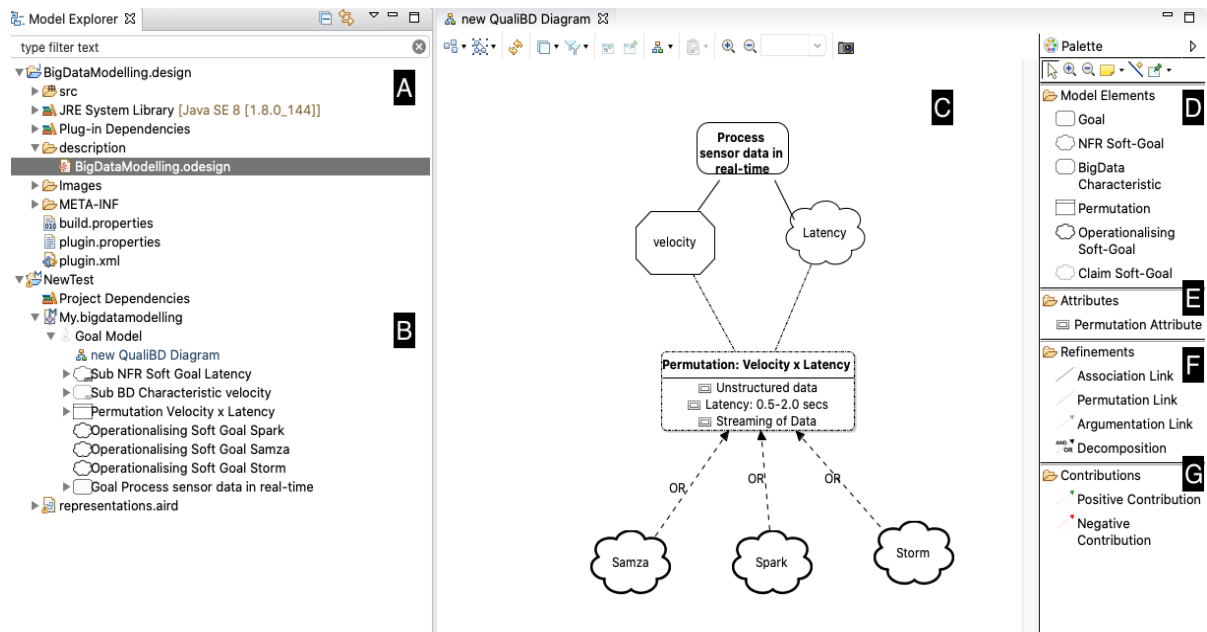


Figure C.1: Graphical User-Interface of the QualiBD Tool

Depicted in this Figure: A) Design project where the nodes, tools, and behaviour attributes of the modelling tool are defined; B) Eclipse project that creates a concrete instance (data) of the defined domain-model; C) Tool canvas where models can be created, edited and deleted; D) Portion of the palette tool that enables end-users to create instances of model elements; E) Portion of the palette tool that enables end-users to add permutation attributes to permutation containers; F) and G) Portions of the palette tool that enables end-users to define the types of refinements (relations) supported by the QualiBD tool.

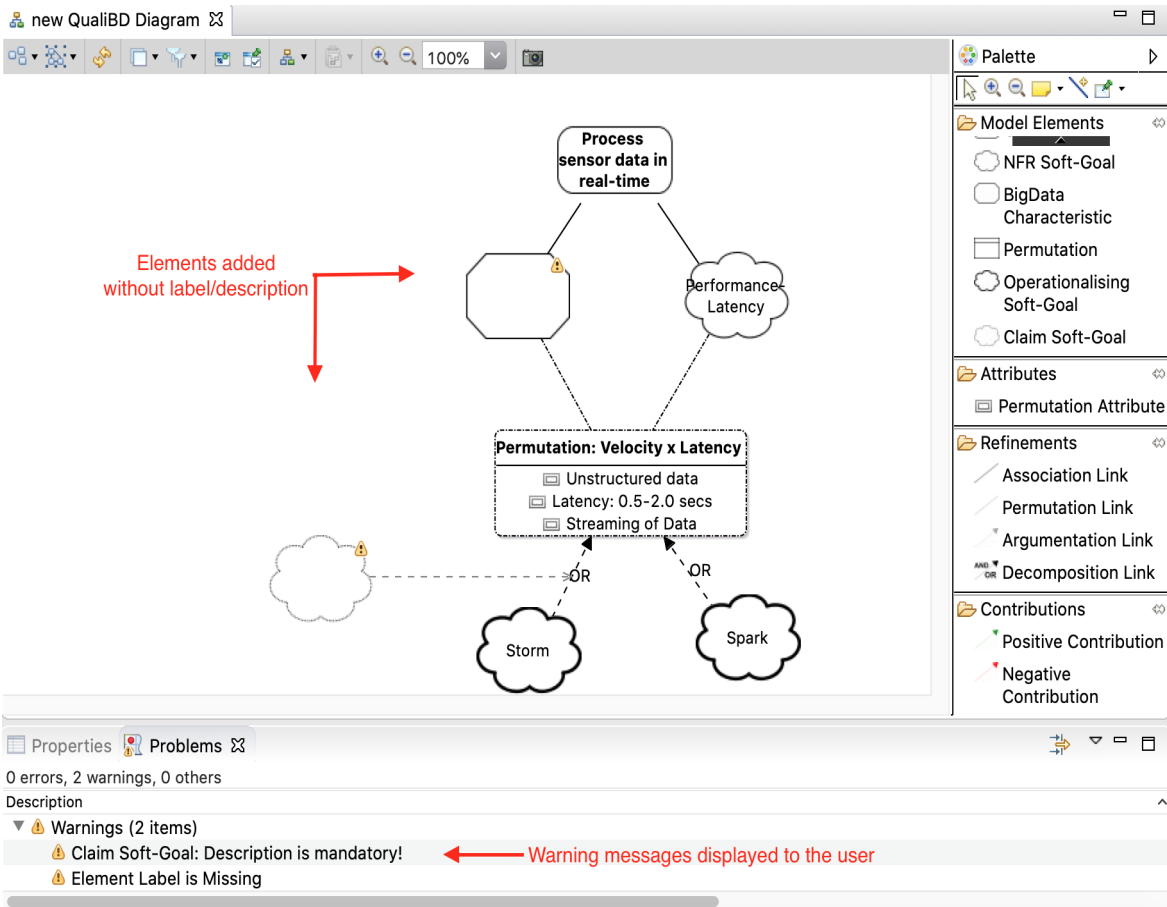


Figure C.2: Validation of model elements added without a label or description

This figure depicts a quality requirement modelled using the QualiBD Tool. In this requirement, two nodes were added to the tool canvas without a label or description. They are: Big Data Characteristic and Claim Soft-goal. By running the diagram validation feature, we invoke a defined AQL expression (AQL expressions are described in Appendix E) that checks the existence of empty labels/descriptions and displays warning messages if the expression returns *true*.

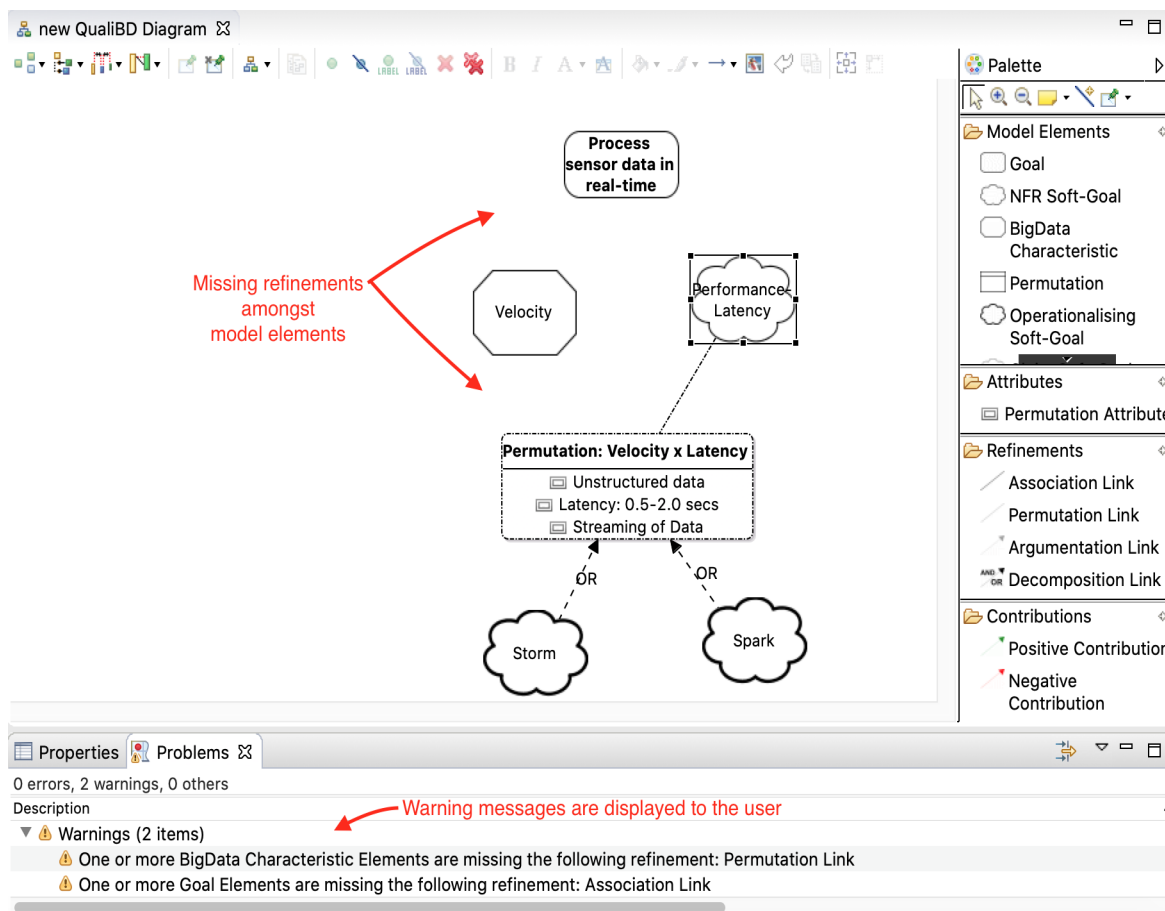


Figure C.3: Validation of missing refinements amongst model elements

This figure depicts a quality requirement modelled using the QualiBD Tool. In this requirement, two nodes were added to the tool canvas without a refinement. They are: Goal and Big Data Characteristic. By running the diagram validation feature, we invoke some defined Java methods (Java methods are described in Appendix D) that check for missing refinements amongst model elements and display warning messages if missing refinements are identified.

Appendix D

QualiBD Tool: Implementation Details

In this appendix, then, we describe the technologies and frameworks used in the implementation of the QualiBD Tool. It is organised as follows: next section overviews the tool implementation process. Section 3 presents the QualiBD Tool graphical user interface and warning messages. Finally, Section 4 summaries this appendix.

D.1 Tool Implementation

The implementation of the QualiBD tool consisted of two major steps: (i) modelling and code generation; and (ii) graphical editor definition. For the modelling and code generation, we used the Eclipse Modelling Framework (EMF), a modelling framework and code generation facility for building tools and applications based on a structured data model [1]. For the graphical portion of the tool, we used Sirius [2], an Eclipse project that allows for the creation of graphical modelling tools by leveraging the Eclipse modelling technologies such as EMF and the Graphical ModellingFramework (GMF).

On the EMF side, we define the domain model and create a concrete instance of that model that is dynamically interpreted using a runtime within the Eclipse IDE environment. On the Sirius side (on the *Sirius Specification Editor*), we define the modelling tool - composed of all modelling elements, behaviour, java services, expressions, and navigation tools. The modelling

tool references domain model defined in EMF. The Graphical representation of the model is created using the defined modelling tool. The graphical representation represents the concrete instance of the defined domain model. The concrete instance of the domain model conforms with the domain model defined in EMF.

D.1.1 Modelling and Code Generation

The implementation of the QualiBD tool started with the definition of a domain model that describes the modelling elements using the EMF. The model used to represent models in EMF is called Ecore [1]. An Ecore can be considered a subset of a UML class diagram [3]. EMF allows for the modelling of meta-class (EClass), packages, and several different types of references (EReferences) such as compositions and inheritance. An EClass can contain different attributes and operations. An attribute (EAttribute) has a data type (EDataType) which can be primitive (e.g., int, float, boolean,) or object type (e.g., a class) [1]. Figure D.1 depicts the Ecore meta-model of the QualiBD Tool.

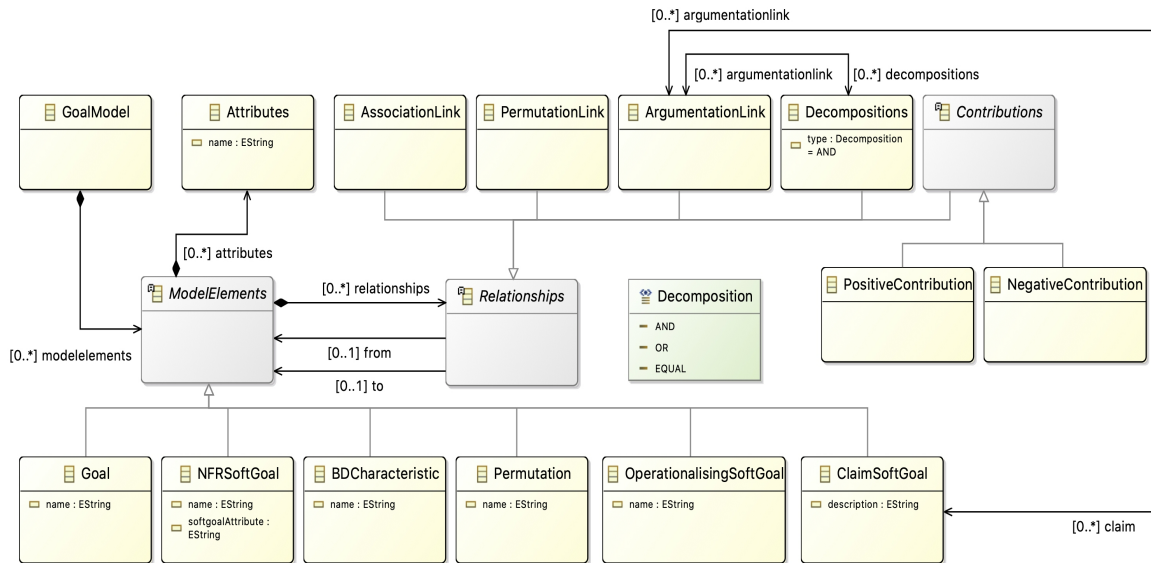


Figure D.1: Ecore Meta-model of the QualiBD Tool.

D.1.2 Graphics Editor

For the definition of the graphical portion of the QualiBD tool, we used Eclipse Sirius. The Sirius official documentation [2] states that a modelling workbench created with Sirius is composed of a set of Eclipse editors (such as diagrams, tables and trees) which enables users to create, edit and visualise EMF models. The editor which summarizes the complete structure of the modelling workbench, its behaviour, and all the edition and navigation tools is dynamically interpreted by a runtime within the Eclipse environment [2]. Before diving into the steps taken during the definition of the graphical portion of the QualiBD Tool, let's discuss some of the concepts underlying Eclipse Sirius [2].

The main concepts in Sirius are (based on the official Sirius documentation [2]) are (i) *Viewpoint*: represents a set of representation specifications and extensions. It is considered one of the core elements of Sirius; (ii) *Representation*: set of graphical elements that represent the domain data, in other words, the concrete instance of the Ecore metamodel; (iii) *Mappings*: identifies the sub-set of semantic model elements that would appear in the representation. It is also used to indicate how they should be represented; (iv) *Styles*: used to customize the appearance of the defined elements; (v) *Tools*: used to add edition capabilities to the graphical editor allowing end-users to create, edit, and delete model elements.

Additionally, when defining model elements, edges and tools in Sirius, we will be using some required interpreted expressions to configure them. These can be queries to select elements or more general-purpose expression to compute a value, for instance [2]. The recommended language for writing queries and expressions in Sirius is the Aceleo Query Language (AQL). It is also used to navigate and query an Ecore model defined in EMF [4]. However, Sirius also supports other common expression interpreters such as (i) *Var*: provides direct access to the value of a named variable; (ii) *Feature*: offers direct access to a named feature of the current element. For example, instead of `aql:self.name`, the equivalent using the Feature interpreter would be `feature:name`; and (iii) *Service*: can used to invoke a service method (e.g., Java services) on the current element.

In the next subsections, we describe the steps followed in order to defined the graphical editor portion of the QualiBD Tool in Sirius (as depicted in Figure D.2). The steps represented

within the “Design and Construction” phase are defined in the Sirius Specification Editor as described in the beginning of Section 2.

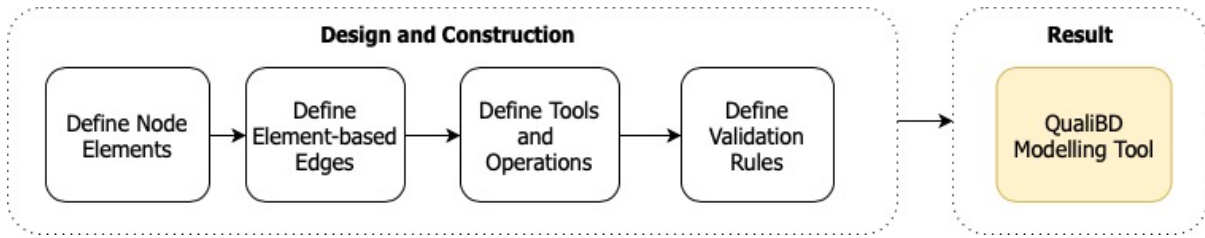


Figure D.2: Steps taken in the definition of the graphical editor portion of the QualiBD Tool.

Defining Node Elements

When creating a node, we must describe which model element will be displayed by the modelling tool. A model element can be displayed via either an image or a geometric shape. For that, we must specify the following properties:

- **ID**: this is the unique identifier of the element we are defining.
- **Domain class**: defines the type of element represented by the node we are creating. For instance, consider: *bigDataModelling::Goal* where *bigDataModelling* is the namespace (NS) prefix, in other words, the name of the Ecore meta-model and *Goal* is the name of the element (Eclass) in that meta-model. Specifying the NS is important to prevent eventual conflicts with another metamodel that could define a class of the same name.
- **Semantic candidate expression**: Restricts to the list of elements to consider before creating the graphical elements. If not set, then all semantic models in session will be browsed and any element of the given type validating the precondition expression will cause the creation of the element. If we set this attribute then, only the elements returned by the expression will be considered. For instance, in the case of our modelling tool, we feature the *modelelements* abstract class where it is the class that extends the goal class and other elements specified in this modelling language.

Defining Element-based Edges

Edges in Sirius can be defined as relation-based or element-based [2]. Relation-based edges are used to represent a relation between model elements such as containment or references whereas element-based edges are used when a semantic model element exists to represent the relation itself [2]. Since all the relationships in the QualiBD tool are represented semantically through classes in the Ecore model, we only used element-based edges. Figure D.3 depicts the properties defined to create the *Permutation Link* element-based edge.

Id*:	<input type="text" value="BDM_PermutationLink"/>
Domain Class*:	<input type="text" value="bigDataModelling::PermutationLink"/>
Source Mapping*:	<input type="text" value="BDM_NFRSoftGoal, BDM_BigDataCharacteristic"/>
Source Finder Expression:	<input type="text" value="feature:from"/>
Target Mapping*:	<input type="text" value="PermutationContainer"/>
Target Finder Expression*:	<input type="text" value="feature:to"/>
Semantic Cand...s Expression:	<input type="text" value="aql:self.modelements.relationships"/>

Figure D.3: Properties of an element-based relation for the *Permutation Link* element in Sirius.

With reference to Figure D.3:

- **ID:** this is the unique identifier of the element we are defining.
- **Domain Class:** the name of the domain class that triggers the creation of the new edge. In the context of Figure 2, the *Permutation Link* class within the *bigDataModelling* meta-model. Again (repeated for convenience), the specification of the NS is important to prevent eventual conflicts with another metamodel that could define a class of the same name.
- **Source Mapping:** maps the element from where the edge should start.
- **Source Finder Expression and Target Finder Expression:** will be evaluated in the context of the semantic element of the edge. It should return the actual elements that the edge connects.

- **Target mapping:** maps the element from which the edge should end.
- **Semantic candidate expression:** Restricts to the list of elements to consider before creating the graphical elements. We use an AQL expression that points the to the *Relationships* abstract class defined in the QualiBD Ecore meta-model. Only the elements returned by the expression will be considered. Example of an AQL expression used:
`aql:self.modelements.relationships.`

Defining Tools and Operations

Once all the elements (nodes and edges) are defined, we can establish the tools that will be displayed in the palette of Eclipse, that turn, will allow the end-user to create and edit new model elements onto the tool container (canvas). Without tools, the models would be “visualisations only”, without any edition capabilities [2].

In the QualiBD Tool, the following types of tools were defined (*i*) *Element Creation*; and (*ii*) *Element Edition*. The former, enables the creation of instances of model elements. The later, adds editing capabilities in the QualiBD Tool. Example of editing capabilities supported by the QualiBD tool are: (*a*) *Direct Edit Label* that allows for the modification a graphical object label (e.g, name of a goal model element) direct from the graphical diagram; and (*b*) *Reconnect Edges* that allows end-users to change the source and/or target of an edge by moving the corresponding end onto another graphical model element [2].

As an example, Figure D.4 depicts the properties - *of the Element Creation Tool* - defined to create the edge tool Permutation Link. Please note that the procedure for creating node tools is similar to edge tools, thus, only one example will be provided.

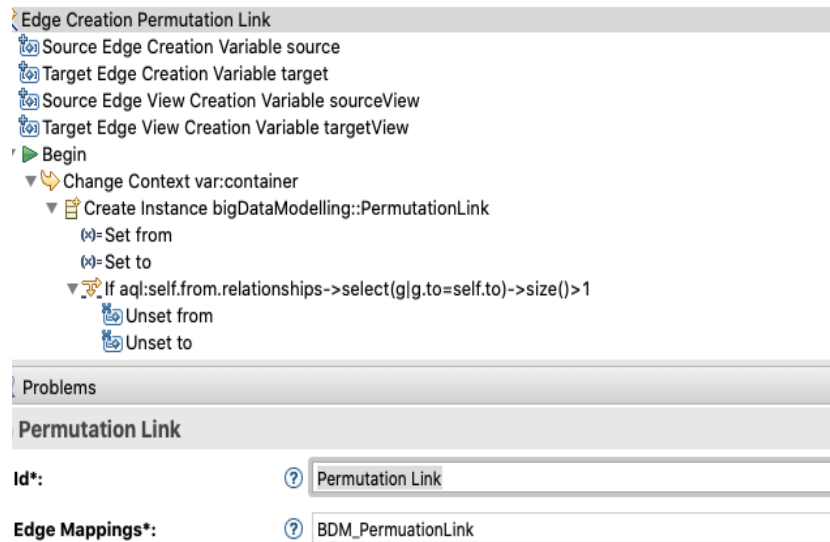


Figure D.4: Properties of an edge creation tool in Sirius for the *Permutation Link* element.

With reference to Figure D.4:

- The *Begin* element has no property. It serves as an entry point to the specification of the behavior of our tool.
- Within the *Begin* element, we define the *Change Context* operation, which serves to changes the context to a new element and executes any contained sub-operations [5]. We also define the *Create Instance* operation that is used to create new semantic elements to be added into the end-users model. For that, we must specify the *Type Name* (using the same syntax as for *Domain Class* properties) of the new object to be created and the *Reference Name* through which the created element will be attached to [5].
- For edge tools (in the context of QualiBD tool), we also specify the *Set* operation that, as the name says, is used to set the value of a feature. It can be an attribute or a reference of the current element. In the case of the *Permutation Link* edge tool, the source and the destination of the relation defined in our Ecore metamodel (in our case the relations *From* and *To*).
- Additionally, we can define some operations to control the behaviour of the tool being created. In the context of the all relation-based elements (such as association, permutation, decomposition, argumentation, and contribution links) defined in the QualiBD

tool, we set an operation to prevent the user to establish duplicated connections amongst model elements. For that, we used the *If* operation that evaluates its *Condition Expression*. An example of a condition expression in AQL used by the *If* operation is:

```
aql:self.from.relationships->select(g|g.to=self.to)->size()>1.
```

If the expression (interpreted as a boolean) returns “false”, *If* does nothing. If it returns “true”, then it executes any sub-operations defined under the *If* operation in the order of definition [5]. In the case of the QualiBD tool, it “cancels” (unsets) the creation of edges between two nodes when the relation already exists in the representation.

Define Validation Rules

In order to allow the QualiBD tool to function properly and minimise possible user induced errors, we specified some rudimentary validation features focused on the *completeness* and *accuracy* of the models created. The validation features are briefly described as follows:

- **Empty Labels:** This rule checks for the existence of model elements with empty labels. For that, we specified a semantic validation rule. Eclipse Sirius offers three levels of semantic validation rules: Message, Warning, and Error. The semantic validation rule is characterised by an *audit expression*. If the audit expression returns *True*, then nothing happens. If the audit expression returns *False*, then a validation issue will be pointed out (e.g., a warning message to be displayed to the end-user in the *Problems* view of Eclipse). An example of an audit expression used is `aql:self.name<>null`.
- **Model Elements with Empty Connections:** This rule checks for the existence of model elements with empty connections (*refinements*). For that, we define the same procedures described in the “Empty Labels Validation”. However, for this one specifically, instead of using an AQL expression in the audit expression definition, we used a Java service method that navigates the existing model elements on the tool canvas, and checks whether there are established connections amongst them. If empty connections are found, it returns the validation warning message. Otherwise, it returns *null*.

D.2 Summary

In this paper, we described the implementation of the QualiBD, a tool for modelling quality requirements for Big Data Software applications. The definition of our tool is comprised of two steps: (i) domain model definition and code generation, and(ii) graphical editor definition. For that, we used eclipse modelling and Sirius Frameworks, respectively. On the EMF side, we defined the tool's structured data model. On the Sirius side, we defined the graphical editor behaviour. The graphical editor portion of the tool was defined in four incremental steps: (i) definition of node elements, (ii) definition of edge elements, (iii) definition of tools and operations, and (iv) definition of validations rules.

NOTE:

The documents containing the Java services methods and AQL expressions used in the definition of QualiBD Tool can be seen in Appendices E and F, respectively.

Bibliography

- [1] D. Steinberg, F. Budinsky, and E. Merks, *EMF: Eclipse Modeling Framework*, ser. Eclipse (Addison-Wesley). Addison-Wesley, 2009.
- [2] (2019) Eclipse sirius official documentation. [Online]. Available: <https://www.eclipse.org/sirius/doc/>
- [3] (2019) Eclipse modelling framework official documentation. [Online]. Available: <https://www.eclipse.org/modeling/emf/docs/>
- [4] (2019) Acceleo query language: Query and navigate in emf models. [Online]. Available: <https://www.eclipse.org/acceleo/documentation/>
- [5] (2019) Specifyin model operations in sirius. [Online]. Available: <https://www.eclipse.org/sirius/doc/>

Appendix E

QualiBD Tool: Java Methods

In this appendix, we describe some of the java methods used in the implementation of the validation procedures (with regards to model refinement completeness and accuracy) of the QualiBD tool.

```
public String CheckGoalElementRefinement(GoalModel element) {
    String output = "";
    for(ModelElements node : ((GoalModel) element).getModelements()) {
        if(node instanceof Goal) {
            Goal goal= (Goal) node;
            if(goal.getRelationships().isEmpty()) {
                output += "One or more Goal Elements are missing the following
                    ↔ refinement: Association Link";
            }
        }
    }
    return output;
}
```

Listing E.1: Method for checking Goal elements empty refinements

```

public String CheckNFRGoalElementRefinement(GoalModel element) {
    String output = "";
    for(ModelElements node : ((GoalModel) element).getModelelements()) {
        if(node instanceof NFRSoftGoal) {
            NFRSoftGoal NFRgoal= (NFRSoftGoal) node;
            if(NFRgoal.getRelationships().isEmpty()) {
                output += "One or more NFR Soft-Goal Elements are missing the
                    ↪ following refinement: Association Link";
            }
        }
    }
    return output;
}

```

Listing E.2: Method for checking NFR Soft-Goal elements empty refinements

```

public String CheckBigDataElementRefinement(GoalModel element) {
    String output = "";
    for(ModelElements node : ((GoalModel) element).getModelelements()) {
        if(node instanceof BDCharacteristic) {
            BDCharacteristic bigdata= (BDCharacteristic) node;
            if(bigdata.getRelationships().isEmpty()) {
                output += "One or more BigData Characteristic Elements are missing
                    ↪ the following refinement: Association Link";
            }
        }
    }
    return output;
}

```

Listing E.3: Method for checking Big Data Characteristic elements empty refinements

```

public String CheckPermutationElementRefinement(GoalModel element) {
    String output = "";
    for(ModelElements node : ((GoalModel) element).getModelements()) {
        if(node instanceof Permutation) {
            Permutation permutation= (Permutation) node;
            if(permutation.getRelationships().isEmpty()) {
                output += "One or more Permutation Container Elements are missing the
                    ↔ following refinement: Decomposition Link";
            }
        }
    }
    return output;
}

```

Listing E.4: Method for checking Permutation Container elements empty refinements

```

public String CheckClaimElementRefinement(GoalModel element) {
    String output = "";
    for(ModelElements node : ((GoalModel) element).getModelements()) {
        if(node instanceof ClaimSoftGoal) {
            ClaimSoftGoal claim= (ClaimSoftGoal) node;
            if(claim.getRelationships().isEmpty()) {
                output += "One or more Claim Soft-Goal Elements are missing the
                    ↔ following refinement: Argumentation Link";
            }
        }
    }
    return output;
}

```

Listing E.5: Method for checking Claim Soft-goal elements empty refinements

Appendix F

QualiBD Tool: AQL Expressions

Below, we describe some examples of AQL expressions used in the definition of the model elements (e.g., Goal, NFR Soft-goal, Permutation Container, Refinement Links, Decomposition Links, etc.) of the graphical portion of the QualiBD Tool.

```
//Expression that points to the Relationships class defined in the  
↪ Ecore meta-model.  
aql:self.modelements.relationships
```

Listing F.1: AQL Expression 1

```
//Expression used to validate if a model element is connected to  
↪ another model element or itself.  
aql:self.from.relationships->first()<>self and self.from.relationships  
↪ ->select(g|g.to=self.to)->size()>1
```

Listing F.2: AQL Expression 2

```
//Expression that calls the CheckNFRGoalElementRefinement() method to
    ↪ check if there are model element (in this example, NFR Soft-goals
    ↪ ) with missing refinements.
aql:self.CheckNFRGoalElementRefinement().toString().size()=0
```

Listing F.3: AQL Expression 3

```
//Expression used to check if there are model elements added to the
    ↪ tool canvas without a description (in the case of Claim Soft-
    ↪ goals) or labels (in the case of all other model elements).
aql:self.name <> null
aql:self.description <> null
```

Listing F.4: AQL Expression 4

```
//Expression used to define a structured naming format for the element
    ↪ Permutation Container
aql:'Permutation: '+ self.name
```

Listing F.5: AQL Expression 5

Curriculum Vitae

Name: Darlan Florencio de Arruda

Post-Secondary Education and Degrees: University of Pernambuco
Recife, PE, Brazil
M.Sc., Computer Engineering (2012 - 2014)

University of Pernambuco
Caruaru, PE, Brazil.
B.Sc., Information Systems (2007 - 2011)

Honours and Awards: Full PhD Scholarship (CNPq - Brazil)
2015-2019

Western Graduate Research Scholarship (WGRS)
2015-2018

Related Work Experience: Teaching Assistant
The University of Western Ontario
2016 - 2019

Software Test Engineer
CESAR Brazil
2013-2015

Software Test Analyst
Accenture Brazil
2012-2013

Software Test Engineer
CIN-UFPE/Motorola Mobility Brazil
2011-2012

Recent Publications:

- D. Arruda and N. H. Madhavji, **QualiBD: A tool for modelling quality requirements for Big Data applications** in *2019 IEEE International Conference on Big Data, (Big-Data 2019)*, Los Angeles, CA, USA, December 09-12, 2019.
- D. Arruda, N. H. Madhavji, and I. Noorwali, **A validation study of a requirements engineering artefact model for Big Data software development projects**, in *Proceedings of the 14th International Conference on Software Technologies, ICSoft 2019*, Prague, Czech Republic, July 26-28, 2019, pp. 106-116.
- I. Noorwali, N. H. Madhavji, D. Arruda, and R. Ferrari, **Towards a meta-model for requirements-driven information for internal stakeholders**, in *Requirements Engineering: Foundation for Software Quality, REFSQ 2019*, Essen, Germany, March 18 - 21, Proceedings, 2019, pp. 262 - 278
- D. Arruda, N. H. Madhavji, and C. A. Taylor, **CASCON workshop on developing Big Data applications and services**, in *Proceedings of the 28th Annual International Conference on Computer Science and Software Engineering, CASCON 2018*, Markham, Ontario, Canada, October 29-31, 2018, pp. 407-409. [*Extended Abstract*]
- D. Arruda and N. H. Madhavji, **State of requirements engineering research in the context of Big Data applications**, in *Requirements Engineering: Foundation for Software Quality*, E. Kamsties, J. Horkoff, and F. Dalpiaz, Eds. Cham: Springer International Publishing, 2018, pp. 307-323.
- D. Arruda, **Requirements engineering in the context of big data applications**, *ACM SIGSOFT Software Engineering Notes*, vol. 43, no. 1, pp. 1-6, 2018.
- D. Arruda and N. H. Madhavji, **Towards a Requirements Engineering Artefact Model in the context of Big Data Software Development Projects: Research in Progress** in *2017 IEEE International Conference on Big Data, (BigData 2017)*, Boston, MA, USA, December 11-14, 2017, pp. 2232-2237.

- D. Arruda and N. H. Madhavji, **Towards a Big Data requirements engineering artefact model in the context of Big Data software development projects: Poster extended abstract**, in *2017 IEEE International Conference on Big Data, (BigData 2017)*, Boston, MA, USA, December 11-14, 2017, pp. 4725–4726.
- D. Arruda and N. H. Madhavji, **The role of big data analytics in corporate decision-making**, in *Proceedings of the 6th International Conference on Data Science, Technology and Applications, DATA Madrid, Spain, July 24-26, 2017*, pp. 28-37. (*Student best paper award nominee*)
- I. Noorwali, D. Arruda, and N. H. Madhavji, **Understanding quality requirements in the context of Big Data systems**, *Proceedings of the 2nd International Workshop on Big Data Software Engineering - BIGDSE 16*, pp. 76-79, 2016.