

Project Title	High-performance data-centric stack for big data applications and operations
Project Acronym	BigDataStack
Grant Agreement No	779747
Instrument	Research and Innovation action
Call	Information and Communication Technologies Call (H2020-ICT-2016-2017)
Start Date of Project	01/01/2018
Duration of Project	36 months
Project Website	http://bigdatastack.eu/

D2.3 – Requirements & State of the Art Analysis – III

Work Package	WP2 – Requirements, Architecture & Technical Coordination
Lead Author (Org)	Orlando Avila-García (ATOS)

Contributing Author(s) (Org)	Paula Ta-Shma, Yosef Moatti (IBM), Mauricio Fadel, Bin Chen (NEC), Ismael Cuadrado, Ana Belén González, Bernat Quesada, Alberto Soler (ATOS), Stathis Plitsos (DAN), Anestis Sidiropoulos, Amaryllis Raouzaïou (ATC), Jose María Zaragoza, Jesus Gallego (LXS), Sophia Karagiorgou, Panagiotis Gouvas (UBI), Dimitris Pouloupoulos, Stavroula Meimetea, Maria Kanakari, Christos Doukeridis, Giannis Poulakis, Dimosthenis Kyriazis (UPRC), Marta Patino (UPM), Richard McCreadie (GLA), Miki Kenneth, Luis Tomas (RHT), Nikos Drosos (SILO), Maurizio Megliola (GFT)
Reviewer(s) (Org)	Sophia Karagiorgou (UBI), Richard McCreadie (GLA), Yosef Moatti (IBM), Dimosthenis Kyriazis (UPRC)
Due Date	31.10.2019
Date	08.11.2019
Version	1.0

Dissemination Level

- PU: Public (*on-line platform)
- PP: Restricted to other programme participants (including the Commission)
- RE: Restricted to a group specified by the consortium (including the Commission)
- CO: Confidential, only for members of the consortium (including the Commission)

Versioning and contribution history

Ver.	Date	Author	Notes
0.1	01.10.2019	Orlando Avila-García (ATOS)	Creation of the first draft based on D2.2 v1.0.
0.2	24.10.2019	Orlando Avila-García (ATOS)	Adding updated to use case requirements from use case providers ATOS, GFT.
0.3	28.10.2019	Orlando Avila-García (ATOS)	Adding updates to system and software requirements from technology providers ATC, GLA, NEC, RHT and UPRC. Adding updates to baseline technologies by ATC, NEC and UPRC.
0.4	30.10.2019	Orlando Avila-García (ATOS)	Adding updates to system and software requirements from technology providers LXS, IBM and ATOS. Adding updates to baseline technologies by IBM and LXS.
0.5	31.10.2019	Orlando Avila-García (ATOS)	Adding updates to baseline technologies by UBI.
0.6	06.11.2019	Orlando Avila-García (ATOS)	Addressing corrections and amendments suggested by the internal reviewers from IBM, UPRC, UBI and GLA.
0.7	07.11.2019	Orlando Avila-García (ATOS)	Addressing comments made by the internal reviewers from IBM, UPRC, UBI and GLA, to requirements from GFT, UPRC and LXS, and baseline technologies from NEC and UPM.
1.0	08.11.2019	Orlando Avila-García (ATOS)	Final version after minor corrections and formatting changes.

Disclaimer

This document contains information that is proprietary to the BigDataStack Consortium. Neither this document nor the information contained herein shall be used, duplicated or communicated by any means to a third party, in whole or parts, except with the prior consent of BigDataStack Consortium.

Table of Contents

1	Executive Summary	9
2	Introduction	10
2.1	Method.....	11
2.2	Organization	14
3	Business Stakeholders and Goals.....	16
3.1	Stakeholder Categories.....	16
3.2	Business Model.....	17
3.3	Business Outcomes	18
3.4	Business Goals.....	19
4	Business Requirements and Scenarios	22
4.1	Real-time Ship Management.....	22
4.2	Connected Consumer	29
4.3	Smart Insurance.....	38
5	Platform Roles.....	44
6	Infrastructure-Data Management Requirements	45
7	Data-as-a-Service Requirements	64
8	Dimensioning, Modelling & Interaction Services Requirements	79
9	Baseline Technologies	95
9.1	Computing Resources Management.....	95
9.2	Storage Resources Management.....	96
9.3	Data-driven Network Management.....	97
9.4	Dynamic Orchestrator	97
9.5	Triple Monitoring	98
9.6	Applications & Data Services Deployment	100
9.7	Distributed Storage & Analytics	103
9.8	Live Migration	105
9.9	Data Quality Assessment.....	105
9.10	Big Data Layout	106
9.11	Real-time CEP	109
9.12	Predictive and Process Analytics	109
9.13	Seamless Analytics Framework	110
9.14	Application Dimensioning Workbench	111
9.15	Process modelling framework	112
9.16	Data Toolkit.....	113
9.17	Adaptable Visualizations	114
10	Bibliography	116

List of tables

Table 1 – Requirements engineering processes	11
Table 2 – Levels of requirement specification as defined in ISO/IEC/IEEE 29148:2011 [41]13	
Table 3 – Stakeholders	16
Table 4 – Stakeholder concerns.....	17
Table 5 – Preliminary business model.....	18
Table 6 – Stakeholder requirements	19
Table 7 – Privacy and security (business goal)	19
Table 8 – Attractive revenue and business model (business goal)	19
Table 9 – High performance, scalability and sharing (business goal)	20
Table 10 – Product integration (business goal)	20
Table 11 – Different analytic capabilities (business goal)	20
Table 12 – Ease of use (business goal)	21
Table 13 – Monitoring and predictive maintenance scenario description (scenario)	24
Table 14 – Order suggestion and dynamic routing (scenario)	25
Table 15 – Main Engine Monitoring (stakeholder requirement)	26
Table 16 – Malfunction alert (stakeholder requirement).....	27
Table 17 – Alert Inspection (stakeholder requirement).....	27
Table 18 – Spare Part Requisition (stakeholder requirement)	27
Table 19 – Requisition Process (stakeholder requirement)	28
Table 20 – Dynamic Routing (stakeholder requirement).....	29
Table 21 – Retail Recommender (scenario).....	32
Table 22 – Retail Demonstrator (scenario).....	34
Table 23 – Predict products required by a recurrent user (stakeholder requirement)	34
Table 24 – Predict products to a new user (stakeholder requirement).....	35
Table 25 – Recommend personalized discounts (stakeholder requirement).....	35
Table 26 – Data Requirement (stakeholder requirement).....	36
Table 27 – Clients requirements (stakeholder requirement)	36
Table 28 – Products requirements (stakeholder requirement)	36
Table 29 – Multi-device (stakeholder requirement).....	37
Table 30 – Easy-to-use (stakeholder requirement).....	37
Table 31 – Multi-user (stakeholder requirement).....	37
Table 32 – Data security (stakeholder requirement).....	38
Table 33 – Services security (stakeholder requirement).....	38
Table 34 – Customers segmentation scenario	40
Table 35 – Customer lifetime value prediction scenario	41
Table 36 – Provide personalized policies required by customer (stakeholder requirement). 41	
Table 37 – Provide personalized guarantees required by customer (stakeholder requirement)	42
.....	
Table 38 – Customers (stakeholder requirement)	42
Table 39 – Policies (stakeholder requirement)	42
Table 40 – Easy-to-use (stakeholder requirement).....	43
Table 41 – Data security (stakeholder requirement).....	43
Table 42 – BigDataStack Platform roles.....	44
Table 43 - Requirement (1) for Cluster Management	45
Table 44 - Requirement (2) for Cluster Management	46
Table 45 - Requirement (3) for Cluster Management	46
Table 46 - Requirement (4) for Cluster Management	46
Table 47 - Requirement (5) for Cluster Management	47
Table 48 - Requirement (6) for Cluster Management	47
Table 49 - Requirement (7) for Cluster Management	47

Table 50 - Requirement (8) for Cluster Management	48
Table 51 - Requirement (1) for Dynamic Orchestrator	48
Table 52 - Requirement (2) for Dynamic Orchestrator	48
Table 53 - Requirement (3) for Dynamic Orchestrator	49
Table 54 - Requirement (4) for Dynamic Orchestrator	49
Table 55 - Requirement (5) for Dynamic Orchestrator	49
Table 56 - Requirement (1) for ADS Ranking	50
Table 57 - Requirement (2) for ADS Ranking	50
Table 58 - Requirement (3) for ADS Ranking	51
Table 59 - Requirement (4) for ADS Ranking	51
Table 60 - Requirement (5) for ADS Ranking	51
Table 61 - Requirement (6) for ADS Ranking	52
Table 62 - Requirement (7) for ADS Ranking	52
Table 63 - Requirement (8) for ADS Ranking	53
Table 64 - Requirement (1) for ADS Deploy	53
Table 65 - Requirement (2) for ADS Deploy	53
Table 66 - Requirement (3) for ADS Deploy	53
Table 67 - Requirement (4) for ADS Deploy	54
Table 68 - Requirement (5) for ADS Deploy	54
Table 69 - Requirement (6) for ADS Deploy	54
Table 70 - Requirement (1) for QoS Evaluation	55
Table 71 - Requirement (2) for QoS Evaluation	55
Table 72 - Requirement (3) for QoS Evaluation	55
Table 73 - Requirement (4) for QoS Evaluation	55
Table 74 - Requirement (5) for QoS Evaluation	56
Table 75 - Requirement (6) for QoS Evaluation	56
Table 76 - Requirement (7) for QoS Evaluation	56
Table 77 - Requirement (1) for Triple Monitoring Engine	57
Table 78 - Requirement (2) for Triple Monitoring Engine	57
Table 79 - Requirement (3) for Triple Monitoring Engine	57
Table 80 - Requirement (4) for Triple Monitoring Engine	58
Table 81 - Requirement (5) for Triple Monitoring Engine	58
Table 82 - Requirement (6) for Triple Monitoring Engine	59
Table 83 - Requirement (7) for Triple Monitoring Engine	59
Table 84 - Requirement (8) for Triple Monitoring Engine	59
Table 85 - Requirement (9) for Triple Monitoring Engine	60
Table 86 - Requirement (10) for Triple Monitoring Engine	60
Table 87 - Requirement (11) for Triple Monitoring Engine	60
Table 88 - Requirement (12) for Triple Monitoring Engine	61
Table 89 - Requirement (13) for Triple Monitoring Engine	61
Table 90 - Requirement (1) for Global Decision Tracker	61
Table 91 - Requirement (2) for Global Decision Tracker	62
Table 92 - Requirement (1) for Information-Driven Networking	62
Table 93 - Requirement (2) for Information-Driven Networking	63
Table 94 - Requirement (1) for Big Data Layout	64
Table 95 - Requirement (2) for Big Data Layout	64
Table 96 - Requirement (3) for Big Data Layout	64
Table 97 - Requirement (4) for Big Data Layout	65
Table 98 - Requirement (5) for Big Data Layout	65
Table 99 - Requirement (6) for Big Data Layout	65
Table 100 - Requirement (7) for Big Data Layout	66
Table 101 - Requirement (8) for Big Data Layout	66

Table 102 - Requirement (9) for Big Data Layout.....	66
Table 103 - Requirement (1) for Adaptable Distributed Storage.....	66
Table 104 - Requirement (2) for Adaptable Distributed Storage.....	67
Table 105 - Requirement (3) for Adaptable Distributed Storage.....	67
Table 106 - Requirement (4) for Adaptable Distributed Storage.....	68
Table 107 - Requirement (5) for Adaptable Distributed Storage.....	68
Table 108 - Requirement (6) for Adaptable Distributed Storage.....	69
Table 109 - Requirement (7) for Adaptable Distributed Storage.....	69
Table 110 - Requirement (8) for Adaptable Distributed Storage.....	69
Table 111 - Requirement (1) for Seamless Data Analytics.....	70
Table 112 - Requirement (2) for Seamless Data Analytics.....	70
Table 113 - Requirement (3) for Seamless Data Analytics.....	71
Table 114 - Requirement (4) for Seamless Data Analytics.....	71
Table 115 - Requirement (5) for Seamless Data Analytics.....	72
Table 116 - Requirement (6) for Seamless Data Analytics.....	72
Table 117 - Requirement (7) for Seamless Data Analytics.....	73
Table 118 - Requirement (8) for Seamless Data Analytics.....	73
Table 119 - Requirement (9) for Seamless Data Analytics.....	74
Table 120 - Requirement (10) for Seamless Data Analytics.....	74
Table 121 - Requirement (1) for Data Quality Assessment & Improvement.....	74
Table 122 - Requirement (2) for Data Quality Assessment & Improvement.....	75
Table 123 - Requirement (3) for Data Quality Assessment & Improvement.....	75
Table 124 - Requirement (4) for Data Quality Assessment & Improvement.....	75
Table 125 - Requirement (5) for Data Quality Assessment & Improvement.....	76
Table 126 - Requirement (6) for Data Quality Assessment & Improvement.....	76
Table 127 - Requirement (1) for Predictive & Process Analytics.....	76
Table 128 - Requirement (2) for Predictive & Process Analytics.....	76
Table 129 - Requirement (3) for Predictive & Process Analytics.....	77
Table 130 - Requirement (4) for Predictive & Process Analytics.....	77
Table 131 - Requirement (1) for Complex Event Processing.....	77
Table 132 - Requirement (2) for Complex Event Processing.....	78
Table 133 - Requirement (3) for Complex Event Processing.....	78
Table 134 - Requirement (4) for Complex Event Processing.....	78
Table 135 – Requirement (1) for Process Modelling Framework.....	79
Table 136 – Requirement (2) for Process Modelling Framework.....	79
Table 137 – Requirement (3) for Process Modelling Framework.....	80
Table 138 – Requirement (4) for Process Modelling Framework.....	80
Table 139 – Requirement (5) for Process Modelling Framework.....	80
Table 140 – Requirement (6) for Process Modelling Framework.....	80
Table 141 – Requirement (7) for Process Modelling Framework.....	81
Table 142 – Requirement (8) for Process Modelling Framework.....	81
Table 143 – Requirement (9) for Process Modelling Framework.....	81
Table 144 – Requirement (10) for Process Modelling Framework.....	81
Table 145 – Requirement (1) for Process Mapping.....	82
Table 146 – Requirement (2) for Process Mapping.....	82
Table 147 – Requirement (3) for Process Mapping.....	82
Table 148 – Requirement (4) for Process Mapping.....	83
Table 149 – Requirement (5) for Process Mapping.....	83
Table 150 – Requirement (6) for Process Mapping.....	83
Table 151 – Requirement (7) for Process Mapping.....	83
Table 152 – Requirement (1) for Data Toolkit.....	84
Table 153 – Requirement (2) for Data Toolkit.....	84

Table 154 – Requirement (3) for Data Toolkit	84
Table 155 – Requirement (4) for Data Toolkit	85
Table 156 – Requirement (1) for Pattern Generator	85
Table 157 – Requirement (2) for Pattern Generator	85
Table 158 – Requirement (3) for Pattern Generator	85
Table 159 – Requirement (4) for Pattern Generator	86
Table 160 – Requirement (5) for Pattern Generator	86
Table 161 – Requirement (1) for Application Dimensioning Workbench	87
Table 162 – Requirement (2) for Application Dimensioning Workbench	87
Table 163 – Requirement (3) for Application Dimensioning Workbench	87
Table 164 – Requirement (4) for Application Dimensioning Workbench	88
Table 165 – Requirement (5) for Application Dimensioning Workbench	88
Table 166 – Requirement (6) for Application Dimensioning Workbench	88
Table 167 – Requirement (7) for Application Dimensioning Workbench	89
Table 168 – Requirement (8) for Application Dimensioning Workbench	89
Table 169 – Requirement (9) for Application Dimensioning Workbench	89
Table 170 – Requirement (10) for Application Dimensioning Workbench	90
Table 171 – Requirement (11) for Application Dimensioning Workbench	90
Table 172 – Requirement (12) for Application Dimensioning Workbench	91
Table 173 – Requirement (13) for Application Dimensioning Workbench	91
Table 174 – Requirement (1) for Adaptable Visualizations	91
Table 175 – Requirement (2) for Adaptable Visualizations	92
Table 176 – Requirement (3) for Adaptable Visualizations	92
Table 177 – Requirement (4) for Adaptable Visualizations	92
Table 178 – Requirement (5) for Adaptable Visualizations	92
Table 179 – Requirement (6) for Adaptable Visualizations	93
Table 180 – Requirement (7) for Adaptable Visualizations	93
Table 181 – Requirement (8) for Adaptable Visualizations	93
Table 182 – Requirement (9) for Adaptable Visualizations	93
Table 183 – Requirement (10) for Adaptable Visualizations	94
Table 184 – Requirement (11) for Adaptable Visualizations	94
Table 185 – Requirement (12) for Adaptable Visualizations	94
Table 186 – Requirement (13) for Adaptable Visualizations	94

List of figures

Figure 1. Requirements engineering method	12
Figure 2. BigDataStack core platform capabilities	14
Figure 3. Netdata role in triple monitoring.....	100
Figure 4. Data ingestion in Watson IoT Platform	111
Figure 5. Node-red programming example.....	112
Figure 6. Rete-based editor for IoT applications modeling.	113

1 Executive Summary

This is the final version of a series of three deliverables specifying the business stakeholder (business) as well as technical (software and technology) requirements for the overall BigDataStack environment. These three versions of the requirements specification have been delivered according to the plan at M6 (D2.1), M11 (D2.2) and M22 (D2.3, the present version).

In the requirements analysis presented in this document, a top-down approach is taken with respect to the user requirements, which have been collected through the BigDataStack use case providers. This is complemented with a bottom-up approach aiming to identify, collect, and analyse the rest of the stakeholder requirements as well as technical requirements from the BigDataStack technology.

The analysis has produced measurable and unambiguous requirements, which inform and drive architectural and design at different levels of the BigDataStack platform: capabilities, services and technologies. To contextualize this analysis, this deliverable also introduces the state-of-the-art (baseline) technologies that may play a role in BigDataStack research and implementation activities. In fact, we do not simply state some state-of-the-art technologies but rather link them to BigDataStack and elaborate what BigDataStack can obtain from them as a baseline.

Note that the set of requirements contained in this deliverable supersedes those specified in the first and second version of the requirements specification (D2.1 at M6 and D2.2 at M11). This version captures the rationale behind the architecture decisions described in the final version of the architecture as described in D2.5 (M18), and design decisions described in the design specifications of the main pillars of BigDataStack as detailed in the corresponding deliverables: D3.2, D4.2 and D5.2 (due M23).

2 Introduction

The main purpose of this deliverable is to describe the set of measurable and unambiguous **business and technical requirements for the BigDataStack environment** at M22. This set will be further tracked to validate the architecture development and implementation during the project lifetime.

This report represents the third official deliverable of BigDataStack project's Task 2.1, whose main goal is to collect the user and technical requirements and track them during the project. The outcomes of this task are a **key input for the architecture as well as component design and implementation activities** of the project.

Task 2.1 started at M1 and produced the first version of the requirements analysis (D2.1) at M6 and second version (D2.2) at M11. The continuous work on analysis has produced a new version of the requirements specification at M22, giving rise to the present deliverable D2.3.

The main contribution (and difference) of this deliverable with respect to the second requirements specification (D2.2) is the following:

- a) Minor updates to the GFT's use case: Smart Insurance ([Section 4.3](#)).
- b) Updates to system and technology requirements ([Sections 6, 7 and 8](#)).
- c) Updates to baseline technologies ([Section 9](#)):
 - 9.3 Data-driven Network Management
 - 9.4 Dynamic Orchestrator
 - 9.9 Data Quality Assessment
 - 9.10 Big Data Layout
 - 9.11 Real-time CEP
 - 9.14 Application Dimensioning Workbench
 - 9.15 Process Modelling Framework
 - 9.17 Adaptable Visualizations

Like in the first and second requirements analysis (D2.1 and D2.2), this elicitation and analysis of the requirements has been carried out considering the needs and concerns coming from the communities and end users related to BigDataStack use cases and technology providers. Therefore, the analysis specifies not only **business requirements** (called "stakeholder requirements" by ISO/IEC/IEEE 29148:2011 [41]) but also **technical requirements** (called either "system requirements" or "software requirements" by the same norm). The method is fully explained at Section 2.1.

Note that use case scenarios, that is, the scenarios of use of the platform in the use case implementations, was firstly described in D6.1 (M18) and will be further refined in D6.2 (M34). **Those scenarios are different** from the stakeholder or business scenarios described in this document: while these scenarios describe the business processes and (abstract or conceptual) requirements behind the use cases, the so-called use cases scenarios presented in D6.1/2 describe the sequence of activities using the different resources and tools of the BigDataStack platform to implement those processes and requirements.

To contextualize the analysis of requirements, Section 3 describes the BigDataStack platform **stakeholders, business model, expected business outcomes and business goals**. To better understand the software technology requirements, this deliverable also includes an introduction to the state-of-the-art (baseline) technologies that are expected to be relevant

for BigDataStack (Section 9). In fact, Sections 3 and 9 are a refinement of the sections dedicated respectively to business goals and baseline technologies in the internal joint (with architecture) report completed at M3 (and brought as-is from D2.1).

The rest of the document is organized as follows: Section 2.1 explains the requirements engineering method and Section 2.2 the organization of requirements in this deliverable; Section 4 specifies business scenarios and use case requirements related to three use cases; Section 5 introduces the most relevant roles that the actors (stakeholders) interacting with the platform may take; Sections 6, 7 and 8 specify the rest of stakeholder requirements as well as technical (system and software) requirements; finally, Section 9 describes baseline technologies relevant for BigDataStack.

It is important to note that the requirements specified in this deliverable are brought into the design specifications of the corresponding key pillars of BigDataStack—captured in D3.2, D4.2 and D5.2 (due M23) for the reader’s convenience, i.e., for a better understanding of the design of BigDataStack capabilities described in those deliverables: Data-driven Infrastructure Management (D3.2), Data as a Service (D4.2), and Dimensioning, Modelling and Interaction Services (D5.2). Nevertheless, the present document should be considered the single source of all requirements and be considered the master version of them in case of discrepancies.

2.1 Method

The requirements engineering method will follow ISO/IEC/IEEE 29148:2011 [41] which describes two main processes or practices to be executed in an iterative and recursive fashion:

Process	Purpose	Output
Stakeholder Requirements Definition Process	To define the requirements for a system that can provide the services needed by users and other stakeholders in a defined environment.	Stakeholder Requirements Specification (<i>StRS</i>)
Requirements Analysis Process	To transform the stakeholder, requirement-driven view of desired services into a technical view of a required product that could deliver those services.	<i>System Requirements Specification (SyRS)</i> <i>Software Requirements Specification (SRS)</i>

Table 1 – Requirements engineering processes

BigDataStack Requirements Engineering¹ Iterative and recursive method

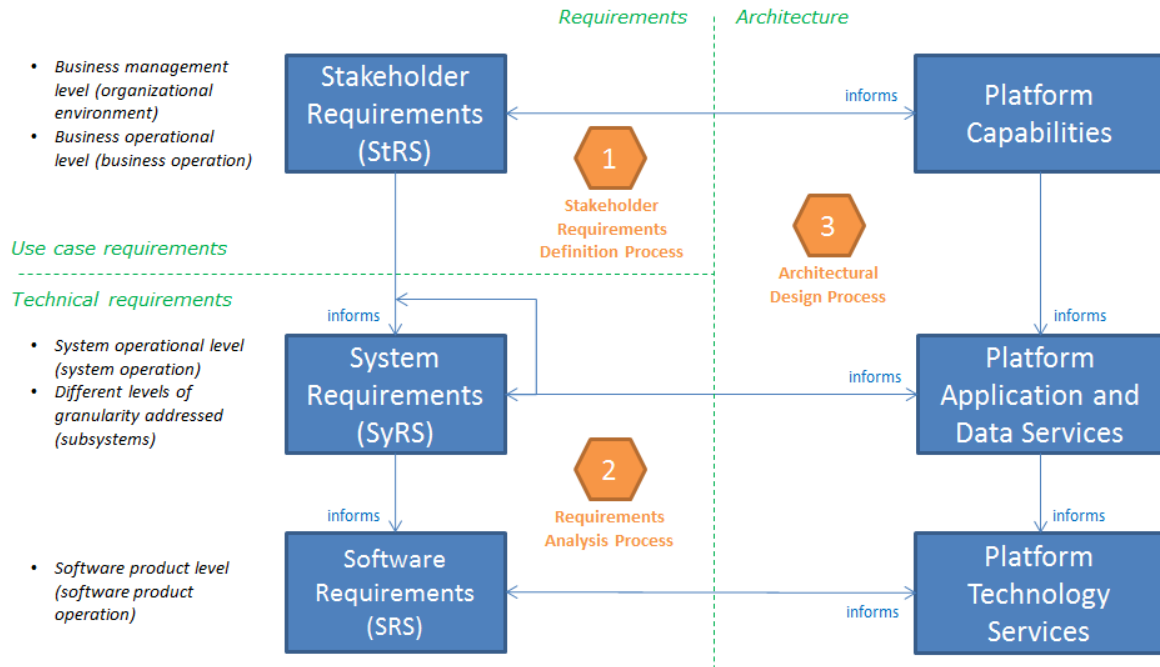


Figure 1. Requirements engineering method following ISO/IEC/IEEE 29148:2011 [41]

The work products are requirements specifications at three levels of detail, which serve as input to different practices or stages in the architectural design process. The following table describes each of those levels (extracted from ISO/IEC/IEEE 29148:2011 [41]), including the architecture domain whose decisions are informed by them.

Work product	Acronym	Description	Informed architecture domain
Stakeholder Requirements Specification	StRS	It identifies stakeholders, or stakeholder classes, involved with the system throughout its life cycle, and their needs, expectations, and desires. It analyses and transforms these into a common set of stakeholder requirements that express the intended interaction the system will have with its operational environment and that are the reference against which each resulting operational service is validated. It specifies: <ul style="list-style-type: none"> A. The required system characteristics and context of use of the product (platform) business functions and services, and operational concepts are specified. B. The constraints on a system solution are defined. C. Traceability of stakeholder requirements to stakeholders and their needs is achieved. 	Platform Capabilities (business architecture)

		<p>D. The stakeholder requirements are defined from the stakeholder’s perspective.</p> <p>E. Stakeholder requirements for validation are identified.</p>	
System Requirements Specification	SyRS	<p>Technical specifications for the selected system of-interest and usability for the envisaged human-system interaction. It characterises system requirements because:</p> <ul style="list-style-type: none"> A. It represents a system (including interfaces of functions and services) that will meet stakeholder requirements. B. Allows lower levels of granularity (recursion), i.e., subsystems. C. Does not imply any specific implementation. <p>It specifies:</p> <ul style="list-style-type: none"> A. The future system requirements from the domain perspective, background information about the overall objectives for the system, and its target environment. B. A statement of the constraints, assumptions and non-functional requirements. C. Measurable system requirements specifying, from the supplier's perspective, what characteristics and with what magnitude it is to possess to satisfy stakeholder requirements. 	Platform Applications and Data Services Architecture
Software Requirements Specification	SRS	<p>A specification for a software product, program, or set of programs) that performs certain functions in a specific environment.</p> <p>The SRS may be written by one or more representatives of the supplier, one or more representatives of the acquirer, or by both.</p> <p>Typically,</p> <ul style="list-style-type: none"> A. there will be a requirement specification that will state the interfaces between the system and a software portion; B. it will place external performance as well as functionality requirements upon the software portion; C. it defines all the required features (e.g., functions) of the specified software product to which it applies; and D. it documents the conditions and constraints under which the software portion must perform, and the intended verification approaches for the requirements. 	Platform Technology Architecture

Table 2 – Levels of requirement specification as defined in ISO/IEC/IEEE 29148:2011 [41]

Finally, to identify requirements from all stakeholders’ point of view, we have taken inspiration from the *TOGAF® Series Guide¹: Business Scenarios* method to shed light on the key business requirements and indicate the implied technical requirements for the BigDataStack architecture. This is a technique to validate, elaborate, and/or change the premise behind an architecture effort by understanding and documenting the key elements of a Business Scenario in successive iterations.

Finally, to better formalize the requirements, we use the following attributes:

- **Level of detail:** Following the use of ISO/IEC/IEEE 29148:2011 (see Section 2.1 Methodology), we use the following levels: Stakeholder, System and Software (i.e., technology details).
- **Type:** Types of requirements are functional: FUNC (function), DATA (data); and non-functional: L&F (Look and Feel), USE (usability), PERF (performance), ENV (operational environment), SUP (maintainability and support), and SEC (security and privacy).
- **Priority:** Requirements can have different priorities: MAN (mandatory), DES (desirable), OPT (optional), ENH (possible future enhancement).

2.2 Organization

In this document, we firstly describe the stakeholder requirements of the **User Enterprises** (see Table 3 – Stakeholders) to which the use cases belong. This is done through business scenarios which represent business problems for the customer organization of a developed Big Data solution (see Section 4 - Business Requirements and Scenarios).

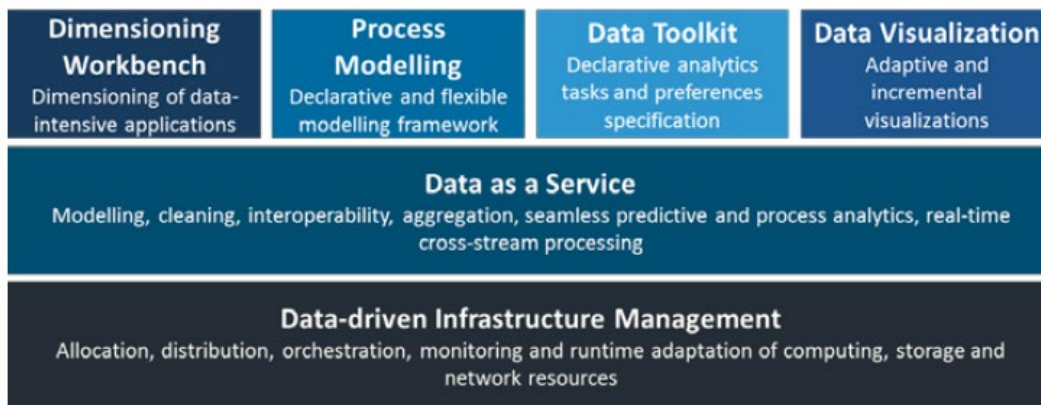


Figure 2. BigDataStack core platform capabilities

Secondly, we present the rest of stakeholder requirements as well as the system and technology (software) requirements organized in terms of the envisioned BigDataStack platform capabilities (see Figure 2), detailed as follows:

- **Data-driven Infrastructure Management.** The platform capability to provide means for efficient and optimized infrastructure, incorporating all aspects of data-driven management for the computing, storage and networking resources.
- **Data-as-a-Service (DaaS).** The platform capability to exploit the underlying data-driven infrastructure management system to offer data as a service in a performant, efficient and scalable way. It includes access to a set of technologies addressing the complete data path: modelling and representation, cleaning, aggregation, and data processing

¹ <https://publications.opengroup.org/g176>

and analytics.

- **Dimensioning, Modelling and Interaction Services:**

- Data Visualization goes beyond adaptable visualization and presentation of data and analytics outcomes, to performance aspects such as computing, storage and networking infrastructure data, data sources information, and data operations outcomes.
- Data Toolkit aims at openness, extensibility and wide adoption. The toolkit allows the ingestion of data analytics functions and the definition of analytics in a declarative way; moreover, it allows data scientists and administrators to specify requirements and preferences both for the data, the analytic functions parameters and infrastructure management.
- Process Modelling allows for declarative and flexible modelling of process analytics. Functionality-based process modelling can then be concretized to technical-level process mining analytics, while a feedback loop will be implemented towards overall process optimization and adaptation.
- Dimensioning Workbench enables the self-dimensioning of applications in terms of predicting the required data services, their interdependencies with the application micro-services and the required underlying resources.

3 Business Stakeholders and Goals

This section aims to identify the business goals to address in the elicitation of requirements and architecture specification of BigDataStack results. The identification of business needs and requirements will help to implement a solution that meets stakeholders’ expectations and allows a better market positioning for a future exploitation.

It is worth noting that a preliminary wide-reaching market analysis was delivered at M18² of the project, which is used to confirm that the business goals envisioned in this phase of the project, as well as the business model and stakeholders, are valid.

3.1 Stakeholder Categories

As defined in **BDVA SRIA Agenda**³, the following key stakeholders are the main categories of actors along the Big Data Value chain: *User Enterprises, Data Generators and Providers, Technology Providers* and *Service Providers*. These categories are described in the following table, including the BigDataStack platform “side” they will play in: supply versus demand, or solution provider versus consumer.

Id	Name	Side	Description
STA-01	User Enterprises	Demand side	These are, for example, enterprises in all domains and of all size that want to improve their portfolio using Big Data technology.
STA-02	Data Generators and Providers	Supply side	Create, collect, aggregate, transform and model raw data from heterogeneous sources and offer it to customers.
STA-03	Technology Providers	Supply side	Provide tools and/or platforms that offer data management and analytics tools to extract knowledge from data, curate and visualize it.
STA-04	Service Providers	Supply side	Develop Big Data applications on top of the tools and platforms to provide services to user enterprises.

Table 3 – Stakeholders

In the BDVA SRIA Agenda and in many digital media, workshops, congresses, etc., the big data stakeholder ecosystem has concerns about problems like those of the main BigDataStack stakeholders, which may cause a slower uptake of big data applications and solutions.

Stakeholder Categories	Concern	Description
STA-01 STA-02 STA-03 STA-04	Privacy and Security	Potential data users are worried about privacy and security of their data. Due to velocity and volume, different data locations and different type of data (including not only personal data, but also sensitive business data), a robust data protection mechanism is needed. For businesses this is a key point, since 89% of companies avoid doing business

² The result of this preliminary wide-reaching Market Analysis is included in the deliverable **D7.2 - Exploitation plan and business potential** [ATOS, Report, Public, M18].

³ http://www.bdva.eu/sites/default/files/EuropeanBigDataValuePartnership_SRIA_v3.pdf

		with companies that they believe do not protect their privacy ⁴ .
STA-02	Cost	The high data volume and quick scalability of big data projects make difficult to foresee of cost management for enterprises. New provider monetization models are emerging to create innovative cost-effective solutions for big data users to control costs as far as possible.
STA-02	Integration with existing systems	The integration of big data technologies with existing systems is a main question for companies planning to implement big data solutions. Companies know that changing operational/process company systems is a major issue since it leads additional costs, new personal training, etc. But the challenge is not only before the big data implementation, since companies must be prepared to make necessary changes to derive business value from big data, which probably will lead to changes in existing systems.
STA-03	Scalability and performance	Nowadays, companies are increasingly using big data in their business and must deal with a large amount of data. Service providers have to offer attractive cost-effective services to their clients to address this problem.
STA-02 STA-03	Heterogeneity of data and data sources	The emergence of IoT has incorporated a new type of data with different existing ones: data-in-flight from sensors, mobiles, etc. which needs a new management data model and capabilities
STA-03 STA-04	Different analytics capabilities	A key challenge for business is to identify clear business objectives, and this will not be the same for the different sectors, so application service providers need to develop different analytics capabilities to address clients in all domains
STA-03 STA-04	Lack of talent	There are not enough skilled people and new training requires time and money, so providers need big data tools ease to use, to deliver new services in a short time to market.

Table 4 – Stakeholder concerns

3.2 Business Model

To meet the needs of the stakeholder ecosystem, the BigDataStack platform should support whole Big Data management and analytics products and services, addressing needs of data operations and data applications in a Data-as-a-Service (DaaS) model. The table below depicts the envisioned BigDataStack platform value proposition for customers at each side of it (demand and supply sides) as well as the revenue model proposed for them⁵:

⁴ <https://www2.deloitte.com/content/dam/Deloitte/ca/Documents/Analytics/ca-en-analytics-ipc-big-data.pdf>

⁵ These assumptions will be deeply explored in the Market Study (*D7.2. Exploitation plan and business potential*) and a best-suited business model will be deployed based of the market analysis result.

Products and Services	Revenue model		Customer
Turn-Key Big Data management and analytics solutions	Pay-as-you-go	Demand side	Enterprises of all sizes and all sectors that want to increase the knowledge or operational efficiency of their business and/or enhance their business offering by using Big Data Analytics and wish a whole outsourcing solution for the management of the data path operation.
Development of different Big Data management and analytics solutions	Pay-as-you-go	Supply side	Solution Providers who want to make use of BigDataStack tools to enhance their Big Data products and services, including <i>technology</i> , <i>applications</i> and <i>data</i> offerings.

Table 5 – Preliminary business model

3.3 Business Outcomes

In the proposal stage of the project, a deeper study of the main actors and stakeholders related to BigDataStack solutions was carried out. That study has been enhanced and is summarized in the following table, where the benefits for each stakeholder of using the BigDataStack results are included:

Side	Stakeholder Category	Stakeholder	Description
Supply side	STA-03	Infrastructure providers	Offer infrastructure solutions to big data needs through efficient and performant management of all resources.
	STA-02	Data providers	Offer cleaned, modelled, stored and analysed data.
	STA-04	Application providers	Provide data-intensive applications with guarantees.
	STA-03	Data practitioners	Develop enhanced algorithms and offer them.
Supply side	STA-03	Infrastructure brokers	Act as second-level entities that take advantage of the BigDataStack data-driven infrastructure management solutions from infrastructure providers.
	STA-02	Data aggregators and data resellers	Act as second-level entities (following data providers) that take advantage of the monetization model of Data as a Service according to their business models and goals.
	STA-04	Marketplace owners	Act as second-step entities that take advantage of data-intensive application provisioning by application providers.
Demand side	STA-01	Citizens	Use applications, services and products with guaranteed levels of quality.

	STA-01	SMEs and big industries	Satisfy their internal data needs to develop new offering and/or streamline operations by utilizing BigDataStack services offered by application and data providers.
	STA-01	Public organizations	Using BigDataStack for handling data.
	STA-01	Entrepreneurs	Developing, deploying and using data-intensive and/or data-driven applications to power their products or services by utilizing BigDataStack services offered by technology and data providers.
	STA-01	Decision makers	Driving business decisions based on accurate, timely, meaningful data and analytic insights.

Table 6 – Stakeholder requirements

3.4 Business Goals

Business goals are often called “vision requirements.” These are top-level requirements appearing first, to which the rest of requirements are subordinated to ensure a successful market uptake. In this stage of the project, the following business goals have been identified:

Field	Description
Id	BG1
Short Name	Privacy and Security
Description	BigDataStack will propose an architecture that enables security and privacy aspects and will be oriented toward the compliance with data protection regulations.
Rationale	Ensure the protection of personal data and business data.
Involved Stakeholders	All stakeholders

Table 7 – Privacy and security (business goal)

Field	Description
Id	BG2
Short Name	Attractive revenue and business model
Description	BigDataStack envisions a Pay-as-you-go as revenue model, delivering a cost-effective service for different costumers and looking for strong marketplace positioning.
Rationale	Deliver cost-effective solutions for the whole stakeholder ecosystem as Data as a Service solution.
Involved Stakeholders	All stakeholders

Table 8 – Attractive revenue and business model (business goal)

Field	Description
Id	BG3
Short Name	High performance, scalability and sharing
Description	BigDataStack will introduce an architecture that enables real-time data-driven management decisions and will provide a performant, scalable, flexible and dependable environment for the efficient delivery of distributed data operations, data- and storage- intensive applications. The performance and optimization will be achieved by basing all infrastructure management decisions on the data aspects.
Rationale	Data management of different data from several sources, including data at rest and in flight.
Involved Stakeholders	<i>Infrastructure providers, Data providers, Application providers, Data practitioners, Citizens, SMEs and Large industries, Public Organisations, Entrepreneurs, Decision makers.</i>

Table 9 – High performance, scalability and sharing (business goal)

Field	Description
Id	BG4
Short Name	Product integration
Description	BigDataStack offers a solution catalogue for providers, which can be used to manage the complete data path or only to address parts of a provider’s whole solution. For end users, BigDataStack-based turn-key solutions will facilitate the integration of analytics in their businesses.
Rationale	Integration with other systems in end user companies and with other analytic tools for providers.
Involved Stakeholders	All stakeholders

Table 10 – Product integration (business goal)

Field	Description
Id	BG5
Short Name	Different analytic capabilities
Description	BigDataStack will validate its solutions in three commercial cases in the maritime, market and financing domains; this will provide a key expertise to BigDataStack to offer guaranteed turn-key big data solutions in other domains.
Rationale	Deliver successful solutions for major challenges in main sectors.
Involved Stakeholders	<i>Citizens, SMEs and Large Industries, Public Organisations, Entrepreneurs, Decision makers.</i>

Table 11 – Different analytic capabilities (business goal)

Field	Description
Id	BG6
Short Name	Ease of use

Description	BigDataStack will put emphasis on usability through data toolkits and visualization environments. Its solutions will include mechanisms for deployed data path operations to become faster.
Rationale	Reduce time to market and cost for new data applications.
Involved Stakeholders	<i>Infrastructure providers, Data providers, Application providers, Data practitioners.</i>

Table 12 – Ease of use (business goal)

4 Business Requirements and Scenarios

This section presents the business usage scenarios and initial requirements elicited from each of the three business use cases of the BigDataStack project. These requirements should be considered as **Stakeholder Requirements** focused on specific solutions as required by specific **User Enterprises** (see Table 3 – Stakeholders). The business scenarios are representative of a significant business need or problem, and enables *data, technology and service providers* to understand the value to the customer organization of a developed Big Data solution.

Each scenario describes the different usage from a use case perspective at a high-level. It is not the intention to define the complete and detailed set of BigDataStack solution scenarios, but to have a wide range of stakeholder behaviour descriptions to guide the technical requirements analysis and architecture work. Scenario descriptions are complemented with UML Use Case Diagrams to identify the different actors, prerequisites and the description of the behaviour.

Each use case can identify one or more scenarios depending on the complexity or the scope of the definition. For instance, on one side, the necessity for the analysis of the data services and data-intensiveness of the provision (at the dimensioning phase), and on the other side, the scenario for the operational phase where the defined Quality of Service (QoS) and rules should be applied. Thus, this can be described only in one scenario (more complex) or can be split into two scenarios differentiating clearly the objectives, the behaviour and the actors. It should be the decision of each use case provider to take the approach that best suits their purpose.

4.1 Real-time Ship Management

4.1.1 Introduction

This section refers to the use case of Real-time Ship Management (RSM): Maintenance and spare parts inventory planning & dynamic routing. The usage scenarios, that is, a higher-level representation of functional requirements (Section 3.1.2), along with a detailed description of use cases (Section 3.1.3) will be presented.

4.1.2 Scenarios

This case addresses two key challenges in the ship management domain: (i) predictive maintenance combined with spare parts inventory planning, and (ii) dynamic routing. In recent years, increasing fuel prices, depressed market conditions and environmental issues such as emissions from ships, have brought a new perspective to ship routing. Besides being cost-efficient, a ship also must be environmentally friendly with regards to its emissions.

One of the project partners faces similar challenges: DANAOS - a leading international maritime player with more than 60 containerships, transports millions of containers, sails millions of miles to thousands of ports, and consumes millions of tons of fuel oil. Each year, DANAOS senior management, investors, and customers evaluate performance after each voyage and demand the highest level of operational quality. Ship engines and other relevant machinery need to achieve high availability not only to deliver transport services (and thus ensure availability of resources) but also for operational safety, occupational health and environmental impact purposes. High availability of ship engines and machines can only be achieved if they are kept under proper conditions using applicable maintenance strategies,

thus the monitoring of machinery has become even more critical to meet the maintenance requirements and achieve predictive maintenance. The latter is based on data that are exploited to estimate the type of failure and time to failure.

An additional problem is the limited availability of spare parts, resulting in expediting inbound replenishment shipments. If the spare parts planning and inventory management processes cannot cope with the unpredictability of the need for parts, then the operation may be starved of critical parts, yet may be flooded with other parts which are not frequently required, resulting in lower productivity due to additional downtime and higher holding costs due to excess inventory, respectively. Spare parts inventory management in relation to maintenance is a complex process because it involves hundreds of parts for a single engine, some of which may have a high level of demand per month whereas some may have a demand of few units per year.

We discuss two different but complementary scenarios: (i) **monitoring and predictive maintenance** and (ii) **requisition of a spare part and dynamic routing to the closest port where this part is available**. The following tables describe in more detail these two scenarios.

Section	Description
Id	SCE-RSM-01
Title	Monitoring and predictive maintenance
Description	A vessel must complete its route within a time-frame. When a part of the main engine fails unexpectedly, the ship risks staying off-hire. This can be very damaging to a shipping company, as chartering revenues decrease, while replacing a spare part immediately increases cost. Thus, identification of potential failure allows timely ordering, or even replacement of spare parts before failure. The main engine, posing the highest risk, consists of various spare parts depending on many parameters. Thus, it is difficult to accurately predict failures. If false alarms occur, the operating costs increase, as ordering of unnecessary parts is not optimal.
Actors	Coordinator, Fleet manager
Objectives	<ul style="list-style-type: none"> - Monitoring the main engine of a vessel. - Notification for an upcoming malfunction. - Minimization of machinery failures that cause the ship to go off-hire.
Pre-conditions	Monitoring and predictive maintenance of the main engine takes for granted a full-scale dataset with measurements from the main engine, along with other factors that may influence the performance of the main engine, such as weather conditions, hull condition, power consumption, fuel quality etc. Furthermore, a recorded history of malfunctions is required.
Process Description	Shipping companies nowadays work preventively against malfunctions via a planned maintenance scheme and a condition-based maintenance scheme. Planned maintenance is performed with the help of a planned maintenance system (PMS) that informs the engineers for actions to be taken for maintenance from a main-engine component down to a spare-part-level. For example, the change of lube oils or a piston component after a defined period. Condition-based maintenance is performed either separately from planned maintenance via a pure human decision and interaction scheme, or can be included in the PMS. For example, if the tubes of the air-cooler have been cleaned, the air-filter should be replaced.

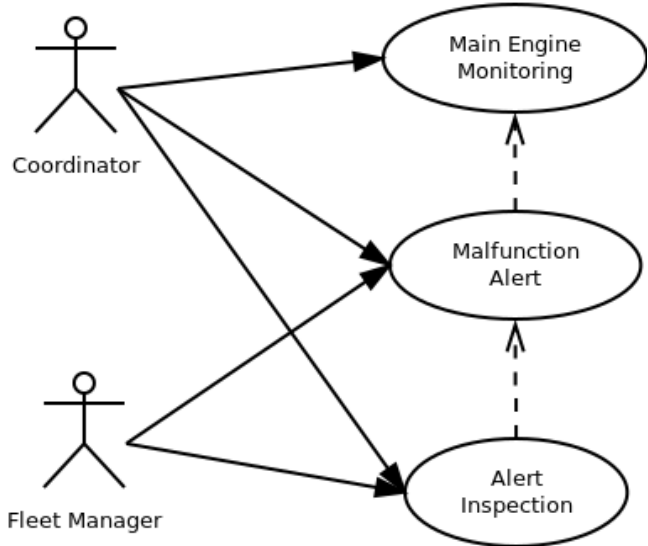
Variations	In this case, preventive maintenance is addressed as the remaining cases of maintenance that are not included in a condition-based or preventive maintenance scheme. If these two categories are excluded, we discuss about malfunctions that occur unexpectedly, thus should be handled in a different manner.
Post-condition	If a malfunction pattern is identified, the actor is informed via an alert.
Diagrams	 <pre> graph TD C[Coordinator] --> MM(Main Engine Monitoring) C --> MA(Malfunction Alert) C --> AI(Alert Inspection) FM[Fleet Manager] --> MM FM --> MA FM --> AI MA -.-> MM AI -.-> MA </pre>

Table 13 – Monitoring and predictive maintenance scenario description (scenario)

Section	Description
Id	SCE-RSM-02
Title	Requisition and dynamic routing
Description	Once a malfunction is identified and the technical department is informed (Fleet manager, coordinator), spare parts or actions to be taken for maintenance should be clarified from the technical department to the supplies department. The supply department should order the required spare part and proceed with the requisition and delivery process of the part to the vessel. The cost of the spare part depends on the location of the vessel, on the distance where the closest port is, and on the supplier, while some qualitative criteria must be taken also into account. Usually, each shipping company has a list of suppliers who are trusted. Thus, the supply department wishes to minimize the cost of the ordered spare part without compromising the quality of the part itself and replace it on time without letting the damage on the main engine put the vessel off-hire.
Actors	Coordinator, Fleet manager, Supplies coordinator
Objectives	<ul style="list-style-type: none"> - Timely alerting of the supply department for a new order. - Timely ordering of spare parts. - Dynamic routing of the vessel to the closest port with available spare part. - Optimization of the requisition and delivery process.
Pre-conditions	A malfunction has been identified, the technical department is alerted

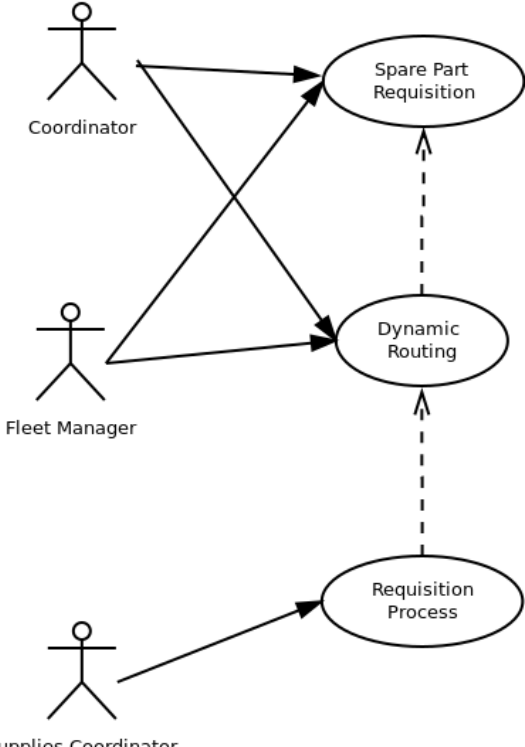
Process Description	The requisition process of a spare part goes as follows; First a requisition is made by the vessel. This request is processed and pre-checked by the supply department. Next, a Request for Quotation (RFQ) is made via the DANAOS platform , which broadcasts the RFQ to the company’s listed suppliers. Given the offers by the suppliers, the supply department performs a comparative table analysis and decides which supplier will place the order. Once the order is placed, it is invoiced and delivered to the vessel.
Variations	An order may be delivered but not invoiced on time.
Post-condition	The required spare part is ordered and delivery is expected to the closest port where the vessel is dynamically routed.
Diagrams	 <pre> graph TD C[Coordinator] --> SR(Spare Part Requisition) C --> DR(Dynamic Routing) FM[Fleet Manager] --> SR FM --> DR SC[Supplies Coordinator] --> RP(Requisition Process) RP -.-> DR DR -.-> SR </pre>

Table 14 – Order suggestion and dynamic routing (scenario)

4.1.3 Requirements

In the following tables, we show the description of the use requirements defined in each scenario, as described in the previous section.

Section	Description
Id	REQ-RSM-01
Level of detail	Stakeholder
Type	FUNC
Short name	Main Engine Monitoring
Description	As data flow, out of sensors installed at the main engine of a vessel, the user wishes to have a look on the current or past values of the provided metrics in a chart, select a set of metrics which are to be drawn on a chart

Additional Information	<p>Data requirements (indicative):</p> <ul style="list-style-type: none"> - Air Cooler Cooling Water Inlet Pressure (Pa) - Air Cooler Cooling Water Inlet Temperature (°C) - Cooling Fresh Water Inlet Pressure (Pa) - Control Air Pressure (Pa) - Cylinder Lube Oil Temperature (°C) - Exhaust Valve Spring Air Inlet Pressure (Pa) - Fuel Oil Flowrate (lt) - Fuel Oil Inlet Pressure (Pa) - Fuel Oil Inlet Temperature (°C) - Heavy Fuel Oil Viscosity High Low (mm²/s) - HPS Bearing Temperature (°C) - Jacket Cooling Fresh Water Inlet Temperature Low (°C) - Order RPM (Bridge Leverer) - Scavenge Air Inlet Pressure (Pa) - Scavenge Air Receiver Temperature (°C) - Starting Air Pressure (Pa) - Thrust Pad Temperature (°C) - Main Lube Oil Inlet Pressure (Pa) - Main Lube Oil Inlet Temperature (°C) - Fuel Oil Temperature (°C) - Fuel Oil Total Volume (lt) - Power (kW) - Scavenge Air Pressure (Pa) - Torque (N/m) - Fuel Oil Consumption (lt/min) - Fuel Oil Consumption (MT)
Actor	Coordinator
Priority	MAN

Table 15 – Main Engine Monitoring (stakeholder requirement)

Section	Description
Id	REQ-RSM-02
Level of detail	Stakeholder
Type	FUNC
Short name	Malfunction Alert
Description	Once a malfunction pattern is identified the user is alerted via a message with minimum and concise information about the upcoming malfunction
Additional Information	<p>Data requirements:</p> <ul style="list-style-type: none"> - Malfunction name - Estimated time before break-down
Actor	Coordinator, Fleet manager

Priority	MAN
-----------------	-----

Table 16 – Malfunction alert (stakeholder requirement)

Section	Description
Id	REQ-RSM-03
Level of detail	Stakeholder
Type	FUNC
Short name	Alert Inspection
Description	Once the user is informed about an alert, he/she can investigate the malfunction pattern, the metrics and the history of this malfunction.
Additional Information	Data requirements: <ul style="list-style-type: none"> - Malfunction name - Estimated time before break-down - Previous occurrence of this malfunction - Actions taken in previous occurrence (e.g. ordered spare part) - Chart with values and anomalies on a minute basis
Actor	Coordinator, Fleet manager
Priority	MAN

Table 17 – Alert Inspection (stakeholder requirement)

Section	Description
Id	REQ-RSM-04
Level of detail	Stakeholder
Type	FUNC
Short name	Spare Part Requisition
Description	Once the Coordinator or the Fleet manager have inspected an alert for an upcoming malfunction, given the history of actions taken in the past, he/she makes a requisition for the same or another spare part of the main engine.
Additional Information	Data requirements: <ul style="list-style-type: none"> - Spare part name - Spare part id - Reason for requisition - Date of requisition - Description
Actor	Coordinator, Fleet manager
Priority	ENH

Table 18 – Spare Part Requisition (stakeholder requirement)

Section		Description
Id	REQ-RSM-05	
Level of detail	Stakeholder	
Type	FUNC	
Short name	Requisition Process	
Description	Once a requisition is made by the technical department, it is processed and pre-checked by the supply department. Next, a Request for Quotation (RFQ) is made via the <i>DANAOS</i> platform, which broadcasts the RFQ to the company's listed suppliers. Given the offers by the suppliers, the supply department performs a comparative table analysis and decides to which supplier will place the order. Once the order is placed, it is invoiced and delivered to the vessel.	
Additional Information	Data requirements: <ul style="list-style-type: none"> - Spare part id - Spare part name - Spare part description - List of suppliers - List of offers - List of available ports - List of estimated time of deliveries 	
Actor	Supplies Coordinator	
Priority	ENH	

Table 19 – Requisition Process (stakeholder requirement)

Section		Description
Id	REQ-RSM-06	
Level of detail	Stakeholder	
Type	FUNC	
Short name	Dynamic Routing	
Description	This use case is a plug-in feature for the requisition process use case, since it can update the data on the comparative table analysis where costs of routing to the port where the spare part is available are included. Furthermore, it can highlight the row in the comparative table the cost-efficient solution, to suggest to the Supplies coordinator the best possible choice.	
Additional Information	Data requirements: <ul style="list-style-type: none"> - Spare part id - Spare part name - Spare part description - List of suppliers - List of offers 	

	<ul style="list-style-type: none"> - List of available ports - List of estimated time of deliveries - Voyage estimations to closest ports
Actor	Coordinator, Fleet manager, Supplies Coordinator
Priority	ENH

Table 20 – Dynamic Routing (stakeholder requirement)

4.2 Connected Consumer

4.2.1 Introduction

This section refers to the use case of Connected Consumer (CC): Multi-sided market ecosystem. Here, we discuss the usage scenarios by giving a detailed description of the use cases (Section 4.2.2) along with a description of use requirements (Section 4.2.3).

In a world with instant access to information, where competition is just one click away, attracting and keeping customers is crucial for survival. Predictive analysis is the challenge. It can help predict which consumers are the most loyal or which potential buyers are more likely to purchase a certain product or service, opening new opportunities for retailers, providing new business prospects to customers, with improved shopping experience for consumers and new business opportunities for traders.

In this business domain, *Eroski*⁶, one of the largest distribution companies in Spain with more than 35.000 workers, is collaborating with ATOS in the definition and test of a use-case related to the grocery business. It is also contributing with real data for the development of the project. The goal of this scenario is to provide data insights to EROSKI to better understand how to create and offer added-value services to their consumers. In this context, the use case objective is to predict both which products and which promotions are more likely to be interesting for the customers at the right time. In this way, EROSKI can adapt the most appropriate message (i.e. product and/or promotion) for each customer and send it at the right time and through the most appropriate channel, thus increasing the ROI of their marketing activities.

From the analysis of different data sources provided by *Eroski*, the goal is first to predict the list of products that customers with recurrent purchases will need in the current purchase period (trend). Afterwards, add to this prediction those products that can be interesting for the user based on other similar user's behaviour (cross-selling). Finally, thanks to a deep knowledge of the customer profile, the goal is also to incorporate those promotions that can be interesting for each customer.

Additionally, a scenario that describes a demonstrator that will help users to display and test recommendations made by the user has also been included.

4.2.2 Scenarios

This case addresses two key challenges in the retail customer services. These include: a) the support of personalised recommendations (i.e. products, discounts, combined special offers, etc.) to customers with focus in associating products according to their customer profile and

⁶ <https://www.eroski.es/>

purchase history profile; and b) the timely provision of those to visitors of the of Eroski's e-commerce service. The following tables describe in more detail these business scenarios.

Section	Description
Id	SCE-CC-01
Title	Retail Recommender
Description	<p>This scenario is distributed in three steps:</p> <ul style="list-style-type: none"> - data collection, - calculate recommendations, and - show predictions. <p>The first step, data collection, provides services to update those entities needed for the recommender with data coming from external systems:</p> <ol style="list-style-type: none"> a) Product Service (products, categories, products x category) b) Sales Service (orders) c) Client service d) Events service (used by the external systems to provide feedback about the visualization of the recommendations) <p>The second step, calculate recommendations, calculates the products and the promotions that would recommend to every user. The input data is being processed, i.e., cleaned (“denoised”) and modelled. In the cleaning process, any unwanted effects in the data are removed (such as missing values or outliers) while maximizing its information. We define noise as any unwanted artefact introduced in the data collection phase that might affect the result of our data analysis and interpretation. In the modelling process, the data is being modelled into some pre-defined model. The pre-defined model is going to be the input for the main process.</p> <p>Therefore, we can split the recommendation process in two phases:</p> <ul style="list-style-type: none"> - Calculate the products that a user would be interested to buy, based on: <ol style="list-style-type: none"> 1. Product sales frequency by user 2. Product sales frequency by product 3. Product seasonality 4. Product cross-selling 5. User feedback (registered with events) - Calculate the promotions that will be recommended to the final users. The promotions are calculated in base to: <ul style="list-style-type: none"> - Representative products that the user could buy (calculated in “calculate products” use case) - Possible promotions, with a priority ranking - User feedback (registered with events) <p>Finally, the third step, show predictions, provides the needed services for getting the recommendations calculated. Consumers of these services will be the client applications that need to show recommended products or promotions to its users.</p>

Actors	External System, Trigger recommender
Objectives	<p>The main objective of this scenario is to calculate the most interesting products and promotions to recommend.</p> <p>To achieve this main objective, the first thing we need is to collect data and refresh the database with fresh data, including: users, products, promotions and sales data. When we have all the new fresh data collected and stored, we need to process data and prepare it for the main process, denoising and modelling. Then, with the modelled data, we calculate the most interesting products and promotions for every user. Finally, when it's requested, we provide data to client applications with the recommendations requested.</p>
Pre-conditions	<p>The system needs, at least, 2 years of data history to do good predictive recommendations. This data includes:</p> <ol style="list-style-type: none"> 1) Sales 2) Clients 3) Products 4) Categories <p>Additionally, we can also have some other data that can be included in the recommendation process, such as user events, user feedback, etc.</p>
Process Description	<ul style="list-style-type: none"> - Data collection <ul style="list-style-type: none"> - External system invokes service - System stores the data provided by the external system - Calculate recommendations <ul style="list-style-type: none"> - Some input data arrives to the system - The system prepares the data pro process it - The system calculates products to recommend - The system calculates promotions to recommend - Process the results from points 3 and 4 and store the final products and promotions to recommend on DB - Data preparation <ul style="list-style-type: none"> - Some input data arrives to the system - The system initiates a denoising process - The system initiates a modelling process - Denoising data <ul style="list-style-type: none"> - The process for denoising is called with some data - The data noise is removed - Denoised data is returned to the caller - Data modelling <ul style="list-style-type: none"> - The process for modelling is called with some data - The data is modelled - Modelled data is returned to the caller - Calculate products <ul style="list-style-type: none"> - The process of calculate products is called with some data - The system calculates recommended products for every user - Calculate promotions <ul style="list-style-type: none"> - The process of calculate promotions is called with some data.

	<ul style="list-style-type: none"> - If the data does not include the suggested products calculated, the system calculates recommended products for every user. - The system calculates recommended promotions. - Show Predictions <ul style="list-style-type: none"> - External system invokes service - System provides the predictions requested by the external system
Variations	N/A
Post-condition	<p>The suggested products and promotions shouldn't be null.</p> <p>The system must store in the DB:</p> <ul style="list-style-type: none"> - Products and promotions being recommended for every user. - Suggested order priority for the recommended products and promotions for every user. - Trace tokens to register possible feedback events for every recommended item.
Diagrams	

Table 21 – Retail Recommender (scenario)

Section	Description
Id	SCE-CC-02
Title	Retail Demonstrator
Description	<p>This scenario is distributed in three modules: login, View Predicted Products and View Predicted Promotions.</p> <ul style="list-style-type: none"> - The login module logs into the demonstrator application for a given customer. - The View Predicted Products module displays the products that the system would suggest to the customer. - The View Predicted Promotions module displays the personalized promotions that the system would suggest to the customer.

	In both modules, the View Predicted Products and View Predicted Promotions, the demonstrator is also giving feedback to the retail recommender about in which products/promotions the customer has shown interest so that the recommender can adapt its recommendations in real time.
Actors	Customer, Products Recommender
Objectives	<ul style="list-style-type: none"> - Provide a way to switch from one user to another in the app, thus, allowing display of the predicted products and promotions for different users. - Provide an example on how end users could display the products calculated by the recommender. - Provide an example on how end users could display the promotions calculated by the recommender. - Provide a way to show that the recommender is considering the feedback given by the client applications.
Pre-conditions	User is in the list of available users
Process Description	<ol style="list-style-type: none"> 1. Display predicted products <ul style="list-style-type: none"> - User logs into the system - System verify username exists - User selects <i>My predicted products</i> - System retrieves the prediction for the current user from the recommender system - Application displays Suggested Products list - Application gives feedback to the recommender about both which products has been shown and which products the user has shown interest 2. Display predicted promotions <ul style="list-style-type: none"> - User is logged in the application - User selects 'My predicted promotions' - System retrieves the prediction for the current user from the recommender system - Application displays a suggested promotions list which takes into consideration the list of products predicted for the user - Application give feedback to the recommender about both which promotions has been shown and which promotions the user has shown interest
Variations	N/A
Post-condition	<p>The user is logged in the demonstrator, and the user can view the following in the application:</p> <ul style="list-style-type: none"> - List of products predicted for the user - List of promotions predicted for the user
Diagrams	

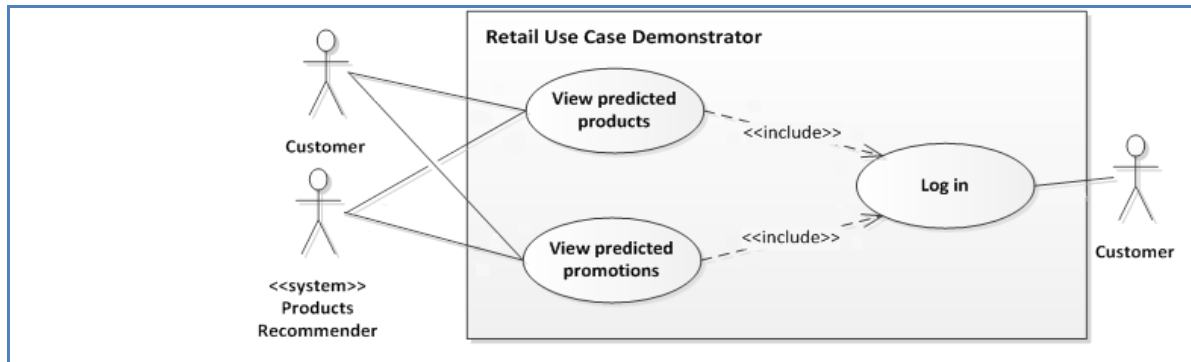


Table 22 – Retail Demonstrator (scenario)

4.2.3 Requirements

In the following tables, we show the description of the use requirements defined in each scenario, as described in the previous section.

Section	Description
Id	REQ-CC-01
Level of detail	Stakeholder
Type	FUNC
Short name	Predict products required by a recurrent user
Description	<p>For a user with previous orders (recurrent), the system should be able to predict a list of items that the user is likely to need in the coming days. The calculation should consider:</p> <ul style="list-style-type: none"> - History of orders made by the user - Seasonality of the products - Similarity with items that the customer bought and is bound to need - What other customers bought - User segment - Receptivity of the user to the items recommended by the system
Additional Information	<p>The list of items should return for each item a rank that helped the external system to prioritize the display of items. The rank should be assigned to the products considering the probability that the user needs them, that is, buys them.</p>
Priority	MAN

Table 23 – Predict products required by a recurrent user (stakeholder requirement)

Section	Description
Id	REQ-CC-02
Level of detail	Stakeholder
Type	FUNC
Short name	Predict products to a new user (user without previous orders)

Description	For a given new user, the system should be able to predict a list of items that the user is likely to buy. The calculation should consider: <ul style="list-style-type: none"> - Seasonality of the products - What other customers bought - User segment
Additional Information	The list of items should return for each item a rank that helps the external system prioritize the display of items. The rank should be assigned to the products considering the probability that the user buys them.
Priority	MAN

Table 24 – Predict products to a new user (stakeholder requirement)

Section	Description
Id	REQ-CC-03
Level of detail	Stakeholder
Type	FUNC
Short name	Recommend personalized discounts
Description	Be able to recommend personalized discounts that users are likely to use in the coming days. The calculation should take into account the following factors: <ul style="list-style-type: none"> - List of predicted items towards the user (see req-1 for further info) - Category (commercial structure) of the items predicted to the user. The list of discounts proposed by the system will contain promotions that apply on products that belong to the same category than the products predicted for the user - User receptivity to the items recommended by the system
Additional Information	The calculation should consider: <ul style="list-style-type: none"> - List of items predicted to the user, - Seasonality of the products, - Similarity with items that the customer bought/is bound to need, and - What other customers in the same segment bought, and rank the products according to the probability he will buy them
Priority	MAN

Table 25 – Recommend personalized discounts (stakeholder requirement)

Section	Description
Id	REQ-CC-04
Level of detail	Stakeholder
Type	DATA
Short name	Orders requirements

Description	New orders placed by the users should be loaded at least once per day. Orders should have at least the following information: <ul style="list-style-type: none"> - Client Id - Order Date - Items <ul style="list-style-type: none"> ○ productId ○ price ○ promotionId
Additional Information	N/A
Priority	MAN

Table 26 – Data Requirement (stakeholder requirement)

Section	Description
Id	REQ-CC-05
Level of detail	Stakeholder
Type	DATA
Short name	Clients requirements
Description	New <i>Eroski</i> customers should be loaded at least once per day. Customers should have at least the following information: <ul style="list-style-type: none"> - Client Id - Client segment
Additional Information	N/A
Priority	MAN

Table 27 – Clients requirements (stakeholder requirement)

Section	Description
Id	REQ-CC-06
Level of detail	Stakeholder
Type	DATA
Short name	Products requirements
Description	New <i>Eroski</i> products should be loaded at least once per day. Products information should have at least the following information: <ul style="list-style-type: none"> - Product reference - Product category
Additional Information	N/A
Priority	MAN

Table 28 – Products requirements (stakeholder requirement)

Section		Description
Id	REQ-CC-07	
Level of detail	Stakeholder	
Type	L&F	
Short name	Multi-device	
Description	The demonstrator application UI should adapt to different devices and displays, including mobile, to provide a proper operation of the solution and a good user experience.	
Additional Information	Give support to both Android and iOS mobile platforms.	
Priority	MAN	

Table 29 – Multi-device (stakeholder requirement)

Section		Description
Id	REQ-CC-08	
Level of detail	Stakeholder	
Type	USE	
Short name	Easy-to-use	
Description	The solution should be easy to use for people of different ages. It should follow the best practices in terms of usability.	
Additional Information	N/A	
Priority	MAN	

Table 30 – Easy-to-use (stakeholder requirement)

Section		Description
Id	REQ-CC-09	
Level of detail	Stakeholder	
Type	ENV	
Short name	Multi-user	
Description	The solution should be portable and reusable for different users.	
Additional Information	N/A	
Priority	MAN	

Table 31 – Multi-user (stakeholder requirement)

Section		Description
Id	REQ-CC-10	
Level of detail	Stakeholder	

Type	SUP
Short name	Data security
Description	Database must be securely accessible and its data must not be breached.
Additional Information	N/A
Priority	MAN

Table 32 – Data security (stakeholder requirement)

Section	Description
Id	REQ-CC-11
Level of detail	Stakeholder
Type	SUP
Short name	Services security
Description	Datasets contain personal information, so security is very important in services.
Additional Information	N/A
Priority	MAN

Table 33 – Services security (stakeholder requirement)

4.3 Smart Insurance⁷

4.3.1 Introduction

This section refers to the use case of Smart Insurance: Customers segmentation and Customer Lifetime Value (CLV) prediction. Here, we discuss the usage scenarios (Section 4.3.2) along with a high-level representation of functional requirements (Section 4.3.3).

The use case focuses on the development of a solution for Insurance companies, developing software and systems addressing the needs of such institutions based on a data-centric paradigm and addressing the provision of services according to the customer “tailored” requirements.

The main goal is to allow insurance companies that focus on customer management, to provide personalized services for their customers, as well as new corporate services for the handling of customers’ profitability and retention.

The datasets that will be used within the process of this use case is corporate data provided by an insurance company, based in Italy, selected from the GFT customers’ portfolio.

⁷ Note that with respect to deliverable D2.1, this use case has been redefined, according to a new context (insurance instead of banking) and scenarios.

4.3.2 Scenarios

This case addresses two key challenges in the smart insurance services: a) the micro segmentation of customers to achieve a prompt customer profiling and targeted/smart contracts according to their activity, credibility and usage; and b) the provision of targeted predictions to facilitate the efficient management of customers insurance portfolio or identify future/rare events. The following tables describe in more detail these business scenarios.

Section	Description
Id	SCE-SI-01
Title	Customers segmentation
Description	<p>The scenario focuses on customers' segmentation according to their financial traits, location, etc. Thus, all the customers are classified in groups by spotting patterns in their personal information, preferences or behaviour. This grouping allows developing attitude and solutions especially relevant for the group of customers. As a result, target cross-selling (recommendations of products to customers based on what other customers bought) and upselling (recommendations of more advanced products to customers based on what they have bought) strategies may be developed and personal services may be tailored for each segment (such as lower priced premiums).</p> <p>This scenario is structured around three main steps:</p> <ul style="list-style-type: none"> - Data collection, including the enrichment of corporate datasets with open data mainly related to territorial risks (of different kinds, e.g. floods, earthquakes, crime). - Optimizing the product recommendations for customers. - Show recommended products.
Actors	Customer, Service Provider
Objectives	<ul style="list-style-type: none"> - Provide a clustering of the insurance company customers according to predefined variables (such as #products, territory). - Optimize product suggestions for campaigns and cross selling/upselling strategies.
Pre-conditions	N/A
Process Description	<ul style="list-style-type: none"> - Data collection: the system collects information related to customers and their policies during their history with the company. - Data processing: the system analyses the collected data and elaborates personalized policies and guarantees for the different customers. - Data visualization: the results from the previous phase are optimized in terms of customers' segments and presented through the system's graphics capabilities.
Variations	N/A
Post-condition	<ul style="list-style-type: none"> - Suggested recommendations and optimizations shouldn't be null. - The system must store in the DB the optimizations and recommendations for every customer.

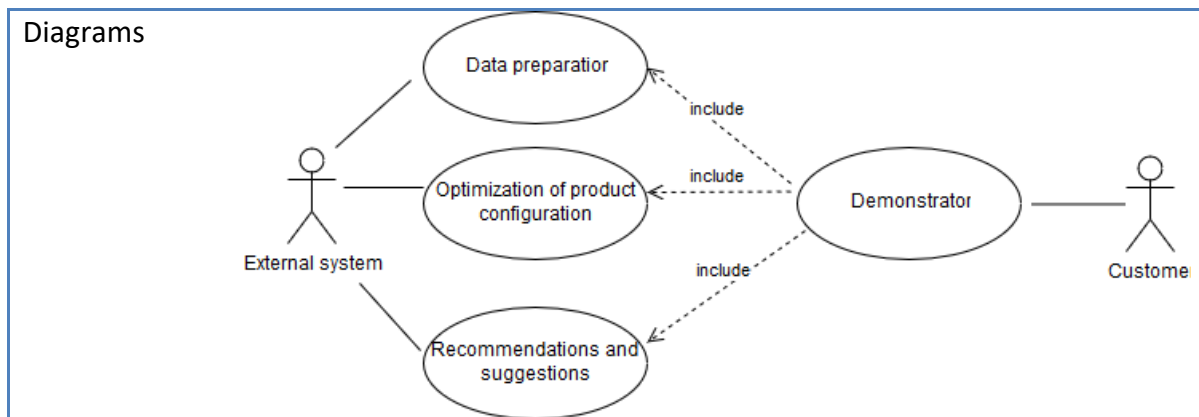


Table 34 – Customers segmentation scenario

Section	Description
Id	SCE-SI-02
Title	Customer Lifetime Value prediction
Description	Customers Lifetime Value (CLV) is a complex phenomenon representing the value of a customer to a company in the form of the difference between the revenues gained and the expenses made projected into the entire future relationship with a customer. Prediction of the CLV is typically assessed via customer behaviour data in order to predict the customer's profitability for the insurer. Thus, the behaviour-based models will be applied to forecast the customer retention . This allows forecasting the likelihood of the customers' behaviour and attitude, as well as churn prevention (identify which customers are likely to cancel contracts soon).
Actors	Customer, Service Provider
Objectives	<ul style="list-style-type: none"> - Compute and update the CLV. - Forecast which customers are likely to cancel contracts soon—i.e. after the expiration of current policies.
Pre-conditions	The prediction approach needs at least 1 year of historical data for efficient predictions.
Process Description	The key aspects in this case are related to data analytics in order to predict the different customers value (e.g., most profitable), analyse present and future profitability, identify target customers, and predict which customers are not satisfied and are likely to cancel their contracts in the future. This information allows enhancing the process described in the first scenario and helps providing better recommendations to customers.
Variations	N/A
Post-condition	<ul style="list-style-type: none"> - The computed CLV should not be null. - The system must store the predicted results about customers.

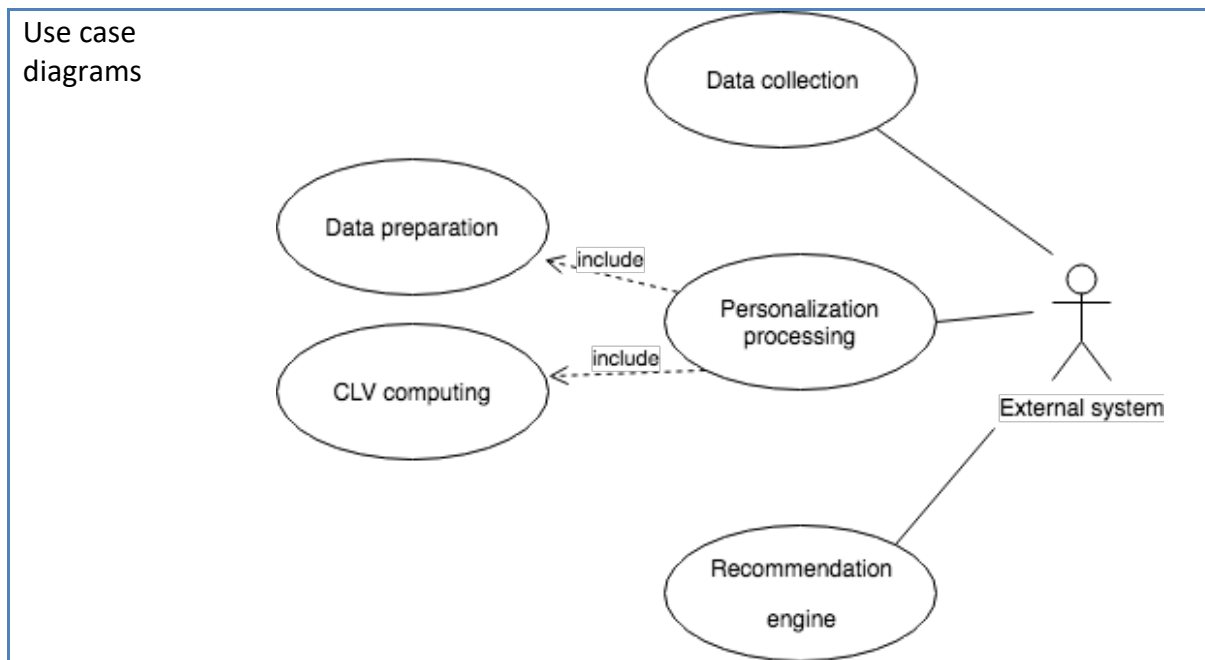


Table 35 – Customer lifetime value prediction scenario

4.3.3 Requirements

The following template introduces the structure of requirements for SI use case.

Section	Description
Id	REQ-SI-01
Type	FUNC
Short name	Provide personalized policies required by customer
Description	For any customer, the system should be able to predict a set of personalized policies for the customer. The calculation should consider the historic of product purchases made by the customer.
Additional Information	N/A
Priority	MAN

Table 36 – Provide personalized policies required by customer (stakeholder requirement)

Section	Description
Id	REQ-SI-02
Type	FUNC
Short name	Provide personalized guarantees required by customer
Description	For any customer, the system should be able to predict a set of personalized guarantees for the customer.

	The calculation should consider the history of product purchases made by the customer.
Additional Information	N/A
Priority	MAN

Table 37 – Provide personalized guarantees required by customer (stakeholder requirement)

Section	Description
Id	REQ-SI-03
Type	DATA
Short name	Customers
Description	Customers should have at least the following information: <ul style="list-style-type: none"> - ClientId - PolicyId - CityId
Additional Information	N/A
Priority	MAN

Table 38 – Customers (stakeholder requirement)

Section	Description
Id	REQ-SI-04
Type	DATA
Short name	Policies
Description	Policies' information should have at least the following information: <ul style="list-style-type: none"> - Policy reference - Policy category
Additional Information	N/A
Priority	MAN

Table 39 – Policies (stakeholder requirement)

Section	Description
Id	REQ-SI-05
Type	Stakeholder
Short name	Usability
Description	Easy-to-use

Additional Information	The application should be easy to use and to understand by people of different ages—i.e., it should follow the best practices in terms of usability and user experience.
Priority	MAN

Table 40 – Easy-to-use (stakeholder requirement)

Section	Description
Id	REQ-SI-06
Type	Stakeholder
Short name	Security
Description	Data security
Additional Information	Database must be accessible by secure means and its data must not be breached.
Priority	MAN

Table 41 – Data security (stakeholder requirement)

5 Platform Roles

The following table shows a description of what BigDataStack offers to different roles related to the development, deployment and operation of Big Data Analytics solutions.

Id	Name	Description
ROL-01	Data Owner	BigDataStack offers a unified Gateway to obtain both streaming and stored data from data owners and store them in its underlying storage infrastructure that supports SQL and NoSQL data stores.
ROL-02	Data Scientist	BigDataStack offers the Data Toolkit to enable data scientists both to easily ingest their analytics tasks by utilizing a declarative paradigm, and to specify their preferences and constraints to be exploited during the dimensioning phase regarding the data services that will be used (for example preferences for the data cleaning service)
ROL-03	Business Analysts	BigDataStack offers the Process Modelling Framework allowing business users to define their functionality-based business processes (through declaratively-defined models) and optimize them based on the outcomes of process analytics that will be triggered by BigDataStack.
ROL-04	Application Engineers and Application Service Owners	BigDataStack offers the Application Dimensioning Workbench to enable application owners and engineers to experiment with their applications and dimension it in terms of its data needs and data-related properties
ROL-05	Data Engineers and Data Service Owners	BigDataStack offers the possibility to Data Service owners (such as the roles implemented by IBM and LXS in this project) to bring in (adapt) their data services to the platform.

Table 42 – BigDataStack Platform roles

6 Data-driven Infrastructure Management Requirements

To facilitate the traceability between requirements and the design decisions as well as technical challenges addressed for the Data-Driven Infrastructure Management capability of BigDataStack, these requirements appear as-is in D3.2⁸, attached to the design of the components implementing them. Nevertheless, the present section should be considered the single source of all requirements for the present capability and be considered the master version of them in case of discrepancies.

	Id	Level of detail	Type	Actor	Priority
	REQ-CM-01	Software	FUNC	Application Engineer, Data Engineer	MAN
Name	Support OpenShift installation on OpenStack VMs				
Description	Include the needed steps on the OpenShift installer to handle OpenShift cluster installation on top of OpenStack resources, i.e., VMs, networks, volumes, etc.				
Additional Information	This needs to be done in the ‘upstream’ way so that it is supported also after the project lifecycle. It entails modification to different repositories, not only the Openshift/installer ⁹ but also other related repositories such as: <ul style="list-style-type: none"> - cluster-network-operator¹⁰ - cluster-api-provider-openstack¹¹ - gophercloud¹² 				

Table 43 - Requirement (1) for Cluster Management

	Id	Level of detail	Type	Actor	Priority
	REQ-CM-02	Software	PERF	Application Engineer, Data Engineer	MAN
Name	Avoid double encapsulation of network packages				
Description	Integrate Kuryr into the OpenShift installer to avoid the double encapsulation problem due to using 2 different overlays (OpenStack SDN and OpenShift SDN on top). Kuryr enables containers running on top of OpenStack VMs to use the same SDN as the VMs itself, i.e., the OpenStack SDN. Thus, avoiding the double encapsulation and enabling a remarkable throughput gain.				
Additional Information	Similarly, to REQ-CM-01, this needs to be done in the ‘upstream’ way so that it is supported after the project. It entails modifications to the same				

⁸ D3.2 – WP3 Scientific Report and Prototype Description – Y2 (due M23).

⁹ <https://github.com/openshift/installer>

¹⁰ <https://github.com/openshift/cluster-network-operator>

¹¹ <https://github.com/kubernetes-sigs/cluster-api-provider-openstack>

¹² <https://github.com/gophercloud/gophercloud>

	repositories plus the addition of a kuryr operator that will handle the kuryr related operational actions,
--	--

Table 44 - Requirement (2) for Cluster Management

	Id	Level of detail	Type	Actor	Priority
	REQ-CM-03	Software	FUNC	Application Engineer, Data Engineer	MAN
Name	Kubernetes Network Policy support at Kuryr-Kubernetes				
Description	As we are integrating kuryr to get network performance optimizations when running OpenShift on top of OpenStack, we need to include the mechanisms needed for kuryr to be able to enforce Kubernetes network policies, i.e., to define in a fine grain manner how pods can communicate with each other				
Additional Information	Similarly, to REQ-CM-01, this needs to be done in the 'upstream' way so that it is supported after the project. It entails modifications to the Kuryr-Kubernetes repositories.				

Table 45 - Requirement (3) for Cluster Management

	Id	Level of detail	Type	Actor	Priority
	REQ-CM-04	System	PERF	Application Engineer, Data Engineer	DES
Name	OVN-base distributed load balancer for Kubernetes services				
Description	Kubernetes services are implemented through Octavia when using Kuryr. This means that for each Kubernetes service an Octavia amphora VM is created. This adds extra latency on the communication, is a single point of failure, adds extra resources need, and it adds delays on the control plane actions. By integrating the OVN distributed load balancer (as a new ovn-Octavia driver) and making Kuryr use it, we avoid all those problems by implementing the load balancing directly with ovn flows. This remove the need for VM resources and speed up both control and data planes.				
Additional Information	Similarly, to REQ-CM-01, this needs to be done in an 'upstream' way so that it is supported after the project. It entails modifications and integration in several upstream projects: <ul style="list-style-type: none"> - OVN - OpenStack Octavia - OpenStack networking-ovn - Kuryr-Kubernetes - OpenShift Cluster Network Operator 				

Table 46 - Requirement (4) for Cluster Management

	Id	Level of detail	Type	Actor	Priority
	REQ-CM-05	System	FUNC	Application Engineer, Data Engineer	MAN
Name	API managed OpenShift cluster (CAPO)				
Description	The OpenShift cluster installed on top of OpenStack consists of X VMs on OpenStack. We need to extend the Cluster API Provider OpenStack in order to allow the modification of the cluster size through Kubernetes API calls. This allows flexibility on the OpenShift cluster to adapt to the current needs				
Additional Information	Similarly, to REQ-CM-01, this needs to be done in the 'upstream' way so that it is supported after the project. It entails modifications to the next upstream projects: <ul style="list-style-type: none"> - openshift/cluster-api - openshift/cluster-api-provider-openstack - openshift/installer 				

Table 47 - Requirement (5) for Cluster Management

	Id	Level of detail	Type	Actor	Priority
	REQ-CM-06	System	ENV	Data Engineer	DES
Name	Spark operator				
Description	This operator will be responsible for handling the Spark cluster, not only its installation but also the scaling actions. It will offer an API to the Spark management through the OpenShift API.				
Additional Information	This is related to the dynamic orchestrator, as the optimization actions could be then simply triggered through standard OpenShift API commands (e.g., modifying the information at the associated spark ConfigMap)				

Table 48 - Requirement (6) for Cluster Management

	Id	Level of detail	Type	Actor	Priority
	REQ-CM-07	System	ENV	Adaptable Distributed Storage	DES
Name	Accept requests to allocate additional resources to storage components				
Description	The Adaptable Distributed Storage component can be scaled in/out independently, considering decisions based on its internal metrics and handle on its own the reconfiguration of the internal data regions. Due to this, it is necessary from the Cluster Management to provide a mechanism that allows the storage layer to request for additional resources or the release of already provided ones.				
Additional Information	This is closely related to requirement REQ-ADS-04 "Be able to request additional resources from the infrastructure layer," described in D4.1.				

Table 49 - Requirement (7) for Cluster Management

	Id	Level of detail	Type	Actor	Priority
	REQ-CM-08	System	ENV	Adaptable Distributed Storage	OPT
Name	Force the storage layer to release some of its available resources				
Description	The cluster management might identify that the overall BigDataStack platform is running out of available resources. To ensure the execution of crucial components, it might decide to reduce resources for some services, to the benefit of others. Due to this, it should be able to request the release of the storage resources and wait for its proper response. The storage should be able to reject such requests, in cases that could lead to data loss.				
Additional Information	This is close related with requirement REQ-ADS-05 “Being able to release resources and adapt if resources are deallocated from the infrastructure,” as described in more details in D4.1.				

Table 50 - Requirement (8) for Cluster Management

	Id	Level of detail	Type	Actor	Priority
	REQ-DO-01	System	FUNC	Application Engineer, Data Engineer	MAN
Name	Playbook Enrichment				
Description	The Dynamic Orchestrator shall ingest the Playbook when an application or service is deployed and enrich this playbook with information about the QoS metrics and intervals to be considered by the Triple Monitoring to monitor the QoS during runtime.				
Additional Information	N/A				

Table 51 - Requirement (1) for Dynamic Orchestrator

	Id	Level of detail	Type	Actor	Priority
	REQ-DO-02	Stakeholder	FUNC	Application Engineer, Data Engineer	MAN
Name	Re-deployment Decision to Meet SLO				
Description	When an application or service is running, the Dynamic Orchestrator shall determine if a deployment change should be performed when there is a violation of an application requirement or Service Level Objective (SLO) and send a signal to the ADS-ranker to trigger a change in the deployment to try to satisfy the requirements or SLOs.				
Additional Information	The Triple Monitoring detects this violation and sends an alert to the Dynamic Orchestrator to start this process.				

Table 52 - Requirement (2) for Dynamic Orchestrator

	Id	Level of detail	Type	Actor	Priority
	REQ-DO-03	Stakeholder	PERF	Application Engineer, Data Engineer	MAN
Name	Decision Efficiency				
Description	The orchestrator shall be able to decide what modification to the deployment (e.g. change the number of replicas or the number of vCPUs) has the highest probability of improving the requirements or SLOs satisfaction, as long as any change is possible (i.e. all resources are at its full capacity due to limits).				
Additional Information	N/A				

Table 53 - Requirement (3) for Dynamic Orchestrator

	Id	Level of detail	Type	Actor	Priority
	REQ-DO-04	System	FUNC	Application Engineer, Data Engineer	MAN
Name	Resources Limits				
Description	The orchestrator shall be able to receive a trigger from the ADS-Ranker when a deployment parameter, such as the number of replicas, the number of vCPUs or the assigned cluster memory, cannot be further increased or decreased (i.e. this resource has reached its maximum or minimum possible value) and use this information in its own decisions.				
Additional Information	The complete list of deployment parameters might vary according to the application/service and its actual deployment. This information should be available in the Playbook or other resource.				

Table 54 - Requirement (4) for Dynamic Orchestrator

	Id	Level of detail	Type	Actor	Priority
	REQ-DO-05	Stakeholder	FUNC	Application Engineer, Data Engineer	DES
Name	Orchestration for Improvements				
Description	When an application or service is running, the orchestrator shall detect changes in the system status or inputs (e.g. less new events per minute) and trigger a change in the deployment that results in lower costs (e.g. to use less replicas) without compromising the application functioning.				
Additional Information	N/A				

Table 55 - Requirement (5) for Dynamic Orchestrator

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSR-01	System	FUNC	Application Dimensioning Workbench	MAN
Name	Ingest Candidate Deployment Playbooks and Benchmarking Information				
Description	The Application Dimensioning Workbench sends a series of candidate deployment patterns (CDP) playbooks and benchmarking information to the ADS Ranking component. ADS Ranking needs to collect all these patterns for subsequent scoring/ranking based on the user requirements and preferences.				
Additional Information	Ingestion occurs via a common publisher/subscriber platform (RabbitMQ).				

Table 56 - Requirement (1) for ADS Ranking

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSR-02	System	FUNC	Dynamic Orchestrator, Application Dimensioning Workbench	MAN
Name	Deployment Suitability Feature Extraction				
Description	Once a series of candidate deployment pattern playbooks and associated benchmarking information has been received, the next step is to determine how each pattern is predicted to perform based on the benchmarking information. In effect, this involves defining a series of functions that relate individual or groups of user requirements to the predicted performances produced by benchmarking. The output of this step is a vector representation for each CDP playbook, representing how that playbook is predicted to perform under different user requirements.				
Additional Information	Features produced here are dependent on the capabilities of the benchmarking system and the amount of information the user provides in terms of requirements and preferences.				

Table 57 - Requirement (2) for ADS Ranking

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSR-03	System	FUNC	Dynamic Orchestrator, Application Dimensioning Workbench	MAN
Name	CDP Playbook Scoring (Heuristic)				
Description	Given a vector representation for a CDP Playbook, we next need to map this vector into a single score, representing how suitable that playbook will				

	be overall (such that we can compare different CDP Playbooks). This involves combining the different elements within the vector (that each represent some aspect of pattern suitability, such as cost, or predicted compute wastage). The first version of this component will use a hand-tuned linear combination.
Additional Information	N/A

Table 58 - Requirement (3) for ADS Ranking

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSR-04	System	FUNC	Dynamic Orchestrator, Application Dimensioning Workbench	DES
Name	CDP Playbook Scoring (Supervised)				
Description	Given a vector representation for a CDP Playbook, we next need to map this vector into a single score, representing how suitable that playbook will be overall (such that we can compare different CDP Playbooks). This involves combining the different elements within the vector (that each represent some aspect of pattern suitability, such as cost, or predicted compute wastage). The second version of this component will learn how to combine the elements based on logging information from past deployments. Models may be non-linear in nature.				
Additional Information	Depends on REQ-ADSR-06.				

Table 59 - Requirement (4) for ADS Ranking

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSR-05	System	FUNC	Dynamic Orchestrator, Application Dimensioning Workbench	MAN
Name	CDP Playbook Selection				
Description	Once all candidate deployment patterns have been scored, the final step is to select one of those patterns to pass to ADS Deployment. In many cases this will simply involve selecting the highest scoring pattern. However, the user may have the option to select an alternative configuration at this stage.				
Additional Information	N/A				

Table 60 - Requirement (5) for ADS Ranking

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSR-06	System	FUNC	Dynamic Orchestrator, Application Dimensioning Workbench	DES
Name	Supervised Model Training				
Description	To support REQ-ADSR-04, a supervised scoring model is needed. To react to changes in the deployment environment over time, this model needs to be frequently updated based on new information from current deployments. This model needs to be trained based on logging data being collected by the Triple Monitoring Framework.				
Additional Information	Requires logging information produced by the Triple Monitoring Framework and stored in the Central Decision Tracker.				

Table 61 - Requirement (6) for ADS Ranking

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSR-07	System	FUNC	Dynamic Orchestrator	MAN
Name	CDP Playbook Re-Scoring				
Description	It is envisaged that in (rare) scenarios, an ongoing application deployment will fail to meet the user's quality of service requirements. For instance, this might occur due to assumptions on data input volumes being violated. In this case, we may not be able to solve this issue without fully redeploying the user application with different resources. To support such re-deployment activities, ADS Ranking supports a re-scoring function, where a previous set of CDP playbooks for a user's application can be re-scored based on updated preferences provided by the Dynamic Orchestrator, as well as data about how the previous deployment performed (and failed).				
Additional Information	N/A				

Table 62 - Requirement (7) for ADS Ranking

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSR-08	System	FUNC	ADS Ranking	DES
Name	Deployment Dataset Generation				
Description	To support REQ-ADSR-06 and hence REQ-ADSR-04, significant volumes of logging data from past deployments are needed to enable effective model creation. To this end, a framework and methodology for generating this data is needed. Such logging data can be produced through either benchmarking, live deployment of the end-user applications and via simulated application deployment.				

Additional Information	Data storage for this task is handled by the Triple Monitoring Framework and Central Decision Tracker. Data generation is supported by deployments by the application dimensioning workbench and other dedicated deployment applications.
-------------------------------	---

Table 63 - Requirement (8) for ADS Ranking

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSD-01	Stakeholder	FUNC	ADS Ranking	MAN
Name	Performance Measurability				
Description	Each environment should be measurable according to a set of characteristics, that is, Key Performance Indicators (KPIs).				
Additional Information	The KPIs considered must include: <ul style="list-style-type: none"> - vCPUs - Memory 				

Table 64 - Requirement (1) for ADS Deploy

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSD-02	Stakeholder	FUNC	Application Engineer, Data Engineer	MAN
Name	Standards-based Playbook				
Description	The description of the environments and deployments (i.e., playbooks) will follow a specification language that is intuitive and as close (similar) as possible to well-known and widely-used schemas to describe software application deployments in cloud infrastructures, such as Docker Compose or Kubernetes Deployment.				
Additional Information	N/A				

Table 65 - Requirement (2) for ADS Deploy

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSD-03	System	FUNC	Application Engineer, Data Engineer	MAN
Name	Standard deployment information				
Description	When communicating with other components, as described in Section 7.2, these components will use the playbook standard defined in REQ-RD-02.				
Additional Information	N/A				

Table 66 - Requirement (3) for ADS Deploy

	Id	Level of detail	Type	Actor	Priority
--	-----------	------------------------	-------------	--------------	-----------------

	REQ-ADSD-04	System	FUNC	ADS Ranking	MAN
Name	Application Scoring System				
Description	The ranking system evaluates each environment's deployment, which keeps track of the most suitable configuration for each application. When trying a deployment configuration for a new application, this ranking will be used to select the most suitable one.				
Additional Information	The evaluation needs to be performed following the measurements defined in REQ-RD-01.				

Table 67 - Requirement (4) for ADS Deploy

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSD-05	Software	FUNC	Cluster Management	MAN
Name	Compatibility with Kubernetes				
Description	Since the technology used to run and orchestrate the applications is based on Kubernetes (OKD ¹³). Thus, the ADS-Deployment component is required to be compatible with Kubernetes.				
Additional Information	The ADS-Deploy component should translate from the playbook standard defined in REQ-RD-01 into Kubernetes primitives.				

Table 68 - Requirement (5) for ADS Deploy

	Id	Level of detail	Type	Actor	Priority
	REQ-ADSD-06	System	PERF	ADS Ranking	MAN
Name	Synchronous communication				
Description	The communication with and within ADS Ranking and ADS Deploy must be done through an API REST.				
Additional Information	N/A				

Table 69 - Requirement (6) for ADS Deploy

	Id	Level of detail	Type	Actor	Priority
	REQ-QOS-01	System	FUNC	Developer	MAN
Name	Regular recording of QoS metrics				
Description	When a user's application is deployed, the Triple Monitoring Engine and QoS Evaluation monitors that application, tracking statistical information about its operation and associated QoS data, including network, data storage, virtualization layers, etc.				

¹³ OKD - <https://www.okd.io/>

	This data is needed to support the learning of ranking models by ADS-Ranking service (part of Application and Service Deployment; see REQ-ADSR-03) and regularly saved in a centralised data store for later access.
Additional Information	N/A

Table 70 - Requirement (1) for QoS Evaluation

	Id	Level of detail	Type	Actor	Priority
	REQ-QOS-02	System	FUNC	Developer	MAN
Name	QoS violation alert				
Description	If the system does not respect the agreed QoS, an alert is raised.				
Additional Information	This alert is used internally to evaluate the performance of an environment, relating to REQ-RD-004.				

Table 71 - Requirement (2) for QoS Evaluation

	Id	Level of detail	Type	Actor	Priority
	REQ-QOS-03	System	FUNC	Developer	DES
Name	QoS violation monitoring				
Description	QoS violations are also monitored and shown to the user/admin.				
Additional Information	N/A				

Table 72 - Requirement (3) for QoS Evaluation

	Id	Level of detail	Type	Actor	Priority
	REQ-QOS-04	System	PERF	Dynamic Orchestrator	DES
Name	Asynchronous rich notification of QoS violations				
Description	QoS violations should be notified by means of a publisher/subscriber mechanism, together with the id of the metrics to which the SLO applies.				
Additional Information	The main consumer of the SLA violations notifications is the Dynamic Orchestrator.				

Table 73 - Requirement (4) for QoS Evaluation

	Id	Level of detail	Type	Actor	Priority
	REQ-QOS-05	System	PERF	Dynamic Orchestrator	MAN
Name	Ability to detect spike violations of QoS				
Description	The QoS Evaluation must consider all data points (i.e. measurements) stored in the monitoring system's time series database when computing				

	SLO violations to cover spikes that may be missed by any sampling method otherwise.
Additional Information	N/A

Table 74 - Requirement (5) for QoS Evaluation

	Id	Level of detail	Type	Actor	Priority
	REQ-QOS-06	System	PERF	Dynamic Orchestrator	MAN
Name	Evaluate aggregated behaviour with a certain level of confidence				
Description	The QoS Evaluation must consider the aggregated measurements during a given time window to determine the compliance of the SLO for the most part or a given period or time window, to avoid notifying violations due to outliers or sporadic spikes in the measurements. We define “for the most part” as the level of confidence we can have in the evaluation of the SLO.				
Additional Information	<p>There exist different ways in which we can “assess” a group of data points or measurements to determine whether they comply with the objective “for the most part”. The most common way is to aggregate data points in groups of n and determine whether the entire group complies with the objective, and using aggregation functions such as quantiles/percentiles. Thus, the SLO evaluation need to follow the following norm:</p> <ul style="list-style-type: none"> - Metric < objective for percentage of measurements collected in time window <p>For example, that:</p> <ul style="list-style-type: none"> - <i>Response time < 900ms</i> for 99% measurements collected in <i>10min</i> <p>This percentage can be calculated as the percentile 99th or 0.99 quantile (also known as 99% quantile), depending on the nomenclature we use.</p>				

Table 75 - Requirement (6) for QoS Evaluation

	Id	Level of detail	Type	Actor	Priority
	REQ-QOS-07	System	FUNC	Dynamic Orchestrator	MAN
Name	Management of the quality evaluation tasks				
Description	The QoS Evaluation must be responsible for managing the life cycle of quality monitoring and evaluation tasks for the applications and services deployed in the BigDataStack platform. These tasks must be initiated and stopped following the lifecycle of the application and service deployments, i.e., to be active only when the corresponding deployment is running.				
Additional Information	N/A				

Table 76 - Requirement (7) for QoS Evaluation

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-01	System	FUNC	Application Engineer, Data Engineer	MAN
Name	Metrics pusher				
Description	The metric pusher retrieves KPI data, cleans them and ingests them into the monitoring collector (<i>Prometheus</i>).				
Additional Information	The metrics pusher is used when the exporter approach is impossible to apply—since Prometheus exporters require an HTTP server to publish metrics for the monitoring collector, components that lack this service need an alternative. Besides, this solution will be very useful for getting application specific metrics. This component is a REST-API and Prometheus Exporter which receives KPI data over HTTP in JSON format, then format them and ingest them into Prometheus.				

Table 77 - Requirement (1) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-02	System	FUNC	QoS Evaluation	DES
Name	RESTful API for accessing the collected monitoring metrics				
Description	The metrics are accessible through a RESTful API.				
Additional Information	This component translates client's requests to Prometheus request compatible. Grafana ¹⁴ will be used for visualization.				

Table 78 - Requirement (2) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-03	System	FUNC	QoS Evaluation, Dynamic Orchestrator	MAN
Name	Metrics publication				
Description	Measurements stored in the metrics repository must be periodically published through a publisher/subscriber mechanism. The publication of measurements must start and stop following the request made by the QoS Evaluation, as this component is also responsible for managing the life cycle of the quality monitoring and evaluation tasks (see REQ-QOS-07).				
Additional Information	The monitoring metrics getter is implemented using RabbitMQ ¹⁵ .				

Table 79 - Requirement (3) for Triple Monitoring Engine

¹⁴ Grafana. <https://grafana.com/>

¹⁵ RabbitMQ. <https://www.rabbitmq.com/>

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-04	Software	FUNC	Application Engineer, Data Engineer	DES
Name	Spark compatible				
Description	The triple monitoring engine monitors the performance of Apache Spark ¹⁶ , which is used in the BigDataStack project as an analytics engine for Big Data, and thus needs to be compatible with this technology.				
Additional Information	Monitoring Spark is done using the Spark measure project, which can be embedded in a Spark application to allow the collection of some metrics after each SQL execution. Those metrics are sent to <i>push gateway</i> to be exported to Prometheus.				

Table 80 - Requirement (4) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-05	Software	FUNC	Application Engineer, Data Engineer	DES
Name	LeanXcale compatibility				
Description	LeanXcale database ¹⁷ already uses Prometheus for its monitoring subsystem. However, the integration relies on deployments that require the manual reconfiguration for the Prometheus in cases of scaling actions. Thus, it should be extended to consider automatic re-deployments driven by an elasticity action and automatically reconfigure the integration with the Prometheus monitoring system. In these scenarios, LeanXcale should reconfigure its integration with the existing Prometheus deployment on the run-time and provide monitoring information for the new nodes.				
Additional Information	N/A				

Table 81 - Requirement (5) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-06	Software	FUNC	Application Engineer, Data Engineer	DES
Name	OKD compatibility				
Description	The Triple Monitoring Engine (TPE) monitors the performance of OpenShift OKD ¹⁸ , which is the baseline technology used in the orchestration of containers. Therefore, the TME should be compatible with this technology.				

¹⁶ Apache Spark. <https://spark.apache.org/>

¹⁷ LeanXcale. <https://www.leanxcale.com/>

¹⁸ Openshift OKD (Origin Kubernetes Distribution). <https://www.okd.io/>

Additional Information	N/A
-------------------------------	-----

Table 82 - Requirement (6) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-07	Software	FUNC	Application Engineer, Data Engineer	DES
Name	CEP compatibility				
Description	The Triple Monitoring Engine (TME) monitors the performance of the UMP CEP (Complex Event Processing), which is used in the BigDataStack project as a streaming engine for processing data in real-time. Therefore, the TME needs to be compatible with this technology. The integration with TME is done by federating Prometheus instances, that is, connecting the CEP' Prometheus-based monitoring system into DECENTER Prometheus-based central monitoring system.				
Additional Information	The CEP exposes several monitoring metrics that are exported to Prometheus.				

Table 83 - Requirement (7) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-08	Software	FUNC	Application Engineer, Data Engineer	DES
Name	Minio compatibility				
Description	The Triple Monitoring Engine (TME) monitors the performance of Minio ¹⁹ , which is used for object storage in the system. Therefore, the TME needs to be compatible with this technology.				
Additional Information	N/A				

Table 84 - Requirement (8) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-09	Software	FUNC	Application Engineer, Data Engineer	DES
Name	OpenStack Networking Services compatibility				
Description	The Triple Monitoring Engine (TME) should monitor the performance of the internal network connecting the different containers inside a deployed application. BigDataStack networking resources and services are provided through a solution stack combining OpenShift, Kuryr and Neutron, that is,				

¹⁹ Minio Private Cloud Storage- <https://www.minio.io/>

	the networking services from OpenStack (see requirements REQ-CM-02, REQ-CM-03 and REQ-CM-04). This means that networking is ultimately provided by OpenStack and hence the need for integrating of TME with it to have networking monitored.
Additional Information	Cluster Management (OpenShift) and Information-Driven Networking components interoperate to provide reliable networking to the applications and services deployed on the BigDataStack platform.

Table 85 - Requirement (9) for Triple Monitoring Engine.

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-10	Software	FUNC	Application Engineer, Data Engineer	MAN
Name	Persistently store the monitoring metrics				
Description	The Triple Monitoring Engine (TME) should use a database for persistently storing monitoring metrics and is connected to <i>Prometheus</i> via <i>PrometheusBeat</i> .				
Additional Information	Metrics saved persistently will be used for historical reason.				

Table 86 - Requirement (10) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-11	Software	FUNC	Application Engineer, Data Engineer	ENH
Name	Spark Monitoring Pushgateway				
Description	This component is used to gather metrics from Spark and ingest them into the metrics collector.				
Additional Information	The connection between this component and the applications will use HTTP.				

Table 87 - Requirement (11) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-12	Software	FUNC	Application Engineer, Data Engineer	ENH
Name	Metrics visualization				
Description	The metrics must be shown to the end-user via a graphical interface. Grafana is used for metrics' visualization.				

Additional Information	Grafana ²⁰ is configured for receiving metrics from two sources (Prometheus, InfluxDB).
-------------------------------	--

Table 88 - Requirement (12) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-TME-13	System	FUNC	QoS Evaluation	DES
Name	Metrics aggregation				
Description	The metrics publication (through publisher/subscriber pattern) process (see REQ-TME-03) should be also in charge of aggregating measurements in periods so the aggregated metrics can be then used by the QoS Evaluation to work in base of confidence levels and time intervals (see REQ-QOS-06).				
Additional Information	The aggregation function to use will be quantiles/percentiles, depending on the nomenclature we use (see REQ-QOS-06 for more details).				

Table 89 - Requirement (13) for Triple Monitoring Engine

	Id	Level of detail	Type	Actor	Priority
	REQ-GDT-01	System	FUNC	ADS Ranking, ADS Deploy, Dynamic Orchestrator	MAN
Name	Application and Deployment Information API				
Description	<p>Once the application engineer uploads a new application deployment, (represented as a CDP playbook) to the Data Toolkit, that application must be persistently stored within the BigDataStack platform and its state monitored. This is the role of the Global Decision Tracker (GDT).</p> <p>In effect, the Global Decision tracker needs to provide an API that allows for new applications to be created, and then updated with information provided by either the Application Dimensioning Workbench or the Realisation layer components of BigDataStack. This includes information about different deployments of that application and any events that impact or alter those deployments.</p>				
Additional Information	The Global Decision Tracker (GDT) in practice acts as a user-friendly API for application information retrieval, backed by a separate storage solution. Any BigDataStack component can use this API to update application data or provide notifications or alerts associated to an application. The visualisation toolkit also uses this API to retrieve application information for display to the application engineer.				

Table 90 - Requirement (1) for Global Decision Tracker

²⁰ Grafana - <https://grafana.com/>

	Id	Level of detail	Type	Actor	Priority
	REQ-GDT-02	Software	SEC	ADS Ranking, Application Engineer, Data Engineer	MAN
Name	Secure Logging Data Export				
Description	<p>To support REQ-ADSR-08, information about previously deployed applications need to be exportable for use in application deployment datasets. Application Engineers should be able to export logging data about their own applications without redacting the contents. Meanwhile, platform administrators should be able to export all applications, subject to redaction of any personal information.</p> <p>This export functionality is strongly linked with the related functionality of the Triple Monitoring Engine (TME), which provides the low-level export functionality for underlying time-series metrics data, while the GDT provides the exporting of the application- and deployment-level data.</p>				
Additional Information	In the cases where data redaction is required, only a sub-set of pre-defined 'safe' fields will be exported which cannot contain personal data. For example, CPU and memory usage would be exportable, but application owner or environment variables would not.				

Table 91 - Requirement (2) for Global Decision Tracker

	Id	Level of detail	Type	Actor	Priority
	REQ-IDN-01	System	FUNC	Data Scientist	MAN
Name	Information-Driven Networking based on type of data				
Description	The Information-Driven Networking mechanisms enforce a set of policies by specifying the rules of how two or more components can communicate (send/receive data) with each other according to the available resources.				
Additional Information	A different policy is enforced based on different incoming data requirements, following the type of processing requirements (stream, micro-batch, batch) and the type of data (structured, semi-structured, unstructured).				

Table 92 - Requirement (1) for Information-Driven Networking

	Id	Level of detail	Type	Actor	Priority
	REQ-IDN-02	Software	FUNC	Data Scientist	MAN
Name	Information-Driven Networking based on application requirements				
Description	The Information-Driven Networking mechanisms enforce a set of policies by specifying the rules of how to handle applications with different requirements according to the available resources. For instance, an				

	<p>application with analytics requiring real-time data processing may impose time-critical constraints on the handling, operation and transformation of data. To support online analytics and decision making in time-critical conditions specific network policies need to be applied to deliver the results within predefined time constraints.</p>
<p>Additional Information</p>	<p>The Data Scientist can set an “allow/deny access” policy regarding the set of applications and their requirements (real-time, close to real-time needs) accessing the backend services of the BigDataStack environment to prioritize/isolate the set of ingress/egress workloads that are enabled/disabled based on their IP & Port in order to achieve efficient services interaction.</p>

Table 93 - Requirement (2) for Information-Driven Networking

7 Data-as-a-Service Requirements

To facilitate the traceability between requirements and the design decisions as well as technical challenges addressed for the Data-as-a-Service (DaaS) capability of BigDataStack, these requirements appear as-is in D4.2²¹, attached to the design of the components implementing them. Nevertheless, the present section should be considered the single source of all requirements for the present capability and be considered the master version of them in case of discrepancies.

	Id	Level of detail	Type	Actor	Priority
	REQ-BDL-01	Stakeholder	FUNC	Data Engineer	MAN
Name	Support data skipping for arbitrary query predicates				
Description	The query predicate could comprise UDFs and AND/OR/NOT. Example UDFs could be geospatial or temporal functions.				
Additional Information	This functionality is important for the ship management use case, which requires geospatial UDFs.				

Table 94 - Requirement (1) for Big Data Layout

	Id	Level of detail	Type	Actor	Priority
	REQ-BDL-02	System	FUNC	Data Engineer	MAN
Name	Support a truly pluggable architecture for data skipping				
Description	To enable the addition of new data skipping index types without changing the core data skipping library. This is needed for requirement REQ-BDL-01 since supporting new UDFs may require new index types.				
Additional Information	External users can also exploit this capability				

Table 95 - Requirement (2) for Big Data Layout

	Id	Level of detail	Type	Actor	Priority
	REQ-BDL-03	Stakeholder	FUNC	Data Engineer	MAN
Name	Enable layout change for (part of) a dataset				
Description	There is a strong relationship between how a dataset is laid out in the object store and the performance of data skipping against this data set. Moreover, this performance may be also very dependent on the queries. Hence the need to adapt the layout, not only for future data but also for heavily queried data already in object store.				
Additional Information	N/A				

Table 96 - Requirement (3) for Big Data Layout

²¹ D4.2 – WP4 Scientific Report and Prototype Description – Y2 (due M23).

	Id	Level of detail	Type	Actor	Priority
	REQ-BDL-04	Stakeholder	FUNC	Data Engineer	MAN
Name	Enable on-line data layout				
Description	Layout is critical for the data skipping performance. As of now data is stored as is and possibly laid out again offline. The need is to upload dataset chunks with the best-known layout as data is ingested.				
Additional Information	N/A				

Table 97 - Requirement (4) for Big Data Layout

The following requirements for the Big Data Layout task were added at M22:

	Id	Level of detail	Type	Actor	Priority
	REQ-BDL-05	Software	FUNC	Data Engineer	MAN
Name	Support Object Store as metadata store				
Description	Currently an ElasticSearch cluster must be installed to store the metadata pertaining to data skipping. This requirement adds a moving part to the Data Skipping solution deployment. In addition, ElasticSearch has some limitations such as not implementing Bloom Filters as first-class citizen. We want to move away from ElasticSearch and replace it by the object store itself.				
Additional Information	N/A				

Table 98 - Requirement (5) for Big Data Layout

	Id	Level of detail	Type	Actor	Priority
	REQ-BDL-06	Software	FUNC	Data Engineer	MAN
Name	Support new indexes				
Description	We currently support three indexes: min/max, value list and geo index (which permits to address 2 columns at once such as latitude and longitude). Initial users feedback leads us to add new index the objects. For example, there are many situations where value list should be replaced by much more compact index such as a bloom filter or a gap list index.				
Additional Information	N/A				

Table 99 - Requirement (6) for Big Data Layout

	Id	Level of detail	Type	Actor	Priority
	REQ-BDL-07	System	USE	Data Engineer	DES
Name	Automatic index selection				

Description	Currently the dataset owner must specify what column should be indexed and with what kind of index. The technology will be more user friendly if we could automatically infer which columns should be indexed and how (e.g., with a value list or with a bloom filter index).
Additional Information	N/A

Table 100 - Requirement (7) for Big Data Layout

	Id	Level of detail	Type	Actor	Priority
	REQ-BDL-08	Stakeholder	PERF	Data Engineer	DES
Name	Enhance the data layout tool				
Description	The indexing of a large dataset takes a long time. Its performance should be improved. This is critical if we detect a change of query load and if we want the flexibility to re-index (parts of) existing dataset to improve the data skipping score.				
Additional Information	N/A				

Table 101 - Requirement (8) for Big Data Layout

	Id	Level of detail	Type	Actor	Priority
	REQ-BDL-09	Software	FUNC	Data Engineer	MAN
Name	Hive Metastore support				
Description	Hive Metastore is a relational database used by Apache Spark SQL to manage the metadata of the persistent relational entities such as databases, columns, etc.				
Additional Information	We need to integrate our Data Skipping technology with Hive Metastore.				

Table 102 - Requirement (9) for Big Data Layout

	Id	Level of detail	Type	Actor	Priority
	REQ-ADS-01	System	DATA	Data Engineer	MAN
Name	Being able to fragment a dataset and move the data fragments across different nodes.				
Description	The adaptable distributed storage should be able to split a dataset into different regions, and move these regions to different data nodes, in order to adapt in case of increased load (both in terms of user workload or data load) to achieve efficient consumption, based on the provided resources.				
Additional Information	When a movement (move, split, join) of a data fragment occurs, the storage must not suffer from a down-time. On the contrary, it must remain operational with minimum overhead on the overall performance.				

Table 103 - Requirement (1) for Adaptable Distributed Storage

	Id	Level of detail	Type	Actor	Priority
	REQ-ADS-02	System	ENV	Data Engineer	MAN
Name	Identify data nodes that are overprovisioned.				
Description	The adaptable storage must be able to identify data nodes that are overprovisioning their available resources and send internal alerts to trigger a dynamic reconfiguration of the deployment of the data fragments.				
Additional Information	N/A				

Table 104 - Requirement (2) for Adaptable Distributed Storage

	Id	Level of detail	Type	Actor	Priority
	REQ-ADS-03	System	FUNC	Data Engineer	DES
Name	Solve the non-linear resource allocation problem to suggest alternative deployment of the data fragments.				
Description	According to the available resources for the deployment of the data nodes and the stored data set, along with its split points that define data fragments, there is a non-linear resource allocation problem for the optimal deployment of the data fragments.				
Additional Information	As a non-linear problem, the solution of the resource allocation problem requires exponential time to be solved, which is not acceptable for run-time requirements. The provided solution should consider possible acceptable solutions that can solve the problem and improve the resource consumption, under a minimum time interval.				

Table 105 - Requirement (3) for Adaptable Distributed Storage

	Id	Level of detail	Type	Actor	Priority
	REQ-ADS-04	System	ENV	Data Engineer	DES
Name	Be able to request additional resources from the infrastructure layer.				
Description	In case of overprovisioning of the resources, the adaptable distributed storage should be able to request additional resources from the infrastructure of BigDataStack.				
Additional Information	<p>As noted in REQ-ADS-02, the adaptable storage must identify data nodes that are overprovisioning, and using REQ-ADS-03, it can suggest different distribution of the data fragments. However, there might be cases that this is not possible due to the overprovisioning of the whole system, and in such cases, a horizontal scale out must take place. The adaptable storage should request additional resources and have granted them if they are available. The communication should be as follows:</p> <ul style="list-style-type: none"> - The adaptable storage requests an additional node with the specific requirements for resources. - The infrastructure responds if it can allocate additional resources for 				

	<p>the storage.</p> <ul style="list-style-type: none"> - The infrastructure informs the storage that the additional resources are now available. <p>This requirement also includes the need of the adaptable storage to inform the infrastructure that it can release resources that are not needed.</p>
--	---

Table 106 - Requirement (4) for Adaptable Distributed Storage

	Id	Level of detail	Type	Actor	Priority
	REQ-ADS-05	System	ENV	Data Engineer	OPT
Name	Being able to release resources and adapt if resources are deallocated from the infrastructure.				
Description	There might be cases where the whole infrastructure is overprovisioned and there are no more resources to be allocated to tasks. Then, the infrastructure might decide to reduce the overall resources of specific components, in favour of others that might execute some critical operations, or they have biggest priority at that point. The adaptable storage engine should be listening to the infrastructure for such cases and adapt accordingly.				
Additional Information	Once the adaptable distributed storage receives a request to release some of its nodes, then it should inform if it can do so: releasing some the data nodes, might result in not having the required amount of storage available for the dataset. In such cases, the adaptable distributed storage should respond to the infrastructure that this is not permitted, as this would lead to data loss. In case that this is permitted, then it should re-distribute its data load, and inform the infrastructure that the node is ready to be released.				

Table 107 - Requirement (5) for Adaptable Distributed Storage

	Id	Level of detail	Type	Actor	Priority
	REQ-ADS-06	System	ENV	Data Engineer	DES
Name	Inform the re-deployment component regarding reconfigurations of the data fragments.				
Description	As it is up to the storage itself to decide its optimal configuration of its data load, the re-deployment component cannot be aware of possible reconfigurations that might affect the overall deployment of an application. Therefore, the storage should inform the re-deployment component about these actions.				
Additional Information	A message should be sent just before the re-configuration takes place, along with the setup, so that the re-deployment component can be notified and not consider possible outlier monitoring information coming from this subcomponent. During this time, the re-deployment component should not modify any deployments that rely on the data set that is being re-configured. When the reconfiguration is finished, the adaptable storage should notify the redeployment component again, for the latter to start				

	looking on the new monitoring information and decide upon possible redeployment of existing applications as well.
--	---

Table 108 - Requirement (6) for Adaptable Distributed Storage

	Id	Level of detail	Type	Actor	Priority
	REQ-ADS-07	System	ENV	Data Engineer	MAN
Name	Re-establish connectivity with the monitoring subcomponent when a horizontal scaling action takes place				
Description	The adaptable storage engine exports its monitoring data to a specific place where the Prometheus, part of the monitoring subcomponent of BigDataStack can periodically pull and gather this information. Prometheus can be configured on where to pull this information upon its initialization. However, in cases of a runtime redeployment that takes place after a horizontal scaling action, information regarding the newly deployed nodes should also reach the monitoring component.				
Additional Information	There should be a monitoring proxy of the adaptable storage that will have the responsibility of sending monitoring information to the target component. This proxy should encapsulate the details of the underlying deployment. It should gather all information of the data nodes, reconfigure itself to consider newly deployed data nodes, and send everything to the Prometheus.				

Table 109 - Requirement (7) for Adaptable Distributed Storage

	Id	Level of detail	Type	Actor	Priority
	REQ-ADS-08	System	ENV	Data Engineer	MAN
Name	Enable a deployment of the data node component using Kubernetes				
Description	As the infrastructure of BigDataStack uses Kubernetes for deploying the various application/platform components, the adaptable distributed engine must be able to deploy and configure additional data nodes via this technology.				
Additional Information	N/A				

Table 110 - Requirement (8) for Adaptable Distributed Storage

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-01	Software	FUNC	Application Engineer, Data Engineer	MAN
Name	Provide access to data stores via a single and common interface.				
Description	BigDataStack includes two different data stores: the LeanXcale relational data store and IBM object store. The dataset can be fragmented and distributed over the two data stores (historic data being moved to object				

	store). However, the application should be kept unaware of these internal data transfers. The application needs a common interface to submit queries, without having to specify where the data is stored.
Additional Information	A federation mechanism is required that will encapsulate the process of data retrieval from the two data stores. The LeanXcale access point will act as the federator between the relational and the Object Storage. The LeanXcale data base already provides a common JDBC interface for data connectivity. The federator will receive the query and execute it in both data stores. For the object store, the access would be via Spark SQL, with the assistance of Apache Hive for storing the metadata of the schema catalogue, which can also be transparently accessible via a JDBC interface. The federator will take into consideration the operations that can be supported in order to push down the operations accordingly. Regarding the relational store, all operations will be pushed down to the store. At the very end, the federator will merge the results and return back the result set. It shouldn't count data that appears in both data stores twice.

Table 111 - Requirement (1) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-02	System	DATA	Application Engineer, Data Engineer	MAN
Name	Move historical data from the relational data store to the object store.				
Description	Data ingested by the use cases will be stored into the relational datastore, as they are operational, in order to ensure data consistency in terms of ACID properties. After a configurable period, called the <i>freshness window</i> (which depends on the data set), the data becomes outdated and is no longer used by operational workloads. However, this historical data is still valuable and can be exploited by Big Data analytics algorithms. This data should be moved from the LeanXcale data base to the IBM object store.				
Additional Information	The LeanXcale data base provides a mechanism that allows to periodically produce a dumb snapshot of the modified data. This information will be transformed accordingly and will be pushed to an Apache Kafka queue. A Kafka based connector will, periodically pull this information and import the historical data to the object store.				

Table 112 - Requirement (2) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-03	Software	DATA	Application Engineer, Data Engineer	MAN
Name	Inform the LeanXcale data store when data are imported to the object store.				
Description	When data are is pulled from the operational datastore, the LeanXcale data base can drop them. However, due to the asynchronous design, the				

	LeanXcale data base cannot know when the data has been made available to the object store. As a result, the object store must inform the LeanXcale data base regarding the successful insertion of the data, so that the LeanXcale data base can safely drop these data.
Additional Information	One possible solution to deal with this requirement will be the introduction of marking the data to be transferred to the object store by additional timestamps. Data that is being flushed and exported to the object store can be marked that way, so that later, the object store can inform the LeanXcale data base that this bunch of data has been successfully imported. By doing so, the <i>federator</i> component can push down operations accordingly, and only request specific data from the underlying data stores. Data that are known to the LeanXcale database that has been previously uploaded to the object store, will not be retrieved by the <i>federator</i> and can be safely discarded by the <i>vacuum</i> process of the LeanXcale data base.

Table 113 - Requirement (3) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-04	Software	DATA	Application Engineer, Data Engineer	OPT
Name	Optimize query execution				
Description	The federator receives a query and executes it into the different stores. The federator will be based on the LeanXcale query engine. The latter provides a query optimizer, which allows it to examine the different execution plans that can be produced in order to execute a query. However, it has been implemented to evaluate plans to be executed locally. It should be extended in order to take into consideration the operations that can be pushed down to the object store, and whether or not it is worth for an operator to be pushed down, according to the response time of the execution from Spark SQL, the amount of data that will be retrieved to the federator etc.				
Additional Information	As every operation that can be supported by the object store will be pushed down to be executed locally, in order to avoid transferring a big amount of data through the network and process them in the query engine level, the implementation of this requirement corresponds to the following two aspects: the choose of the optimal strategy for executing the JOIN operation concerning data tables that are distributed and split to the two stores, and the redefinition of the query execution plan, in order for the query federator to exploit data locality and reduce the number of rows that will be retrieved and transferred from the object store via the network.				

Table 114 - Requirement (4) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-05	Software	DATA	Application Engineer, Data Engineer	OPT

Name	Optimize access to Object Storage.
Description	In order to perform analytics efficiently on Object Storage, a client-side caching/acceleration layer is needed. This is critical for a hybrid cloud scenario, where some of the customer data is on premise (potentially the LeanXscale data base and Spark) and some is in the cloud (potentially IBM COS). In such a scenario, when performing analytics, data needs to move from COS to Spark across the WAN, therefore minimizing the amount of data movement when part of the data is retrieved multiple times is of utmost importance. A similar scenario occurs in a multi-cloud environment, where a dataset may be distributed among more than one cloud, also requiring data transfer across the WAN for the purposes of analytics.
Additional Information	This complements data skipping and data layout techniques to further reduce the KPI measuring the number of bytes sent from Object Storage to Spark.

Table 115 - Requirement (5) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-06	Stakeholder	FUNC	Application Engineer, Data Engineer	MAN
Name	SQL Grammar extension				
Description	In order to better support the seamless, an extension of the SQL grammar is needed				
Additional Information	The grammar extensions will allow the database administrator to define that a data table can be split across the two datastores, and will allow him to provide additional information like the time window of the data slice, along with other configuration attributes like the minimum size of a data slice that is allowed to be moved, time frequency of the moving action etc.				

Table 116 - Requirement (6) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-07	Stakeholder	DATA	Application Engineer, Data Engineer	MAN
Name	Better decision of datasets partition				
Description	Currently slices of a dataset will start to be moved to the Object Store on a time basis (e.g., data older than 3 months). However, this is not flexible enough when the growth of the dataset is not known in advance. We often will prefer to leave the full dataset within the LeanXscale database if it will be under a given size.				
Additional Information	This requirement is very useful for the implementation of JOINS. Indeed, it is a typical case that one of the 2 tables being joined is small. In this case,				

	the JOIN can be implemented both in a simpler and more efficient way when the (small) table is fully stored at LeanXcale.
--	---

Table 117 - Requirement (7) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-08	System	DATA	Application Engineer, Data Engineer	OPT
Name	Handle JOINS for datasets distributed within both data stores				
Description	Currently, the seamless technology does not handle JOINS which limits the applicability of the seamless technology. The updated version of the query federator should support the JOIN operator thus being fully SQL compliant. The strategy for implementing this operator should consider that data has been split between two non-homogeneous datastores, and should exploit their locality, in order to avoid moving huge amounts of data over the network.				
Additional Information	As of the current release, JOINS are performed in the query engine level of the LeanXcale datastore. This is not efficient as it requires the data from both sides to be fetched into memory, and the JOIN must be performed at that level. The new JOIN operator must be able to push down the operator itself to the two datastores, joining data locally where possible, and only send data concerning the smaller datable to the object store, applying strategies like bind join and get the result. The JOIN operator must then perform a union over the intermediate results.				

Table 118 - Requirement (8) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-09	System	PER	Application Engineer, Data Engineer	MAN
Name	Ensure data consistency when a moving action is taking place				
Description	When data is moving from the operational store to the object store, data might either co-exist in both stores, or are non-existent in any store. The framework must be able to serve requests for data retrieval with no downtimes during this process, and the data should be consistent, meaning that the result of the execution of a query should be the same, no matter if the data are being moved.				
Additional Information	The operational datastore must not withdraw a data slice, until an acknowledgement of a persistence storage is being notified by the object store. In this case, data can co-exist in both stores. The Query Federator of the framework must take this into account, and re-write the queries to be executed in both stores accordingly in order to scan records on the visible data set in each store. In order to ensure data consistency when parallel transactions are being executed, before, during and after the data moving				

	process, it will rely on the transactional manager of the operational datastore.
--	--

Table 119 - Requirement (9) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-SDAF-10	System	PERF	Application Engineer, Data Engineer	OPT
Name	Efficiently drop a data slice from the operational datastore				
Description	Dropping data slice from the operational datastore will usually involve the deletion of numerous tuples that might affect the operational behaviour of the datastore, as it will push a lot of workload to its transactional engine. A most efficient way should be implemented.				
Additional Information	When the moving action should be performed, the Seamless Analytical Framework should inform the operational datastore to be prepared to drop that slice afterwards. In correspondence with the REQ-ADS-01, it will move this slice to specific data region, and additionally it will mark it as read-only. By doing so, no data modification operation will allowed be performed when the data is being moved, which is also aligned with the REQ-SDAF-09. When the acknowledge of the persistent storage of the slice in the object store is received, the data slice can be dropped from the operational datastore by removing the data region, with no further action from the transactional manager, as this data region has been already defined as read-only.				

Table 120 - Requirement (10) for Seamless Data Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-DQAI-01	Software	DATA	Data Engineer	MAN
Name	Data store connection				
Description	The Data Quality Assessment and Improvement module should be able to connect to the Data Source, to retrieve and update its tables as well as validate them.				
Additional Information	The connection is established via a common JDBC driver.				

Table 121 - Requirement (1) for Data Quality Assessment & Improvement

	Id	Level of detail	Type	Actor	Priority
	REQ-DQAI-02	Software	DATA	Data Engineer	OPT
Name	Data Augmentation				
Description	The Data Quality Assessment and Improvement module should be able to augment the original dataset, to design a new training set that alleviates the problem of class imbalance, for modelling purposes. This problem exists				

	because there are only a few corrupted examples in any given dataset, if most of the corpus is valid.
Additional Information	The data augmentation module will be able to ingest random typos, format errors and value swaps and is an optional component in the DQAI pipeline.

Table 122 - Requirement (2) for Data Quality Assessment & Improvement

	Id	Level of detail	Type	Actor	Priority
	REQ-DQAI-03	Software	DATA	Data Engineer	MAN
Name	Correlation				
Description	The Data Quality Assessment and Improvement module should be able to compute the correlation between several attribute columns in a dataset. This way we can discard non-informative columns, such as primary keys, that deteriorate the model's performance and accuracy. For example, in DANAOS dataset there might be a direct correlation between the engine's RPM and fuel consumption. This is something that we can leverage inside our model and check if the two values reported for RPM and fuel consumption have high probability to co-exist. On the other hand, primary keys do not offer any insight about any other column in the dataset. Thus, we only use them to update the probability score of a whole tuple in the end.				
Additional Information	The correlation metric that we use is the "uncertainty coefficient" or Theil's U, which is a non-symmetric correlation metric. This can uncover information that otherwise would be lost, if for example the metric used were a simple "Pearson correlation coefficient" or "Crammer's V."				

Table 123 - Requirement (3) for Data Quality Assessment & Improvement

	Id	Level of detail	Type	Actor	Priority
	REQ-DQAI-04	Software	DATA	Data Engineer	MAN
Name	Model storage				
Description	The Data Quality Assessment and Improvement module should be able to serialize and store the artefacts of the model.				
Additional Information	The artefacts contain the model itself as well as several other entities, such as tokenizers and parameter configuration.				

Table 124 - Requirement (4) for Data Quality Assessment & Improvement

	Id	Level of detail	Type	Actor	Priority
	REQ-DQAI-05	Software	DATA	Data Engineer	DES
Name	Scheduler				
Description	The Data Quality Assessment and Improvement module should be able to request for new data that were inserted in the data store. Thus, a scheduler will be implemented to make those requests periodically.				

Additional Information	The scheduler is implemented as a simple Python script.
-------------------------------	---

Table 125 - Requirement (5) for Data Quality Assessment & Improvement

	Id	Level of detail	Type	Actor	Priority
	REQ-DQAI-06	Software	DATA	Data Engineer	MAN
Name	Database ingestion module				
Description	The Data Quality Assessment and Improvement module should be able to insert the assessed data back in the data store.				
Additional Information	The ingestion module is sending assessed data requests directly to the transaction manager, bypassing the SQL layer. This makes the process significantly faster and more efficient.				

Table 126 - Requirement (6) for Data Quality Assessment & Improvement

	Id	Level of detail	Type	Actor	Priority
	REQ-RD-01	System	FUNC	Application Engineer	ENH
Name	Global decision tracker connection				
Description	A connection to the Global Decision Tracker (GDT) is needed for the Predictive & Process Analytics component.				
Additional Information	The information stored in GET is crucial to the implementation of this module.				

Table 127 - Requirement (1) for Predictive & Process Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-RD-02	Stakeholder	FUNC	Business Analyst, Data Engineer	ENH
Name	Connection to the Process Modelling Framework				
Description	A connection between this component and the Process Modelling Framework needs to be established, so information can be sent and received.				
Additional Information	The recommendations made by this component will be in real time, as the Business Analyst – Data engineer is modelling the process.				

Table 128 - Requirement (2) for Predictive & Process Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-RD-03	Software	FUNC	Application Engineer	ENH
Name	Data pre-processing				

Description	The data ingested by this component needs to be in an eXtensible Event Stream ²² (XES) file format. A tool is created, depending on the format of the Global Event Tracker.
Additional Information	The XES standard defines a grammar for a tag-based language whose aim is to provide designers of information systems with a unified and extensible methodology for capturing systems behaviours by means of event logs and event streams is defined in the XES standard. An XML Schema describing the structure of an XES event log/stream and an XML Schema describing the structure of an extension of such a log/stream are included in this standard. Moreover, a basic collection of so-called XES extension prototypes that provide semantics to certain attributes as recorded in the event log/stream is included in this standard.

Table 129 - Requirement (3) for Predictive & Process Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-RD-04	Software	FUNC	Application Engineer	ENH
Name	ProM framework				
Description	ProM is an extensible framework that supports a wide variety of process mining techniques in the form of plug-ins.				
Additional Information	The process mining techniques used will be utilized to derive metrics of the event log, to create the semantics needed between events for the recommendation process.				

Table 130 - Requirement (4) for Predictive & Process Analytics

	Id	Level of detail	Type	Actor	Priority
	REQ-CEP-01	System	FUNC	Application Engineer, Data Engineer	MAN
Name	Manage data from different sources to generate alarms if required.				
Description	The Complex Event Processing (CEP) will process data on the fly coming from sensors. Each sensor sends events each minute. CEP will analyse the data according to a set of rules and generate alarms.				
Additional Information	The processing will be both stateless and over windows of time and number of events.				

Table 131 - Requirement (1) for Complex Event Processing

	Id	Level of detail	Type	Actor	Priority
	REQ-CEP-02	System	FUNC	Application Engineer, Data Engineer	MAN

²² <http://www.xes-standard.org/>

Name	Send alarms and data from each node of the distributed environment to the data centre.
Description	Once metrics have been analysed, the CEP will send the alarms and the data to a central location (data centre).
Additional Information	The CEP will run on nodes of a geographically distributed environment.

Table 132 - Requirement (2) for Complex Event Processing

	Id	Level of detail	Type	Actor	Priority
	REQ-CEP-03	System	PERF	Application Engineer, Data Engineer	MAN
Name	Data from distributed nodes is aggregated at a central location.				
Description	Further processing over remote data will be performed at a central location.				
Additional Information	The CEP processing will scale to tens streams coming from different remote sources.				

Table 133 - Requirement (3) for Complex Event Processing

	Id	Level of detail	Type	Actor	Priority
	REQ-CEP-04	System	PERF	Application Engineer, Data Engineer	ENH
Name	Store complex event data on the data store				
Description	The CEP will store the data at the rate is being produced.				
Additional Information	Both CEP and LX shall run at the same location.				

Table 134 - Requirement (4) for Complex Event Processing

8 Dimensioning, Modelling & Interaction Services Requirements

To facilitate the traceability between requirements and the design decisions as well as technical challenges addressed for the Dimensioning, Modelling & Interaction Services of BigDataStack, these requirements appear as-is in D5.2²³, attached to the design of the components implementing them. Nevertheless, the present section should be considered the single source of all requirements for the present capability and be considered the master version of them in case of discrepancies.

	Id	Level of detail	Type	Actor	Priority
	REQ-PMF-01	System	USE	Business Analyst	MAN
Name	Good process modelling UX				
Description	The system should guide the users to complete the business diagram / flow with easy steps. It should clearly indicate what connections–interactions are possible and provide comprehensive error messages.				
Additional Information	N/A				

Table 135 – Requirement (1) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
	REQ-PMF-02	System	FUNC	Data Scientist, Business Analyst	MAN
Name	Multi-user support				
Description	Multiple users should be able to use the Process Modelling Framework and create diagrams at the same time. It should also support different roles: business analysts and data analysts. A business analyst will define a process in a higher level and a data analyst will provide the concrete implementations				
Additional Information	N/A				

Table 136 – Requirement (2) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
	REQ-PMF-03	System	FUNC	Business Analyst	MAN
Name	Process workflow creation				
Description	A business analyst should be able to create a process workflow in a higher level. The analyst will select nodes from a catalogue and using a drag-and-drop interface will link them together to create the flow.				

²³ D5.2 – WP5 Scientific Report and Prototype Description – Y2 (due M23).

Additional Information	N/A
-------------------------------	-----

Table 137 – Requirement (3) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
	REQ-PMF-04	System	FUNC	Data Scientist	MAN
Name	Process workflow configuration				
Description	The data analyst should be able to configure a process workflow with all the required details. The data analyst will set up the nodes parameters and define the rules for moving from one node to another.				
Additional Information	N/A				

Table 138 – Requirement (4) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
	REQ-PMF-05	Software	FUNC	Data Scientist	MAN
Name	Process workflow export				
Description	The data analyst should be able to export/edit/import the process workflow in BigDataStack format.				
Additional Information	The default format of the export will be in JSON. It will include information regarding the flows and their interconnections. Alternative export formats (e.g., YAML, <i>Dockerfile</i>) will be considered based on the requirements of other components.				

Table 139 – Requirement (5) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
	REQ-PMF-06	System	FUNC	Business Analyst	MAN
Name	Support for end-to-end (in terms of process workflow) objectives				
Description	The business analyst should be able to define end-to-end service-level objectives (SLO), e.g., end-to-end response time or throughput.				
Additional Information	Indicatively, these objectives may refer to the specification of minimum required quantitative results obtained by the analytic algorithms (i.e., precision, recall, f-score, accuracy, mean squared error, etc.), the specification of an execution time limit or minimum throughput per workflow, or the minimum qualitative results per analytic task compared to the ground truth (i.e. binary / multi-class classification, recommender system, regression, etc.).				

Table 140 – Requirement (6) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
--	-----------	------------------------	-------------	--------------	-----------------

	REQ-PMF-07	System	FUNC	Business Analyst	MAN
Name	Node constraints				
Description	The business analyst should be able to set apply constraints per node and/or activity of the workflow.				
Additional Information	N/A				

Table 141 – Requirement (7) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
	REQ-PMF-08	System	FUNC	Business Analyst	MAN
Name	Node edge constrains				
Description	The business analyst should be able to apply constraints / parameters per node edge (i.e. connections between activities of the workflow).				
Additional Information	N/A				

Table 142 – Requirement (8) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
	REQ-PMF-09	System	FUNC	Data Scientist, Business Analyst	MAN
Name	Graph Saving and Editing				
Description	The user should be able to export/import/edit/save the generated graph.				
Additional Information	N/A				

Table 143 – Requirement (9) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
	REQ-PMF-10	Stakeholder	FUNC	Data Scientist, Business Analyst	MAN
Name	Arsenal of Services				
Description	The component should provide an enriched and complete collection of services that will fulfil all Process Modelling Scenarios				
Additional Information	N/A				

Table 144 – Requirement (10) for Process Modelling Framework

	Id	Level of detail	Type	Actor	Priority
	REQ-PM-01	Stakeholder	FUNC	Application Engineer	MAN
Name	Compatibility with Process Modelling				

Description	The Process Mapping component can process the output of Process Modelling, in order to select appropriate algorithm(s) for the specific processes modelled in the process modelling environment.
Additional Information	This requirement practically ascertains that the two components (Process Modelling and Process Mapping) are compatible and that the output of the first can be consumed by the second.

Table 145 – Requirement (1) for Process Mapping

	Id	Level of detail	Type	Actor	Priority
	REQ-PM-02	Stakeholder	FUNC	Application Engineer	MAN
Name	Extraction of metadata				
Description	Given a dataset, extract a set of metadata that is enough in order to discover similarities between datasets, regarding the underlying data distributions and other statistical properties.				
Additional Information	The metadata should cover at least statistical and information-theoretic characterization of a given dataset.				

Table 146 – Requirement (2) for Process Mapping

	Id	Level of detail	Type	Actor	Priority
	REQ-PM-03	Stakeholder	FUNC	Application Engineer	MAN
Name	Build and maintain a meta-knowledge repository				
Description	Collect and store information about datasets, metadata, and the performance of ML algorithms that have been executed on the datasets. This information is referred to as meta-knowledge, because it is essentially knowledge about the learning process. This meta-knowledge repository should be used for <i>meta-learning</i> , which is defined as the study of methods that exploit meta-knowledge to obtain efficient models and solutions by adapting machine learning processes.				
Additional Information	The meta-knowledge repository is augmented with information about the execution of ML algorithms on new datasets.				

Table 147 – Requirement (3) for Process Mapping

	Id	Level of detail	Type	Actor	Priority
	REQ-PM-04	Stakeholder	FUNC	Application Engineer	MAN
Name	ML algorithm selection				
Description	Given a machine learning task, a dataset, and a set of available ML algorithms that can handle the given task, select (or recommend) the subset of ML algorithms with best performance.				

Additional Information	It assumes the availability of a pool of ML algorithms (e.g., a ML library) and an execution environment for running ML algorithms on different datasets and evaluating their result quality.
-------------------------------	---

Table 148 – Requirement (4) for Process Mapping

	Id	Level of detail	Type	Actor	Priority
	REQ-PM-05	Stakeholder	FUNC	Application Engineer	MAN
Name	Hyperparameter Tuning				
Description	Given a specific machine learning algorithm (determined by algorithm selection), automatically select optimal values for the input parameters.				
Additional Information	The hyperparameter tuning process is performed for (a) a given machine learning task (e.g., clustering, classifications, etc.), and (b) the input parameters of each algorithm. Its objective is to automatically compute appropriate values for the input parameters, which will be used for the invocation of the specific ML algorithm.				

Table 149 – Requirement (5) for Process Mapping

	Id	Level of detail	Type	Actor	Priority
	REQ-PM-06	System	FUNC	Application Engineer	MAN
Name	Data retrieval from the storage engine				
Description	The Process Mapping component can connect to and query the LeanXcale datastore of the storage engine, to retrieve input datasets or filtered datasets.				
Additional Information	The two sub-tasks of Process Mapping, namely ML algorithm selection and hyperparameter tuning, should be considered dataset-specific, i.e., they are executed for a given input dataset. The Process Mapping component can access the datasets stored in the LeanXcale datastore in order to provide an integrated solution at system level.				

Table 150 – Requirement (6) for Process Mapping

	Id	Level of detail	Type	Actor	Priority
	REQ-PM-07	Stakeholder	FUNC	Application Engineer	MAN
Name	Output recorded in the playbook				
Description	The output of the Process Mapping component is used to update the playbook.				
Additional Information	The output of the Process Mapping component is written in a form that is compatible with the playbook format, in order to enable the execution engine to have all the necessary information for the invocation of the selected machine learning algorithm.				

Table 151 – Requirement (7) for Process Mapping

	Id	Level of detail	Type	Actor	Priority
	REQ-DTK-01	Software	FUNC	Data Scientist, Business Analyst	MAN
Name	Describe data mining and analysis processes through data workflows				
Description	Support for the description of data mining and analysis processes, interconnected to each other in terms of input/output data streams/objects. The corresponding metadata and an algorithms taxonomy for the categorisation of the analytic processes, type of data and connection details will be used to facilitate the description of individual nodes.				
Additional Information	The playbook must be represented in the form of a descriptor (e.g. through a YAML file) that can be incorporated into the Dimensioning Workbench as well as the Dynamic Orchestrator.				

Table 152 – Requirement (1) for Data Toolkit

	Id	Level of detail	Type	Actor	Priority
	REQ-DTK-02	System	FUNC	Data Scientist, Business Analyst	MAN
Name	Express data workflows through graphs using nodes and edges				
Description	Data workflows are represented in the form of an analysis application graph that includes the set of individual processes as nodes of the graph along with their binding/dependencies in the form of virtual links (i.e. edges). The links may include properties representing constraints, KPIs or objectives which are desirable at specific analytic stage.				
Additional Information	N/A				

Table 153 – Requirement (2) for Data Toolkit

	Id	Level of detail	Type	Actor	Priority
	REQ-DTK-03	System	USE	Business Analyst	MAN
Name	Validate graph through chain-ability constraints				
Description	This requirement resolves chain-ability constraints through different nodes in the data workflows. The target is to produce a valid graph. This is the reason why a set of checks will be performed to meet these prerequisites. If these prerequisites are not met, the graph is not considered valid.				
Additional Information	N/A				

Table 154 – Requirement (3) for Data Toolkit

	Id	Level of detail	Type	Actor	Priority
	REQ-DTK-04	System	USE	Business Analyst	MAN
Name	Link valid graphs with viable executables for Big Data analytic processes				

Description	This step links the graph with the actual executables. In order to cope with the problem of vendor lock-in format of the executable, the Docker container standard has been chosen. Therefore, the actual container corresponding to each executable task or subprocess will be pulled from the appropriate docker image repository.
Additional Information	N/A

Table 155 – Requirement (4) for Data Toolkit

	Id	Level of detail	Type	Actor	Priority
	REQ-PG-01	System	FUNC	Application Engineer	MAN
Name	Ingest Playbook				
Description	The Data Toolkit sends to the Pattern Generation a Playbook containing the graph of the user's application. The Pattern Generation receives the playbook and initiates creation of candidate deployment patterns.				
Additional Information	N/A				

Table 156 – Requirement (1) for Pattern Generator

	Id	Level of detail	Type	Actor	Priority
	REQ-PG-02	System	FUNC	Application Engineer	MAN
Name	Load Hardware Directory (File)				
Description	To produce candidate deployment patterns, Pattern Generation needs to know what hardware is available to deploy the components of the user's application upon. Initial versions will load this information from a static file.				
Additional Information	N/A				

Table 157 – Requirement (2) for Pattern Generator

	Id	Level of detail	Type	Actor	Priority
	REQ-PG-03	System	FUNC	Application Engineer	MAN
Name	Service-Hardware Mapping (1-1)				
Description	The main process in Pattern Generation is mapping the different components (services) to potentially suitable hardware. The first version of this functionality produces only 1-1 mappings, i.e. one service is mapped to one piece of hardware (e.g. machine).				
Additional Information	N/A				

Table 158 – Requirement (3) for Pattern Generator

	Id	Level of detail	Type	Actor	Priority
	REQ-PG-04	System	FUNC	Application Engineer	MAN
Name	Service-Hardware Mapping (1-M)				
Description	The main process in Pattern Generation is mapping the different components (services) to potentially suitable hardware. The second version of this functionality produces only one to many mappings, i.e. one service can be mapped to multiple piece of hardware (e.g. spread over multiple machines). This may be advantageous in cases such as where a single 'big' machine is more expensive than multiple smaller machines.				
Additional Information	N/A				

Table 159 – Requirement (4) for Pattern Generator

	Id	Level of detail	Type	Actor	Priority
	REQ-PG-05	System	FUNC	Application Engineer	DES
Name	Service-Hardware Mapping (M-1/Pods)				
Description	The main process in Pattern Generation is mapping the different components (services) to potentially suitable hardware. The third version of this functionality produces only many to one mapping, i.e. multiple services can be co-located on a single piece of hardware. This may be advantageous when services perform high-volume data transfers that would be expensive over a network.				
Additional Information	N/A				

Table 160 – Requirement (5) for Pattern Generator

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-01	System	PERF	Data Scientist	MAN
Name	Response Time and Workload				
Description	The service provided by the data applications (e.g. recommender system) must have enough speed so consumers will not notice the time taken by the request. This implies that the Data Scientist should be able to dictate what are the required levels of QoS, selecting them from available metrics and appropriate levels for them.				
Additional Information	This requirement poses initially the feature of metric selection and insertion at the Data Toolkit layer, for the Data Scientist to express their desires. Then the annotated Playbook gets passed to the following components (primarily ADW). Inside the Application Dimensioning Workbench, an initial candidate solution set is created, its estimated QoS level is enriched and the solution set is returned to the Data Scientist for final selection. Workload features				

	(e.g. maximum/average etc. number of concurrent users) should also be able to be specified for the system to estimate the anticipated QoS levels for the desired range of application level workload. This indicates that per category of data service or data service plus analytics function a suitable selection of workload and QoS metrics should be performed and supported across the system (including also other components like monitoring).
--	--

Table 161 – Requirement (1) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-02	System	PERF	Application Engineer	MAN
Name	Scalability and configurability of stress tests for load injection				
Description	The system should have knowledge of a mapping between workload and QoS levels of the data services and algorithms (in order also to support REQ-SY-DW-02). Therefore, it should be able to launch stress tests against the data services that can easily scale to support the client sizes needed. Furthermore, different parameters of workload should be able to be determined.				
Additional Information	Given that different data services exist in the project ecosystem, different baseline benchmarking tools should be identified per case. Following their selection, they need to be configured based on the respective workload parameters and scaled based on an abstracted generic approach (e.g. Docker containerization and Docker swarm approach).				

Table 162 – Requirement (2) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-03	System	FUNC	Application Engineer	MAN
Name	Dimensioning output				
Description	The Application Dimensioning Workbench (ADW) must provide a list of candidates dimensioning suggestions along with the expected QoS levels towards the ADS Deploy component (and eventually the Application Engineer role), for the former to filter them based on an extra set of criteria and the latter to perform the final selection.				
Additional Information	Upon reception of the playbook with the service graph, ADW needs to estimate QoS level based on the results obtained through REQ-SYS-DW-02 and populate the respective fields. The operation should be offered through a REST service interface for automating the process and hiding complexities.				

Table 163 – Requirement (3) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-04	Stakeholder	FUNC	Application Engineer	MAN

Name	Monitoring requirements for dimensioning
Description	The Application Dimensioning Workbench (ADW) must have a means to obtain monitoring information from the deployed data services and application components for a given deployment to extract training data for the performance models. The rationale of the requirement is that for every needed metric (workload oriented e.g. number of current users, requests etc. or QoS oriented e.g. response time, throughput) in the model, the respective endpoint should exist from which the monitoring component would extract metrics values. This applies to both actual runtime and benchmarking phase.
Additional Information	Relevant Tools affected: Data services, application components, triple monitoring engine.

Table 164 – Requirement (4) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-05	Software	MAN	Application Engineer	MAN
Name	Load injector <i>dockerization</i>				
Description	To support a generic load injection process as indicated by REQ-SY-DW-02, “dockerization” of the respective load generators per type of service needs to be performed. Thus, a specific Docker container image per needed load generator tool needs to be provided, along with a unified process for feeding the per case load description file based on the Docker API and configuration process.				
Additional Information	N/A				

Table 165 – Requirement (5) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-06	Software	FUNC	Application Engineer	MAN
Name	Service structure specification				
Description	The service graph specification coming as input from the Process Modelling and Data Toolkit should follow the Docker Compose specification, to be understandable by the Application Dimensioning Workbench (ADW). Next, the Dimensioning phase should add the respective candidate resource deployment options as additional custom metadata in the file to be used by the Deployment selection. The same applies for the benchmarking runs, which should be based on the same format (even without the inclusion of the PM and Data Toolkits).				
Additional Information	All requirements needed for deploying the benchmarking environment should be described using this common agreed standard.				

Table 166 – Requirement (6) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-07	Software	FUNC	Application Engineer	MAN
Name	Representative nature of gathered data samples				
Description	In order to create representative and accurate performance models, dataset creation from benchmarking should consider different conditions such as applied workloads, configuration aspects of the service, deployment options etc. In this way different bottlenecks may be examined, and the final decision making can be adapted per case of service usage.				
Additional Information	N/A				

Table 167 – Requirement (7) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-08	Stakeholder	PERF	Application Engineer	ENH
Name	Deployment time for stress tests				
Description	The overhead added by the benchmarking setup should be negligible and not included in the measurement process.				
Additional Information	Since the deployment phase is done in a containerized manner, the time used in instructions different than launching the benchmark or storing data should not be significant.				

Table 168 – Requirement (8) for Application Dimensioning Workbench

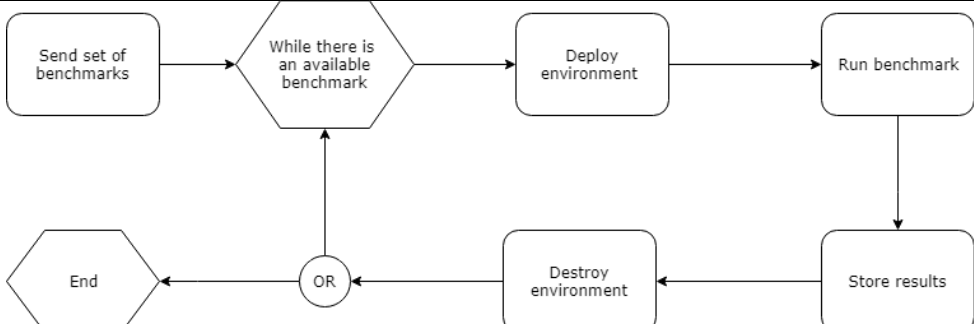
	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-09	Software	FUNC	Application Engineer	ENH
Name	Benchmarking Workflow implementation				
Description	During the benchmarking phase, there should be a controlled way the various combinations described in REQ-SY-DW-02 and REQ-SO-ADW-03 are enforced during an automated process in order to ease data collection.				
Additional Information	 <pre> graph TD A[Send set of benchmarks] --> B{While there is an available benchmark} B --> C[Deploy environment] C --> D[Run benchmark] D --> E[Store results] E --> F[Destroy environment] F --> G((OR)) G --> H{End} </pre>				

Table 169 – Requirement (9) for Application Dimensioning Workbench

The following requirements for the Application Dimensioning Workbench were added at M22:

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-10	System	FUNC	Application Engineer	MAN
Name	Trace driven load simulation				
Description	Except for the parameter sweep definition of a stress test that is used in the context of REQ-ADW-03, the Application Dimensioning Workbench (ADW) user must also be able to define a trace file with a time series of load aspects for the framework to undertake its sequential implementation, simulating a historical fluctuation of demand for the service.				
Additional Information	Trace definition should be made as simple as possible, for example through uploading a trace file on a GitLab account. Trace file structure should also be defined (e.g. one row per setup). Since the deployment phase is done in a containerized manner, the time used in instructions different than launching the benchmark or storing data should not be significant.				

Table 170 – Requirement (10) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-11	System	FUNC	Application Engineer	MAN
Name	Enablement of parallel and isolated modes of execution				
Description	Application Dimensioning Workbench (ADW) load injection can be configured to launch a variety of configurations under the same test setup (e.g. in the parameter sweep definition of a test). The test instances stemming from the parameter combinations must be able to be performed in either a sequential, isolated mode, that guarantees repetitiveness of an experiment under the same conditions, or in a parallel mode, in order to check aspects of multitenant applications.				
Additional Information	The tool logic should prevent different modes from being applied at the same time, thus accession to the stress test cluster should be considered as a race condition and handled appropriately.				

Table 171 – Requirement (11) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-12	System	USE	Application Engineer	DES
Name	Enhanced benchmark results filtering				
Description	Results acquired through the benchmarking may be queried in multiple, less or more advanced forms. In the case of more advanced ones, setup of the				

	experiment and how closely it relates to a specific desired deployment option should be considered; while an additional level of complexity may be included if the query includes a given metric goal and relevant tolerance for the returned results (in terms for example of a percentage coverage of the goal).
Additional Information	Definition of the aforementioned parameters should be included in the available interfaces towards the platform so that they can be set by the user (i.e. through a REST API, data toolkit annotations, the pattern generator candidate patterns or the end user UI of the Load Injector tool).

Table 172 – Requirement (12) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-ADW-13	Software	MAN	Application Engineer	DES
Name	Functionality offered through REST APIs				
Description	In order to further automate the various processes such as result querying, test setup submission etc, the main functionalities that relate to these processes should be offered via a RESTful API, so that they can be included in relevant software implementations.				
Additional Information	This mean of exposure does not reduce the UI based implementation but can be used to further extend the rationale and launch of the relevant experiments.				

Table 173 – Requirement (13) for Application Dimensioning Workbench

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-01	System	USE	Data Scientist, Business Analyst, Application Engineer	MAN
Name	Interactive and Responsive UI				
Description	The system should provide an interactive UI that should adapt to different devices and displays in order to provide a proper operation of the solution and a good user experience.				
Additional Information	N/A				

Table 174 – Requirement (1) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-02	System	FUNC	Application Engineer	MAN
Name	Automatic graph selection				
Description	Appropriate graphs and reports should automatically be selected for				

	different data sets.
Additional Information	N/A

Table 175 – Requirement (2) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-03	System	FUNC	Application Engineer	MAN
Name	Live data for different data sources				
Description	The system should be able to display live data obtained from application probes, resource probes and data operations probes.				
Additional Information	Adaptable selection of sources should be possible both in terms of application, resources or data operations, as well as in terms of the datasets selected and visualized per each one of these cases. Combinations should also be enabled.				

Table 176 – Requirement (3) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-04	System	FUNC	Business Analyst	MAN
Name	The system should be able to consume/integrate with the Process Modeller Component.				
Description	Business Analyst can create graphs and export/edit/import them later.				
Additional Information	N/A				

Table 177 – Requirement (4) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-05	System	FUNC	Data Scientist	MAN
Name	The system should be able to consume/integrate Data Toolkit Component.				
Description	Data Analyst can use the output provided by Process Modeller Component to apply Data Analyst Logic and enrich it				
Additional Information	N/A				

Table 178 – Requirement (5) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-06	System	FUNC	Application Engineer	MAN
Name	The system should be able to integrate Benchmarking API.				
Description	Application owner can define tests by providing the proper playbook and				

	extract metrics
Additional Information	N/A

Table 179 – Requirement (6) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-07	System	FUNC	Application Engineer	MAN
Name	The system should be able to integrate Application Dimensioning Workbench.				
Description	Application owner imports a playbook produced by the Data Toolkit Component and choose assisted Mode Deployment to get Deployment Recommendations. These recommendations can be automatically deployed and monitored.				
Additional Information	N/A				

Table 180 – Requirement (7) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-08	System	FUNC	Application Engineer	MAN
Name	The system should be able to monitor the Deployed Application and receive notifications regarding QoS that are violated by the Dynamic Orchestrator and ADS-Ranking.				
Description	Monitoring Screen is provided in the Application Owner View for each application deployed.				
Additional Information	N/A				

Table 181 – Requirement (8) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-09	System	FUNC	Application Engineer	MAN
Name	The system should be able consume and visualize datasets provided by the Data Quality Assessment Component.				
Description	User can extract information provided by the above component.				
Additional Information	N/A				

Table 182 – Requirement (9) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-10	System	FUNC	Application Engineer	MAN

Name	The system should be able consume and visualize datasets provided by the Maintenance component.
Description	User can extract information provided by the above component.
Additional Information	N/A

Table 183 – Requirement (10) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-11	System	FUNC	Application Engineer	MAN
Name	The system should be able consume and visualize graphs from CEP1 and CEP2 components.				
Description	User can extract information provided by the above components.				
Additional Information	N/A				

Table 184 – Requirement (11) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-12	System	USE	Application Engineer	MAN
Name	The system should be able to provide different Views of the UI depending on the user role.				
Description	Different Components should be visible and accessed depending of the role that each user has. This role can be Business Analyst, Data Analyst, Application Owner and Administrator. Access to any view of the above requires authentication of the user.				
Additional Information	N/A				

Table 185 – Requirement (12) for Adaptable Visualizations

	Id	Level of detail	Type	Actor	Priority
	REQ-AVI-13	System	FUNC	Data Scientist, Business Analyst, Application Engineer	MAN
Name	User Authentication				
Description	Authentication of the User should be performed once upon logging in the platform. Any additional authentication for individual components should happen in the background without further user interaction.				
Additional Information	N/A				

Table 186 – Requirement (13) for Adaptable Visualizations

9 Baseline Technologies

This chapter presents the baseline technologies to be used to introduce the ground for the proposed work and to ensure that non-conflicting (in terms of functional properties and integration with other components) technologies will be used. It does not simply state some state-of-the-art technologies, but rather link them under the context of BigDataStack and what BigDataStack can obtain from them as a baseline set of technologies.

9.1 Computing Resources Management

Public clouds (such as **GCP**²⁴ or **IBM Cloud**²⁵) offer infrastructure-as-a-service (IaaS). IaaS is a pay-per-use service model to consume virtual computational (e.g., virtual machines), storage and networking resources. This service provides high levels of QoS and wide variety of resource types (e.g., machines with different memory, CPU and acceleration chipset features).

OpenStack²⁶ is a set of software components to setup and manage public and private clouds. It lets companies to create an IaaS in their data centre. OpenStack lets you add servers, network and storage components easily and efficiently to a cloud. It controls large pools of compute, storage, and networking resources throughout a data centre, managed through a dashboard or via the OpenStack API.

Container management/orchestration platforms are becoming popular solutions to provision resources to and manage cloud applications and services. Some examples are **Kubernetes**²⁷, **Docker Swarm**²⁸, **Amazon ECS**²⁹, or **Mesosphere Marathon**³⁰. These typically provide an API for developers to upload, organize, run, scale, manage and stop containers. They also normally provide a Command Line Interface (CLI) and a web console to configure and monitor the performance of the services and resources of the platform, such as:

- container deployment and configuration,
- performance monitoring,
- cluster management and scaling,
- logging, and
- container lifecycle management.

Kubernetes is the most popular and widespread container orchestration platform; its main purpose is to schedule container (process) execution in a single machine or a cluster of machines and optimize (maximize the utilization) of the resources of the cluster. Everything in the platform is treated as an API object representing a concrete instance of a resource type in the cluster, notable resources include:

- Kubernetes Pods are the simplest object to be managed by Kubernetes. A pod is a group of related containers (processes) collocated on a Kubernetes cluster's node, sharing

²⁴ GCP (Google Cloud Platform). <https://cloud.google.com>

²⁵ IBM Cloud. <https://www.ibm.com/cloud/>

²⁶ Open Stack. <https://www.openstack.org/>

²⁷ Kubernetes. <https://kubernetes.io/>

²⁸ Docker Swarm. <https://docs.docker.com/engine/swarm/>

²⁹ Amazon ECS (Elastic Container Service). <https://aws.amazon.com/en/ecs/>

³⁰ Mesosphere Marathon. <https://mesosphere.github.io/marathon/>

storage/network and a specification for how to run the containers³¹. Replica sets are the number of pod instances running and represent the way to scale out/in deployments.

- Kubernetes Namespaces support multiple virtual clusters on the same physical cluster. Namespaces are used to organize objects in a cluster and provide a way to divide cluster resources to isolate environments and better ensure Quality of Service (QoS).
- Kubernetes Services describe how to access sets of pods and can describe ports and load-balancers.

Kubernetes scheduler is built around the concept of managing CPU and memory resources at a container level. Every Kubernetes cluster' node (instances to schedule containers to) is assigned an amount of schedulable memory and CPU. At deployment time, every container specifies how much memory and CPU it will request. And the scheduler finds the best fit given the allocated CPU and memory on the nodes.

Kubernetes defines the CPU and MEMORY resources using basic constructs [53]: The *request value* specifies the min value you will be guaranteed; the *limit value* specifies the max value you can consume (i.e., a CPU quota or a memory limit) and the value applications should be tuned to use.

Kubernetes defines several different quality of service (QoS) tiers based on how request and limit are specified [53]: *Best-Effort*, *Guaranteed* and *Burstable*.

9.2 Storage Resources Management

As we base our infrastructure on Kubernetes (see previous section), we need to tackle usage of storage by the application through the Containers Orchestrators perspective.

The current model for persistent storage for containers is settling out on attaching volumes (LUNs) to a container and formatting that volume with a file system. Through the container engine, the file system is exposed as a mount point within the container. Storage could be taken from local disks (e.g. a volume created using an LVM) or from external storage presented to the container server/node.

To make it easier to plug in storage interfaces into the Container Platform, K8S created the CSI. The aim of the Container Storage Interface (CSI) is to provide a common standard to connect container orchestration platforms (COs) like Kubernetes, Docker Swarm and Mesos to a plugin and ultimately to persistent storage (see previous section).

Theoretically, with a standardised communication protocol, storage vendors will only need to write a plugin to a single specification. CSI sets out the definition of how to talk to the plugin; exactly how the plugin is managed or operates is up to the storage provider. CSI provides the following capabilities.

- Dynamic provisioning and decommissioning of volumes.
- Attachment and detachment of volumes from a host node.
- Mounting and unmounting of a volume from a host node.

This technology enables us in BigDataStack to create flexible and dynamic management for the storage resources needed by the different workloads of the applications.

³¹ <https://kubernetes.io/docs/concepts/workloads/pods/pod/>

9.3 Data-driven Network Management

Data-driven network management regards a set of functions related to optimal network setup to cover the operational requirements of analytic processes in terms of efficient data exchange. In most of the cases, analytic process deployments are realised within a data centre (DC) environment and do not impose strict requirements in terms of layer-3 routing mechanisms. Optimal network flows' setup and management must be realised within a DC environment. Towards this direction, Software Defined Networking Mechanisms (SDN) can be applied along with network management mechanisms (e.g. enforcement of network policies) supported by the Orchestration Engine.

For instance, in case of Kubernetes, a network policy is a specification of how groups of pods can communicate with each other and other network endpoints. Network Policy resources use labels to select pods and define rules which specify what traffic should be allowed. In case of application of SDN-based mechanisms, **Kubernetes Network Policies**³², **Istio**³³ and **Kuryr**³⁴ are currently considered as the dominant network engineering and communication means that enable to monitor traffic at network, transport and service layers. The latter enables the deployment of special sidecar proxies throughout the cloud infrastructure which intercept all network communication among the microservices running in the underlying environment.

The adopted set of solutions for Data-driven Network Management is going to be well-bound to the orchestration solution to be designed and implemented within BigDataStack, guaranteeing the support of novel and intelligent network management mechanisms through the enforcement of rules specifying what traffic can reach selected services, applicable to data-intensive, network-demanding and safely ensured processes.

9.4 Dynamic Orchestrator

Orchestration (as an infrastructure management service) refers to the enablement and the coordinated handling of various optimizations inside the platform. Examples of such optimizations are the placement (or allocation) of tasks to computing resources, decisions regarding parallelization degrees of parallelizable tasks/services, load balancing, algorithm selection, and more. In the state of the art, such optimizations are handled in a way that we call "service-driven". This means that the optimization functions, the criteria, and the system setup are built around basic features such as CPU power, network bandwidth, and task/service requirements to optimize certain metrics (e.g. latency) with respect to the examined service.

Related works [46][47] exploit obvious known synergies such as the fact that running tasks on low-CPU nodes can increase processing time or the fact that overloading certain links can create bottlenecks. Apart from the fact that such concrete optimizations have rarely been handled homogeneously or investigated in a common context, there are also gaps towards making them data-driven rather than service-driven.

To support data-driven overall orchestration and data-driven solutions of specific optimization problems (e.g. placement), we propose the combination of Reinforcement Learning with domain heuristics that define the synergies between characteristics of data

³² <https://kubernetes.io/docs/concepts/services-networking/network-policies/>

³³ <https://istio.io/>

³⁴ <https://wiki.openstack.org/wiki/Kuryr>

analytics and system KPIs, e.g. functions that represent how data I/O volumes affect the CPU-intensity of certain tasks etc.

On the one hand, Reinforcement Learning (RL) - a machine learning technique in which an agent can interact and learn from the environment in which it is located - provides us with a framework to define flexible and customized orchestration for applications and services running on BigDataStack. Multiple works have shown the efficiency of RL to balance QoS [54][55] and configuration of systems [56]; however, the use of RL to orchestrate applications and services to comply with their requirements has not been addressed yet to our knowledge.

On the other hand, the incorporation of domain heuristics for the orchestration allows us to exploit well-known synergies and trade-offs throughout all applications and services until enough experience is gathered to customize the orchestration for the specific application/service.

The Dynamic Orchestrator will communicate and cooperate with other controllers to monitor, learn and change the deployment of running applications and services when it is needed, improving their reliability and performance. The technologies that will support the Dynamic Orchestrator implementation and integration are the following:

- The component will be developed using the programming language **Python**.
- **Prometheus** will provide monitoring metrics of the system, applications and services.
- **RabbitMQ** for the communication with other components.
- **Kubernetes** for application placement and container orchestration.
- **LeanXcale** will provide an interface for data services monitoring and deployment.
- **OpenFlow** for network functions recompilation.

9.5 Triple Monitoring

To be able to maintain a good quality-of-service (QoS) and perform effective adaptation based on the environment or data changes over an application's lifetime, metrics need to be taken continuously and exposed to any components involved in the evaluation of QoS and adaptation. In the context of BigDataStack, tracking information will be performed by the Triple monitoring engine. Three different groups of metrics need to be tracked: infrastructure information, data operations (data produced by applications running on the platform) and all data involved in database transactions.

Since these metrics are produced by applications with different purposes, specifications, functionalities and technologies, two approaches will be used, the first is to use probes to directly ingest metrics into the monitoring collector. The second approach is to provide a sanitizer to prepare metrics conforming with the specification of the collector and ingest them. This sanitizer will act as a unified API.

The triple monitoring engine has an input REST API which is an entry point of the system and an output REST API for exposing data to each application's data consumer. The monitoring system should provide an efficient and fast way of transferring metrics from the input to the manager that handles the engine action logic. The large number of metrics from different sources must be organized chronologically and presented in a correct format for their visualization. We use two main technologies:

Prometheus is a technology for monitoring management, which includes metrics collection facilities. This technology will be very convenient for the following reasons³⁵:

- Powerful queries: A flexible query language (NoSQL based) allows slicing and dicing of collected time series data.
- Efficient storage: Prometheus stores times series in memory and on local storage in an efficient custom format. Scaling is achieved by function sharing and federation.
- Extensive integration: Many existing exporters allow bringing data from third-party application to its collector.
- Push gateway: In case it's impossible to scrape metrics (using probes), metrics can be exposed to the Prometheus collector by this mechanism³⁶.

The manager needs a persistent connection with the output REST API, a connection oriented based technology will be used. **RabbitMQ** will be very convenient because of the following³⁷:

- Availability in many languages and platforms
- Asynchronous Messaging: Supports multiple messaging protocols, message queuing, delivery acknowledgement, flexible routing to queues, multiple exchange types. Those features allow for publish/subscribe communication and high-speed asynchronous I/O engines, in a tiny library.
- Distributed Deployment: Deploy as clusters for high availability and throughput; federate across multiple availability zones and regions.

Persistent data needs to be stored for later use. Since all RESTful APIs within the triple monitoring engine use JSON format, metrics don't always have the same structure. A convenient technology for saving this data will be using a database that handles JSON format to facilitate data transfer within the triple monitoring engine and to allow polymorphism. Based on the amount of data arriving per second and the huge quantity of operations that need to be performed, solutions like MongoDB will be very efficient [48].

As said before, the triple monitoring engine provides two REST interfaces.

- The first has the goal of receiving data from different sources and sending them to the Netdata collector (plugin). This interface will be the input of the monitoring engine. The API keeps data in memory until it is consumed by the plugin. Applications (data producers) will have access to this API for sending their measurements.
- The second interface provides the output of the monitoring engine to applications (consumers). This interface has two kinds of connection to serve results: a REST API and a Publish/Subscribe mechanism that is a connection-oriented service.

³⁵ Prometheus. <https://prometheus.io/>

³⁶ Prometheus <https://prometheus.io/>

³⁷ RabbitMQ <https://www.rabbitmq.com/>

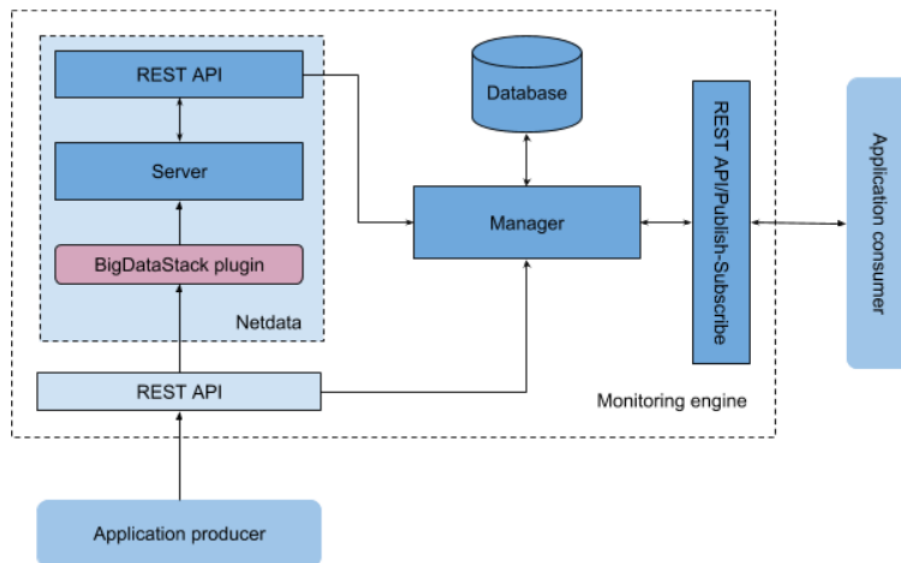


Figure 3. Netdata role in triple monitoring

Netdata is a system for health and performance monitoring of distributed real-time systems. It provides real-time insights of everything happening on the system it runs (including applications such as web and database servers), using interactive web dashboards [7]. Netdata’s main capabilities are gathering data from different sources and exposing them through a REST API. The Netdata architecture is extensible through plugins to read measurements (metrics) from different sources. In Figure 3 , the component named “BigDataStack plugin” is an adapter that needs to be deployed to ingest data into Netdata.

9.6 Applications & Data Services Deployment

Application and Data Services Deployment is concerned with how to deploy the user’s application onto the cloud infrastructure, as well as subsequent re-deployment in cases where the initial deployment did not meet the user’s quality of service requirements. It belongs within the realization engine of the overall BigDataStack platform. More precisely, it is comprised of two main components:

- I. **ADS-Ranking:** A component that ranks different possible deployment configurations of the user’s application on the cloud hardware (referred to as candidate deployment patterns) such that we can select the most suitable one given the user’s requirements. In a redeployment scenario, this component re-ranks and selects a new candidate deployment pattern that solves issues identified with the previous deployment.
- II. **ADS-Deployment:** A component that takes the best candidate deployment pattern and physically deploys it on the cloud infrastructure. Meanwhile, in a re-deployment scenario, this service facilitates the transition of one-or-more already deployed user services to a new configuration based on an updated candidate deployment pattern.

9.6.1 ADS-Ranking

What is ADS-Ranking? ADS-Ranking is a component that is concerned with how best to deploy the user’s application onto the cloud based on information about the application and cluster characteristics. From a practical perspective, its role is to identify which - of a range of potential deployment options - is the best for the current user, given their stated (hard) requirements and other desirable (soft) characteristics (e.g. low cost or high throughput).

ADS-Ranking is strongly coupled to the Application & Data Services Dimensioning (ADS-Dimensioning) component of BigDataStack that sits above it in the overall architecture stack. The main output of ADS-Dimensioning is a series of candidate deployment patterns (ways that the user's application might be deployed). It is these deployment patterns that ADS-Deploy takes as input, and subsequently selects the best one to deploy. In practice, this is accomplished using state-of-the-art supervised machine learning techniques to rank the candidate deployment patterns, which model the relationships between the features of each candidate deployment pattern and the user requirements/preferences. ADS-Ranking is also used for application re-deployment in cases where a previously selected deployment was deemed unsuitable, for example, because the provided quality of service was too low or the cost too high. In this case, ADS-Ranking updates the underlying model with evidence from the failing deployment and re-ranks the candidate deployment patterns with the aim of finding a new one that will provide superior performance.

Literature Review Machine Learning: Machine learning refers to the field of approaches that automatically learn solutions to problems using prior data [22]. Machine learning has become closely linked with information retrieval, as many tasks in information retrieval can be formulated in a manner that can be tackled by machine learning approaches, e.g. categorising documents [23] or learning how to rank documents [24]. Moreover, machine learned approaches have shown to be effective for many of these tasks [25]. Indeed, commercial Web search engines like Google and Bing use machine learned models to drive their search rankings. An important concept within machine learning is that of a *feature*. A feature is some property about the subject of the learning. For example, for information retrieval ranking problems, the features might be about the documents or user queries. An example of a query feature is query length, while a document feature might be its PageRank [26] score. For the purposes of ADS-Ranking that we are concerned with here, our features are derived from the 'predicted performance' information within each candidate deployment pattern, i.e. estimations about how effective a candidate deployment pattern is likely to be.

Literature Review Learning-to-Rank: Learning to rank (LTR) approaches use machine learning to tackle item ranking problems [25]. In an information retrieval setting, this typically involves ranking documents with respect to relevancy, although other ranking criteria are possible. The aim of learning to rank approaches is to improve a given item ranking with respect to some property. This is achieved by re-ranking an initial ranking such that items with the desired property are promoted into the top ranks.

In their simplest form, learning to rank techniques use initial item rankings for a set of topics, features about the individual items within those rankings, and relevance assessments about the individual items for each topic, to form a ranking model. This model can then be applied to unseen item rankings, re-ranking them to increase some desired ranking property. When building (or training) a model, an initial item ranking is created, referred to as the *sample*. A sample should have high recall in terms of items with the desired ranking property, e.g. for relevancy-based rankings, the sample should contain many relevant documents [27]. However, these documents do not need to appear within the top ranks; indeed, it is the aim of LTR to achieve this through re-ranking. Next, features about each of the items are extracted. An effective feature should aid in distinguishing the items that have the desired property, e.g. relevance to the query. In effect, the LTR approach aims to find a combination of these features that leads to effective ranking. Indeed, given the sample and its features, a

learning to rank approach will try different combinations of those features to find those that lead to increased effectiveness when ranking the sample. LTR approaches repeat this process for many document samples to find the feature combination that leads to improved effectiveness across all those samples. This feature combination is referred to as the ranking model. The idea is that the resultant ranking model will generalise to unseen sample rankings, if the training samples are representative of the types of rankings encountered. The ranking model can be updated with new samples over time, which is referred to as Active/Reinforcement Learning.

Learning to rank approaches can be categorised into three different types. Each type of approach uses a different strategy to evaluate the sample ranking. These types are point wise, pair wise and list wise. Point wise techniques learn on a per-item basis, i.e. each item is considered independently. Pair wise techniques optimise the number of pairs of items correctly ranked. List wise techniques optimise the entire ranking list at one time. Prior work has indicated that list wise techniques learn more effective Models [25].

Within ADS-Ranking we use learning-to-rank techniques to produce models to rank different candidate deployment patterns for the user.

Selected Technologies: For BigDataStack, the ADS-Ranking component will be built on top of the open source Apache Flink framework and will be written in Java. ADS-Ranking will be deployed as a real-time stream processing service using Flink, which assesses candidate deployment patterns for different user applications and emits a single selected pattern for each. In this way, the component will be scalable to high-demand periods, as well as extensible. The component will be operationalized as a series of transformers within the Flink service, divided into: transformers for converting candidate deployment patterns into features; the learning of ranking models based on those features; and the application of those models for ranking during service operation.

9.6.2 ADS-Deployment

Cloud application service orchestration frameworks manage all dependencies and relationships between components of the application, facilitating infrastructure-centric orchestration in the cloud. They provide a solution for the provisioning and management (from deployment to adaptation) of complex applications in the cloud. We use the concept of “service template” or “application chart” to refer to the specification of the service structure and “orchestration” to the management of the behaviour of IT infrastructure services where the application is deployed and operated.

OpenStack Heat³⁸ and **AWS CloudFormation**³⁹ are based on specific language descriptors. TOSCA⁴⁰ specification provides a language to describe application service components and their relationships using a service topology. This specification also provides mechanisms to describe management procedures that create or modify services using orchestration processes. Popular cloud tools such as OpenStack Heat conform to this specification.

³⁸ Openstack Heat. <https://wiki.openstack.org/wiki/Heat>,

³⁹ AWS CloudFormation. <https://aws.amazon.com/cloudformation/>,

⁴⁰ OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) , https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca

jClouds⁴¹ is a toolkit that facilitates interoperability with different cloud tools. It gives developers the freedom to create applications that are portable across clouds while giving you full control to use cloud-specific features. It is an open source multi-cloud library for the Java platform.

Docker⁴² is a technology for operating-system-level virtualization and is a popular container solution. Docker uses the resource isolation features of the Linux kernel to allow independent “containers” to run within a single Linux instance and therefore avoiding the overhead of managing multiple virtual machines (VMs). To achieve this, Docker lets developers package and deploy libraries and any other configuration and dependency at operating system level with the application code.

Container orchestration solutions are based on the concept of Docker containers. **Docker Swarm**⁴³ is the native container orchestrator of Docker. Swarm uses the same Docker interface, which enables transparent scalability from Docker use to Swarm use. Swarm manager is responsible for the management of a cluster of so-called Docker hosts over which it distributed the containers for execution.

Kubernetes⁴⁴ is the most popular container orchestration solution. Kubernetes scheduler runs as a process alongside the other master components such as the API server. A Kubernetes pod models an application-specific “logical host” and contains one or more application containers, which are relatively tightly coupled. Pods can also be exposed as Kubernetes services, which facilitates the management of containerized application services. Kubernetes charts offer a way to model, deploy and manage the life-cycle of complex containerized applications (i.e. consisting of several inter-related containers, pods and services). **Helm**⁴⁵ is an open source library implementing this capability. It offers an API to deploy applications on Kubernetes and manage their life-cycle. By means of Helm, developers can install, delete and upgrade whole applications/services with one command.

Other solutions for application management in the cloud are **Alien4Cloud**⁴⁶, supporting application SLA (service level agreement) specification, and **Serf**⁴⁷, offering a decentralized solution for cluster management, failure detection, and orchestration; **Cloudify**⁴⁸, **Chef**⁴⁹ and **Puppet**⁵⁰, on the other hand, are general purpose infrastructure configuration solutions which provide very flexible automated application lifecycle management.

9.7 Distributed Storage & Analytics

ACID database systems providing transactional semantics have been to the foreground for many decades. However, after the evolve of the cloud computing ecosystem, application components could scale in numerous nodes and serve highly intense workloads, thus

⁴¹ Apache jClouds. <https://jclouds.apache.org/>

⁴² Docker. <https://www.docker.com/>

⁴³ Docker Swarm. <https://docs.docker.com/engine/swarm/>

⁴⁴ Kubernetes. <http://kubernetes.io/>

⁴⁵ Helm. <https://helm.sh/>

⁴⁶ Alien4Cloud. <http://alien4cloud.github.io/>

⁴⁷ Serf. <https://www.serf.io/>

⁴⁸ Cloudify. <http://docs.getcloudify.org/3.3.1/intro/what-is-cloudify/>

⁴⁹ Chef. <https://www.chef.io/chef/>

⁵⁰ Puppet. <https://puppet.com/>

requiring from the persistent storage systems to be able to handle these intensive loads. Traditional database systems however rely on the two-phase-commit [28, 29] as the de facto atomic commit protocol. The latter inevitably introduces increased latency, which makes it slow, as it requires two rounds of messages between the coordinator and the participants of the transaction. Additionally, the existence of a central coordinator to manage the transaction lifecycle during the commit phase makes the protocol very difficult to scale adequately to be able to handle numerous nodes. This limited scalability of transactions offered by traditional database systems lead to the emergence of new data management technologies, frequently known as NoSQL, that trade the lack of support for ACID transactions for scalability, thus delegating consistency checks and transactional management on the application level.

To overcome these limitations, during the last decade, quite a few solutions that try to combine scalable transactional support without sacrificing consistency, such as Percolator [30] or Spanner [31] have been proposed. However, they continue to suffer from the limitations as most of them still rely on variations of the two-phase-commit protocol, which inherently introduces increased latency by design or having a centralized transactional manager like Omid [32] and Apache Tephra [33], which prevents them from being able to scale linearly when large deployments are needed. Finally, others make use of expensive hardware for requesting time events, like Spanner, Deuteronomy and LEAP.

Moreover, traditional database systems providing transactional capabilities, thus serving OLTP workloads ensuring ACID properties, rely on locking mechanism to provide the required isolation level. This means that heavy analytical queries will prevent write operations on the database until the formers finish, and vice versa, intensive operational workloads prevent analytical queries to be executed, as each of these types of queries block each other. To overcome this limitation, enterprises used to make use of ETLs to take a snapshot of the data, duplicate it in a data warehouse and perform the analytical operations on the latter. However, the past few years have witnessed a rise in demand for real-time Big Data Analytics for real-time business intelligence with a large range of research terms being adopted [34]. The goal is to develop tools that enable analytical processing over data that should be most up to date, thus processing data as soon as they arrive into the system [35]. Even if most systems claim to have real-time capabilities [36], they should be considered “near-real-time” as they still rely on an update process to acquire the latest data snapshot. As organizations increasingly require analytics on fresh operational data to derive timely insights, the notion of hybrid OLTP and OLAP databases have emerged, currently most known as HTAP (Hybrid Transaction and Analytical Processing) that does not involve the use of some kind of ETLs which are cost-expensive and introduce data duplication while they do not provide analytical processing over real-time data at the very end. **Hyper** [37] and **SAP HANA** [38] are typical HTAP database systems. The limitation of Hyper however is that it cannot be scaled horizontally as it must be deployed on a single machine, while SAP HANA suffers under intensive OLTP workloads and performs worse than a single-node typical database system.

To overcome all these challenges, **LeanXcale**⁵¹, which is a relational data base on top of a key-value store, is capable of handling write-intensive workloads, exploiting its ultra-scalable transactional management engine with its ability to linearly scale to 100s of nodes, while ensuring ACID properties and data coherence. Moreover, it provides tools for analytical

⁵¹ <https://www.leanxcale.com/>

processing as an integral part of the data store, which can exploit both inter- and intra-query parallelism, to provide real-time Big Data Analytics, thus truly implement the “just-in-time warehousing” model. LeanXcale internal OLAP engine is also distributed and each instance can be co-located with a corresponding data node, thus exploiting data affinity for improved performance. Moreover, its internal key-value storage engine can distribute its data load, by splitting, and merging data regions, whilst moving them across its data nodes on the runtime. During this process, the data store is fully operational, and no performance overhead is being noticed. Thus, the online re-distribution of the data load is performed seamlessly from the application point of view. Finally, for the application developers to use the distributed engine, an Elastic JDBC driver is provided, which implements all functionalities that can be found in traditional relational database management systems. Moreover, an additional driver can be used for directly accessing the internal key-value data store, which skips the overhead introduced by the Query Engine of the platform for improved performance.

9.8 Live Migration

Live migration is a pervasive technique in the realm of virtual machines, allowing transparent movement of virtual machine instances (VMs) from one physical machine to another with negligible service disruption (hence the term live). Live migration describes a mean to transfer a running VM from one physical host to another host without interrupting the VM execution and transparent to the VM’s users. The required information is transferred over the network (e.g., Ethernet) and includes an option to use an encrypted connection.

With recent trends toward scalability and distributed microservices, containerization has quickly become a lightweight and widely adopted alternative to VMs or physical nodes as the unit of deployment. The biggest benefit of containers is that they can be quickly created and destroyed in large numbers, and therefore relocating a container/pod in Kubernetes is only possible by disposing of the source pod, then recreating a new pod of the same type/template from scratch (see Section 8.1). Application developers need to design around this fact by not relying on longevity of pod-local state and by storing any necessary data into pod-independent persistent storage, such as an external database. Along with the fact that there are many scenarios that would benefit from the ability to relocate active pods, the work on developing live migration for containers has started, but it’s still in the early stages.

9.9 Data Quality Assessment

Data quality and verification is of major importance given that it affects the complete data path: storage, processing, analytics results, decision-making, etc. It poses many challenges in several phases of data management. In contrast to the much more researched modelling and analysis phases, the quality analysis and verification step is often seen as a sore point, even though without it the modelling and analysis phases could be of limited value. If the data are not verified and of acceptable quality (e.g. missing values, outliers etc.), the conclusions might be associated with a high level of uncertainty or even reduced to garbage.

In this project, we will develop mechanisms to evaluate the data quality, in terms of validity. These approaches will exploit artificial neural networks (ANN) and Deep Learning (DL) techniques, taking advantage of the work done on Natural Language Processing (NLP), to extract latent features that correlate different fields, and identify possible defects in the content.

Several methods to automate the error detection process have been proposed in the literature. Most of the research is focused on approaches that examine methodologies that look at the side effects as a proxy to errors. Outlier detection processes [57,8,11,13,21,27] investigate data points that deviate significantly with respect to their underlying distribution. De-duplication mechanisms identify clusters of records that refer to the same entities [58,7,9,13,18,25]. Rule-based methodologies consider constraints (e.g. denial constraints) that are explicitly set and look for tuples that violate them [59,1,5,10,17,26]. However, accurate these approaches maybe they can only capture errors corresponding to the specific side effect they are looking for.

To address this limitation, ensemble methods are used to capture different error types [60,2]. However, ensemble methods add a significant overhead to the complexity of the whole design, as they are very sensitive to how different detectors are combined [2]. They are easy to implement and straightforward to interpret; however, a lot of effort should be involved on how to tune the systems that unify the models, either sequentially or by using a simple voting scheme.

To this end, machine learning approaches are emerging to automatically address the heterogeneity of errors. The problem of error detection is now transformed into a binary classification question and an appropriate model is inquired as to whether a tuple is erroneous or correct. Moreover, this model should be able to implicitly encode the latent factors of heterogeneity and successfully detect any case of errors.

However, even though this level of automation is attractive, this approach needs many labelled examples, a responsibility that falls into the hands of the end user. **HoloDetect** [61,12] introduces a few-shot learning framework that builds an expressive model to learn rich representations of the nature of different error types and a data augmentation model to address the problem of data scarcity and relieve the user of the burden of manually defining labelled examples. **PIClean** [62,28], although not exactly an ML approach, adopts techniques that are extensively used in machine learning fields such as recommender systems, to automatically detect errors and suggest possible corrections. The error detection step computes a lower-ranked approximation \hat{X} of a given data set matrix X , and flags cells that have big differences (i.e. the bigger the difference between X_{ij} and X_{ij} the more likely X_{ij} is an error). To facilitate the implementation of such intricate deep learning algorithms, a suitable framework will be used. The prevailing option is that of Google's **Tensorflow**⁵². Several implementations like Yahoo!'s **TensorFlowOnSpark**⁵³ or **SparkNet**⁵⁴ allows this framework to run in a distributed way over a Spark cluster, thus, making it totally compatible with the BigDataStack's platform. Yahoo!'s implementation seems superior because it requires minimal changes in the original *TensorFlow* code.

9.10 Big Data Layout

Today's best practices to deploy and manage cloud compute and storage services independently leaves us with a problem: it means that potentially huge datasets need to be shipped from the storage service to the micro service to analyse data. If this data needs to be

⁵² <https://www.tensorflow.org/>

⁵³ <https://github.com/yahoo/TensorFlowOnSpark>

⁵⁴ <https://github.com/amplab/SparkNet>

sent across the WAN then this is even more critical. Therefore, it becomes of ultimate importance to minimize the amount of data sent across the network, since this is the key factor affecting cost and performance in this context. Many cloud-based **SQL** services, for example Amazon **Athena**⁵⁵, bill users according to the amount of data scanned in object storage, outlining the importance of this metric. There are currently three main approaches to limit the number of bytes sent from the storage to Spark. (Note we focus on object storage although this can also be applied more broadly).

The first is to use specialized column-based formats such as **Parquet**⁵⁶ and **ORC**⁵⁷. These formats provide column wise compression, which significantly reduces the number of bytes to be sent. They also support column pruning, so that only columns requested by a query need to be sent to Spark. Finally, they sometimes support specialized metadata which can be used to filter columns following query predicates. Parquet can provide more than an order magnitude performance improvement over other formats such as **CSV**⁵⁸.

The second approach is to use Hive style partitioning to name the objects using a certain convention. In this case, information about object contents is encoded into object names. For example, if we partition per a date column then each object will contain data records with the same date, and the date is encoded in the object name. **Spark SQL** understands this convention and can filter the set of objects retrieved when query predicates apply to partitioning columns. This can significantly reduce the number of objects sent to **Spark** and the number of bytes shipped (for more information see IBM's recent blog post⁵⁹).

The third approach is called Data Skipping and it can complement the first two approaches. This approach utilizes metadata to track information about objects and their dataset columns which can then be used for data skipping i.e. to show that an object is not relevant to a query and therefore does not need to be accessed from storage or sent on the network from object storage to Spark. IBM Research proposes the indexing of metadata, so that during query execution, can quickly filter out irrelevant objects from the list of objects to be accessed by the query. Note that this technique applies to all data formats, and avoids touching irrelevant objects altogether (see IBM's presentation⁶⁰ at the Spark Summit, where Databricks announced Data Skipping support in their platform).

To obtain good results in terms of Data Skipping, one typically needs to pay attention to Data Layout. Data layout refers to all details regarding the storage of the data including object size, format, Hive style partitioning, and data partitioning, i.e. the assignment of data records to objects. We focus now on data partitioning. For any given query, we would like the records which satisfy the query to be grouped together in a small set of objects, so that the remaining objects can be skipped. In general, we need to partition the data so that it gives as much as possible data skipping for an incoming stream of queries (i.e. a workload), not just a single

⁵⁵ <https://aws.amazon.com/athena/pricing/>

⁵⁶ <https://parquet.apache.org/>

⁵⁷ <https://orc.apache.org/>

⁵⁸ <http://blog.cloudera.com/blog/2016/04/benchmarking-apache-parquet-the-allstate-experience/>

⁵⁹ <https://www.ibm.com/blogs/bluemix/2018/06/big-data-layout/>

⁶⁰ <https://databricks.com/session/using-pluggable-apache-spark-sql-filters-to-help-gridpocket-users-keep-up-with-the-jones-and-save-the-planet>

query. Note that the various queries may have conflicting requirements. Moreover, the workload changes over time, as does the data.

This multi-dimensional partitioning and indexing problem has been addressed in the past with space-filling curves. Techniques based on Space-filling curves⁶¹ such as Z-order curves (or Morton curves⁶²) map a multi-dimensional space into a single indexing dimension represented by an encoding string (the metadata). These techniques can handle varying data density by issuing a geohash code of varying length. Possible usage of these techniques is to convert a given query bounding box into a one-dimensional code range and to use it against the indexed data. However, the main drawback of these techniques is the fact that the chosen space filling curve and the dataset points completely determines the partitioning. In addition, the query history is not considered. Also, one cannot dynamically change the way partitioning is done, and Space-filling curves treat all dimensions in a symmetric way so no way to “prefer” one dimension over the other (e.g., to achieve metadata compactness and to fit the representation to the query distribution). **QUILTS**⁶³ is an interesting novel paper on space-filling curves that was pointed to by the UPRC team. In the context of this project, we plan a collaboration between the IBM and UPRC teams for comparing the effectiveness of data partitioning based on of k-d-tree methods on the one hand and Space filling curves methods on the other hand.

Recently data partitioning for the specific purpose of data skipping has become an active research area, and there has been significant work to define the problem and various approaches to solve it. A fundamental paper shows that the general problem is NP-hard, but devises a heuristic algorithm which performs well in practical use cases [49, 50]. A follow-on paper shows how improve the layout further for handle column-based formats⁶⁴.

In 2017, as part of **IOStack**, IBM Research independently developed the notion of k-d tree partitioning which uses query history to choose the partitioning columns and dataset medians as cutting points for partitioning⁶⁵. In parallel, a paper was published by an MIT team which used a similar approach and similarly applied it to **Apache Spark** for data skipping [51]. The work in this paper went beyond previous work by providing an adaptive approach to repartition datasets on the fly according to a cost model. A recent companion paper covers how their technique can be applied to join processing [52].

This is a cutting-edge research area which is also promising in terms of its applicability to analytics on real world big datasets. We plan to undertake further research in this area as well as apply it to a commercial setting.

⁶¹ http://wwwmayr.informatik.tu-muenchen.de/konferenzen/Jass05/courses/2/Valgaerts/Valgaerts_paper.pdf

⁶² “Z-order curves.” https://en.wikipedia.org/wiki/Z-order_curve

⁶³ QUILTS: Multidimensional Partitioning Framework Based on Query-Aware and Skew-Tolerant Space-Filling Curves http://delivery.acm.org/10.1145/3040000/3035934/p1525-nishimura.pdf?ip=195.110.40.7&id=3035934&acc=ACTIVE%20SERVICE&key=90259622823F2C32%2E795DEE178C5AE221%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&acm=1571747931_daf1b6381c87c54a1a6f7dbb76d4602e

⁶⁴ Skipping-oriented partitioning for columnar layouts VLDB 2016 <https://dl.acm.org/citation.cfm?id=3025123>

⁶⁵ IOStack. <http://iostack.eu/deliverables/send/3-deliverables/31-d4-3-summary-and-demonstration-of-results>

9.11 Real-time CEP

Streaming engines are used for real-time analysis of data collected from heterogeneous data sources with very high rates. Given the amount of data to be process in real-time (from thousands to millions of events per second), scalability is a fundamental feature for data streaming technologies. In the last decade, several data streaming systems have been released. **StreamCloud** [39] was the first system addressing the scalability problem allowing a parallel distributed processing of massive amounts of collected data. **Apache Storm**⁶⁶ and later **Apache Flink**⁶⁷ followed the same path providing commercial solutions able to distribute and parallelize the data processing over several machines to increase the system throughput in terms of number of events processed per second. Apache Spark added streaming capability onto their product later⁶⁸. **Spark** approach is not purely streamed, if fact it divides the data stream into a set of micro batches and repeat the processing of these batches in loop.

The streaming engine for the BigDataStack platform is a scalable complex event processing (CEP) able to run in federated environments and to aggregate and correlate real-time events with structured information stored in LeanXcale.

BigDataStack CEP is built upon the streaming engine owned by UPM and extended to run in federated environments with different resources performing correlation on the edge closer to the data sources.

The metrics exported by UPM's CEP technology are the following:

- CPU load: percentage of CPU used by an Instance Manager (IM, i.e. CEP Worker). One value per IM.
- Subquery # tuples received: Number of tuples received by a sub-query. One value per sub-query instance.
- Operator # tuples received: Number of tuples received by a streaming operator. One value per operator instance.
- Operator # throughput: Number of tuples produced by a streaming operator. One per operator instance.
- Operator latency: Time taken by a streaming operator to process a tuple. One value per operator instance.

9.12 Predictive and Process Analytics

In the predictive and process analytics component of the project there are two main goals to be achieved: Predictive Analytics and Process Analytics. Following is a brief introduction to the tools and technologies which will be used for the above.

9.12.1 Predictive Analytics

In Predictive Analytics, the main goal is the selection of a correct algorithm from a set of available algorithms and model hyper-parameter tuning. To this end Predictive Analytics will utilize the resources of the Spark libraries.

⁶⁶ <http://storm.apache.org/>

⁶⁷ Apache Flink. <https://flink.apache.org/>

⁶⁸ Apache Spark. <https://spark.apache.org/streaming>

To begin, tools from *Spark SQL*⁶⁹, *Dataframes* and *Datasets Guides* will be used such as sampling a feature of *Hive*. When data volume is large, the need to find a subset of data to speed up data analysis becomes apparent. Here it comes to a technique used to select and analyse a subset of data to identify patterns and trends. In Hive, there are three ways of sampling data: random sampling, bucket table sampling, and block sampling.

Finally, tools from the *Spark MLlib* library will be used such as Cross-Validation, Train-Validation Split, and Approximate Nearest Neighbour Search etc. better described in the Spark documentation under: Model selection and tuning and Extracting, transforming and selecting features.

9.12.2 Process Analytics (Process Mining)

In Process Analytics, the main goal is to enhance, optimize process models derived from the process modelling framework and to discover process models from raw event log files. For the enhancement and optimization phase to take place an event log for the process itself will be required from the global tracker consisting of the steps taken for the application in each component of the architecture.

For these tasks a tool such as ProM (which is short for Process Mining framework) [40] will be a good candidate. ProM is an Open Source framework for process mining algorithms.

The ProM framework integrates the functionality of several existing process mining tools and provides many additional process-mining plug-ins. The ProM framework supports multiple formats and multiple languages, e.g., Petri nets, EPCs, Social Networks, etc. The plug-ins can be used in several ways and combined to be applied in real-life situations.

9.13 Seamless Analytics Framework

Typically, logical data sets of IoT data will become too big to be kept in a single “storage entity” (e.g., a database for *Cloudant*, or a bucket/container for *Object Storage*). Therefore, accessing a single logical data set may require targeting a continually changing and possibly large set of storage entities, thus rendering difficult the access to the data. To hide this complexity, the seamless storage driver analyses queries and maps them to exactly the storage entities that contain relevant data.

Figure shows an example where data is ingested with the *Watson IoT Platform* within multiple *Cloudant* databases with a daily bucketing interval. Using the seamless storage driver, which implements the data sources API⁷⁰ (a pluggable mechanism for accessing structured data through Spark SQL), a user can write simple and intuitive queries against the logical dataset without needing to refer to the various underlying databases. Moreover, the driver analyses the queries and accesses only the relevant databases are accessed.

⁶⁹ Spark SQL. <https://spark.apache.org/docs/2.2.0/sql-programming-guide.html#supported-hive-features>

⁷⁰ <https://databricks.com/blog/2015/01/09/spark-sql-data-sources-api-unified-data-access-for-the-spark-platform.html>

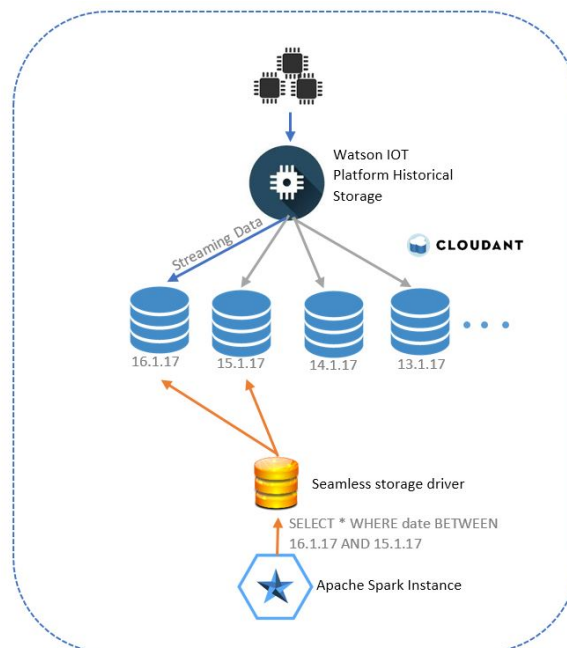


Figure 4. Data ingestion in Watson IoT Platform

Similarly, we want to investigate the more involved architecture where two completely different stores would be teamed together and present a seamless single access point. In the context of BigDataStack, we are speaking of the LeanXcale Data Base and an Object Store.

9.14 Application Dimensioning Workbench

For the Load injector case, we will utilize **Docker Swarm** as an easy mean to scale stress tests towards the target application/data service. For each case/category that needs to be tested, the respective tool from a wide set of available ones can be used to understand and implement a load, to cater for the different range of application types. Thus, baseline **Docker** images will be used, prepared in advance for the specific baseline tool case, in which the load file will be injected based on the test scenario. As an example, targeting at the project data stores, YCSB will be one candidate towards the LXS case while **Jmeter** jmx files may be used to target application level components such as web servers or for example the case of HTTP requests against the IBM Object store. One of the main benefits of this approach is that we separate the Load Injector framework from the baseline load generation tool, thus being able to reuse the former in different cases of the latter. Another benefit is the easy scale up of the Docker Swarm service approach to cater for extreme test conditions and avoidance of client-side bottleneck phenomena. However, further adaptation for other deployment specifications such as **OpenShift** or **AWS EC2** may be pursued in order to include multi-platform capabilities.

With relation to other baseline technology aspects, we anticipate inputs coming in and out of the dimensioning tool to be based on the Docker service manifest template (Docker Compose), to indicate service structure, component interconnection and naming. To extend the level of information in the file, we will also abide by the Docker specification (e.g. to include size of the resource per service element), but also inclusion of further metadata based on the custom metadata structure of the specification.

For the case of the dimensioning workbench front end, one candidate tool refers to **Node-RED** that can be used to easily create custom dashboards and integrate between different

services, while performing the various transformation and adaptation tasks needed (e.g. to understand the input file, start the testing process and generate the output file). This tool is expected to be used mainly for coordinating the various actions and presenting results to the user and not for the core modelling work which would be performed in the backend. For the case of the Modelling Engine backend, dimensioning models are anticipated to be based on artificial neural networks. To this end, candidate technologies include tools such as **GNU Octave**, an open source equivalent of **Matlab**, while other candidates include the **Tensorflow** library. Selection will be performed during the project. Potential integration of ML within **Node-RED** (e.g. through the usage of node-red-contrib-machine-learning node) will also be investigated. However, given that the latter primarily deal with classification aspects (and not regression ones as expected to be used by Dimensioning), this approach would need to be based on a concept such as QoS (Quality of Service) class categories.

9.15 Process modelling framework

For the case of implementing the Process Modeler component, initially **Node-Red** was used. Node-RED is open source software released under the Apache 2.0 license. Although initially targeted for Internet of Things (IoT) environments, it is highly extensible and has been successfully been employed in a wide range of applications.

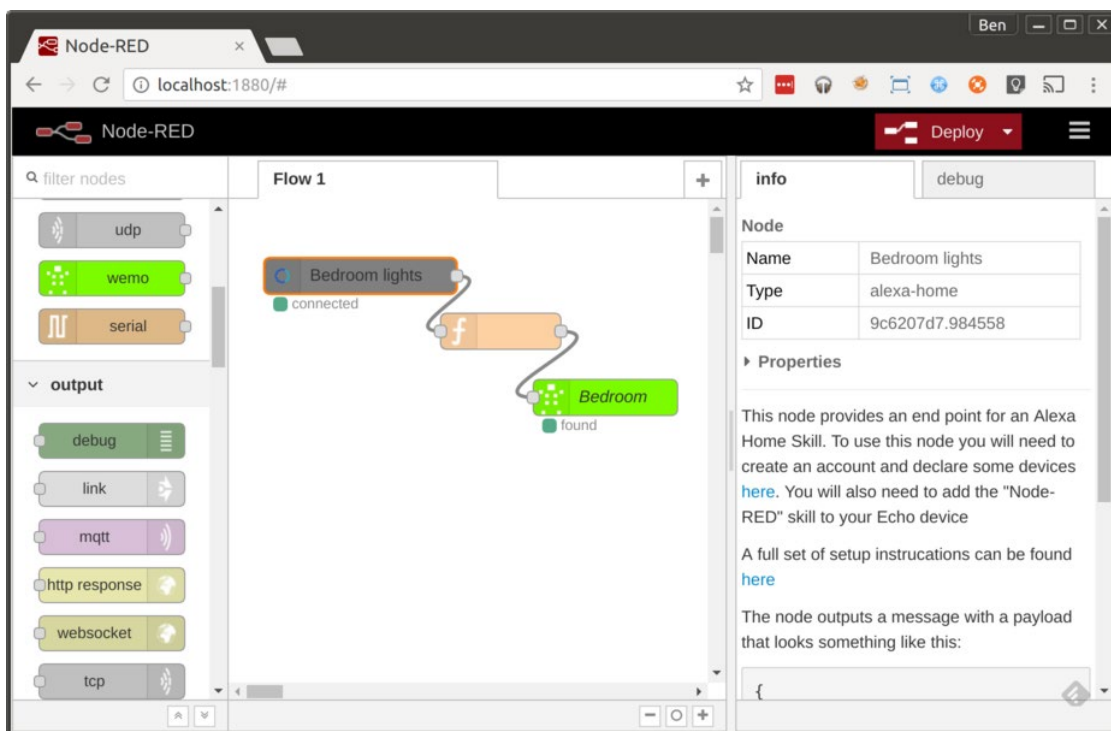


Figure 5. Node-red programming example

For the purposes of enhanced user interface and user experience, a transition to **Rete JS** over **Vue JavaScript** framework was necessary.

Rete is a modular framework for visual programming. Rete allows you to create node-based editor directly in the browser. You can define nodes and workers that allow users to create instructions for processing data in your editor without a single line of code.

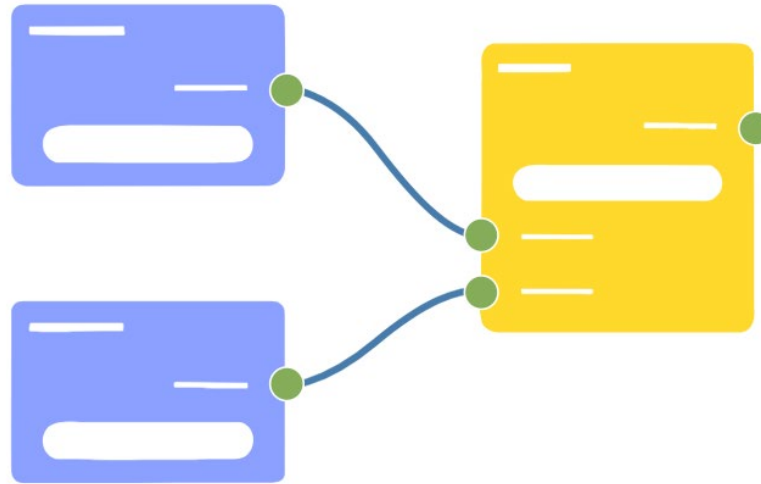


Figure 6. Rete-based editor for IoT applications modeling.

The Process Model Framework makes these and more building blocks in a suitable palette. The building blocks are classified in comprehensive categories. Users pick-by-click a building block from the palette into the editor and re-arrange its position by drag & drop. These building blocks which correspond to individual services can be linked together, so a higher-level graph is generated, and a business flow is defined.

It is to be expected that during the project the list of building blocks undergoes many changes. The Process Model Framework based on ReteJS is designed in such a way, so that it can easily and quickly adapt to the changes.

The Process Model Framework exports the designs/models into JSON files; this facilitates the integration with other BigDataStack tools that need to receive those as input.

9.16 Data Toolkit

The main objective of the Data Toolkit is to design and support diverse data analysis workflows in the support of streaming and batch data processing tasks. Such analysis workflows are designed according to the business requirements, the analysis experiments the user needs to execute and drive the process modelling which includes the abstracted definition of the analytic processing steps. The set of high level business process analysis steps already identified through the Process Modelling Framework, along with recommendations for the data analysis algorithms that are suitable to be used per step provided through the Process Mapping, need to be detailed in a scientific basis leading to the production of an end-to-end analysis workflow that can be realised over an application orchestration engine. Such an end-to-end analysis workflow is defined as the analysis playbook within the BigDataStack context.

Playbooks are service descriptions which consist of a set of data mining and analysis processes, interconnected among each other in terms of input/output data streams/batch processing objects. For instance, a single playbook is represented in the form of an analysis application graph (following concepts from cloud applications deployment and orchestration) that includes the set of individual processes as nodes of the graph along with their

binding/dependencies in the form of links representing the way these processes are interacting.

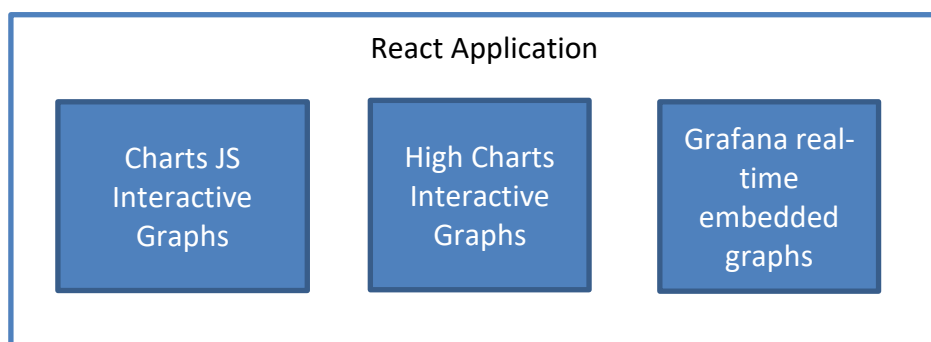
Playbooks should let data scientists design and develop complex analytic processes by combining a set of *already available* or *newly onboarded* analytic primitives. Each analytic primitive constitutes a functional component realized through a microservice and their collection can be combined within a palette of reusable components. Besides, each component may include characteristics related to input data parameters (type of data sources without any binding), output data parameters, analysis configuration parameters, execution substrate requirements and the software packages. Following, interconnection of such nodes (i.e. processes) leads to the production of the playbook (overall application graph).

A strict requirement regards the capacity to support various technologies/programming languages for development of analytics processes, given the existence and dominance of set of them (e.g., *R*, *Python*, *Java*, *Scala*). The developed analytics processes have also to be deployable over big data computing frameworks (e.g. **Apache Spark**, **Apache Flink**).

The data toolkit is currently implemented based on existing open source solutions along with their appropriate extension and customisation. Such solutions include -among others- tools like **Spring Cloud Data Flow**⁷¹ that provides tools to create complex topologies for streaming and batch data pipelines, **Conductor**⁷² that supports orchestration of micro-services-based process flows, **OpenCPU**⁷³ that is a system for embedded scientific computing and reproducible research. Currently, the development activities have been based on the Spring Cloud Data Flow due to its flexibility in configuration and parameterization of the available functional components and its high level of extendibility in the implementation of new components.

9.17 Adaptable Visualizations

The Adaptive Visualizations component is implemented as a web *Single Page Application* (SPA). **JavaScript** libraries are used for a fast loading, interactive and adaptable user interface (**React JS**) and for data visualization (**Charts JS** and **High Charts**). The following diagram depicts the main technologies to implement the application.



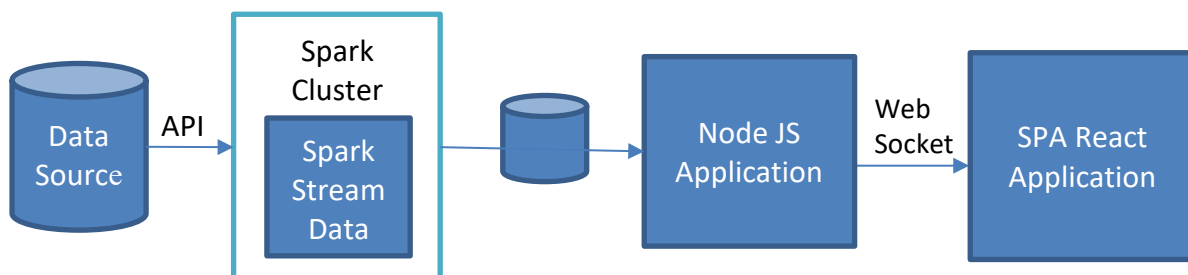
⁷¹ <https://spring.io/projects/spring-cloud-dataflow>

⁷² <https://netflix.github.io/conductor/>

⁷³ <https://www.opencpu.org/>

The application is implemented with React⁷⁴ *javascript* library. React is a modern JavaScript library that encourages good architectural design and follows a component-based approach. It makes it easy to design, debug and test fast interactive applications. For the graphs, Charts JS and **Highcharts**⁷⁵ libraries are used. Highcharts is a widely used, royalty-free commercial JavaScript library for creating impressive interactive web diagrams. It supports numerous diagram types that we expect to cover all BigDataStack needs. Should a more advanced or custom diagram be needed, the **D3.js**⁷⁶ library will be employed. For both Highcharts and D3.js ready-made components for React are available.

For certain use-cases visualization of real-time data is required. The following diagram depicts the components that will implement the real-time visualizations.



A Spark Cluster will be set-up (if necessary) and will read data from a data source through an appropriate API. The Spark Cluster will provide the Spark Streams that will process the data and produce the appropriate aggregations to be pushed in an intermediate database. A Node JS application will consume the aggregated data and will provide a Web Socket interface to the React Application. This will allow the real-time update of the visualization without the user having to refresh the page.

Furthermore, for time demanding tasks that need to be initiated from the SPA, message queueing and Web Socket technologies are employed. In detail, RabbitMQ message broker is set up and the responses received by the broker are propagated to the SPA from the NodeJS application using a *Socket.IO*.

All remaining datasets are acquired through RestFull API calls to external services and components.

⁷⁴ <https://reactjs.org/>

⁷⁵ <https://www.highcharts.com/>

⁷⁶ <https://d3js.org/>

10 Bibliography

- [1] G. Beskales, I. F. Ilyas, and L. Golab, “Sampling the repairs of functional dependency violations under hard constraints,” *Proc. VLDB Endow.*, vol. 3, no. 1–2, pp. 197–207, 2010.
- [2] W. Fan, J. Li, S. Ma, N. Tang, and W. Yu, “Towards certain fixes with editing rules and master data,” *Proc. VLDB Endow.*, vol. 3, no. 1–2, pp. 173–184, 2010.
- [3] J. Wang and N. Tang, “Towards dependable data repairing with fixing rules,” in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014, pp. 457–468.
- [4] X. Chu, I. F. Ilyas, and P. Papotti, “Holistic data cleaning: Putting violations into context,” in *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, 2013, pp. 458–469.
- [5] M. Heinsman, “Trifacta,” *Trifacta*. [Online]. Available at <https://www.trifacta.com/>. [Accessed: 23-May-2018].
- [6] M. Dallachiesa et al., “NADEEF: a commodity data cleaning system,” in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, 2013, pp. 541–552.
- [7] J. Wang, S. Krishnan, M. J. Franklin, K. Goldberg, T. Kraska, and T. Milo, “A sample-and-clean framework for fast and accurate query processing on dirty data,” in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014, pp. 469–480.
- [8] Z. Khayyat et al., “Bigdancing: A system for big data cleansing,” in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 2015, pp. 1215–1230.
- [9] Y. Altowim, D. V. Kalashnikov, and S. Mehrotra, “Progressive approach to relational entity resolution,” *Proc. VLDB Endow.*, vol. 7, no. 11, pp. 999–1010, 2014.
- [10] Z. Li, S. Shang, Q. Xie, and X. Zhang, “Cost reduction for web-based data imputation,” in *International Conference on Database Systems for Advanced Applications*, 2014, pp. 438–452.
- [11] D. Haas, J. Wang, E. Wu, and M. J. Franklin, “Clamshell: Speeding up crowds for low-latency data labeling,” *Proc. VLDB Endow.*, vol. 9, no. 4, pp. 372–383, 2015.
- [12] C. Gokhale et al., “Corleone: hands-off crowdsourcing for entity matching,” in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, 2014, pp. 601–612.
- [13] B. Mozafari, P. Sarkar, M. Franklin, M. Jordan, and S. Madden, “Scaling up crowd-sourcing to very large datasets: a case for active learning,” *Proc. VLDB Endow.*, vol. 8, no. 2, pp. 125–136, 2014.
- [14] X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang, “Data Cleaning: Overview and Emerging Challenges,” 2016, pp. 2201–2206.
- [15] P. Bohannon, W. Fan, M. Flaster, and R. Rastogi, “A cost-based model and effective heuristic for repairing constraints by value modification,” in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, 2005, pp. 143–154.
- [16] J. Wang, T. Kraska, M. J. Franklin, and J. Feng, “Crowder: Crowdsourcing entity resolution,” *Proc. VLDB Endow.*, vol. 5, no. 11, pp. 1483–1494, 2012.
- [17] A. Chalamalla, I. F. Ilyas, M. Ouzzani, and P. Papotti, “Descriptive and prescriptive data cleaning,” in *Proceedings of the 2014 ACM SIGMOD Int. Conf. on Management of data*, 2014, pp. 445–456.
- [18] L. Golab, H. Karloff, F. Korn, D. Srivastava, and B. Yu, “On generating near-optimal tableaux for conditional functional dependencies,” *Proc. VLDB Endow.*, vol. 1, no. 1, pp. 376–390, 2008.
- [19] G. Beskales, I. F. Ilyas, L. Golab, and A. Galiullin, “On the relative trust between inconsistent data and inaccurate constraints,” in *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, 2013, pp. 541–552.

- [20] M. Yakout, A. K. Elmagarmid, J. Neville, M. Ouzzani, and I. F. Ilyas, "Guided data repair," Proc. VLDB Endow., vol. 4, no. 5, pp. 279–289, 2011.
- [21] S. Krishnan, J. Wang, E. Wu, M. J. Franklin, and K. Goldberg, "Activeclean: Interactive data cleaning while learning convex loss models," ArXiv Prepr. ArXiv160103797, 2016.
- [22] Carbonell, J. (1990). Machine learning: paradigms and methods. Elsevier North-Holland, Inc.
- [23] Yu, H., Han, J. & Chang, K. C.-C., "PEBL: Positive example -based learning for Web page classification using SVM." In 'Proceedings of ACM SIGKDD 2002 International Conference on Knowledge Discovery and Data Mining'.
- [24] Agichtein, E., Brill, E. & Dumais, S. T., "Improving Web search ranking by incorporating user behavior information." In 'Proceedings of the 29th International ACM SIGIR Conference on Research and Development in Information Retrieval'.
- [25] Liu, T.-Y., "Learning to rank for information retrieval." Foundations Trends Information Retrieval. 3, 225–331.
- [26] Page, L., Brin, S., Motwani, R. & Winograd, T., "The PageRank Citation Ranking: Bringing Order to the Web." Technical report. Stanford InfoLab. 1999
- [27] Macdonald, C., Santos, R. & Ounis, "The whens and hows of learning to rank." Information Retrieval. 2012
- [28] J. N. Gray, "Notes on data base operating systems," Lecture Notes in Computer Science, vol. 60, pp. 393-481, 1978.
- [29] H. Sturgis and B. Lampson, "Crash recovery in a distributed data storage system," Computer Science Laboratory, Xerox, Palo Alto, 1976.
- [30] D. Peng and F. Dabek, "Large-scale incremental processing using distributed transactions and notifications," in Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI'10), 2010.
- [31] J. C. Corbett, J. Dean, M. Epstein, A. Fikes, C. Frost, J. J. Furman, S. Ghemawat, A. Gubarev, C. Heiser, P. Hochschild, W. Hsieh, S. Kanthak, E. Kogan, H. Li, A. Lloyd and S. Melnik, "Spanner: Google's globally-distributed database," in Proceedings of the 10th USENIX Conference on Operating Systems Design and Implementation (OSDI '12), 2012.
- [32] D. G. Ferro, F. Junqueira, I. Kelly, B. Reed and M. Yabandeh, "Omid: Lock-free transactional support for distributed data stores," in IEEE 30th International Conference on Data Engineering (ICDE), Chicago, 2014.
- [33] Apache, "Apache Tephra," [Online]. Available at <http://tephra.incubator.apache.org>. [Accessed May 2018].
- [34] Amr Osman, Mohamed El-Refaey, Ayman Elnaggar, Towards Real-Time Analytics in the Cloud, In Proceedings of IEEE SERVICES, 2013
- [35] Mike Barlow, Real-Time Big Data Analytics: Emerging Architecture, O'Reilly Media, Inc., 2013
- [36] T. Özsu, P. Valduriez. Principles of Distributed Database Systems. Springer, 2011
- [37] Alfons Kemper and Thomas Neumann. HyPer: A hybrid OLTP&OLAP main memory database system based on virtual memory snapshots. In Proceedings of ICDE, 2011
- [38] Franz Färber, Sang Kyun Cha, Jürgen Primsch, Christof Bornhövd, Stefan Sigg, and Wolfgang Lehner. SAP HANA database: data management for modern business applications. In Proceedings of SIGMOD, 2012.

- [39] V. Gulisano, R. Jiménez-Peris, M. Patiño-Martínez, C. Soriente, P. Valduriez (2012) StreamCloud: An Elastic and Scalable Data Streaming System. *IEEE Trans. Parallel Distrib. Syst.* 23(12): 2351-2365.
- [40] B. F. van Dongen, A. K. A. de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, and W. M. P. van der Aalst, "The ProM Framework: A New Era in Process Mining Tool Support," in *Applications and Theory of Petri Nets 2005*, vol. 3536, G. Ciardo and P. Darondeau, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 444–454.
- [41] International Organization for Standardization, "ISO/IEC/IEEE 29148:2011 – Systems and software engineering — Life cycle processes — Requirements engineering," ISO/IEC/IEEE, Nov. 2011.
- [42] Open Grid Forum, "Web Services Agreement Specification (WS-Agreement)," Oct. 10, 2011. <http://ogf.org/documents/GFD.192.pdf>
- [43] Open Grid Forum, "WS-Agreement Negotiation Version 1.0," Jan. 31, 2011. https://www.ogf.org/Public_Comment_Docs/Documents/2011-03/WS-Agreement-Negotiation+v1.0.pdf
- [44] P. Pietzuch, J. Ledlie, J. Shneidman, M. Roussopoulos, M. Welsh, and M. Seltzer, "Network-Aware Operator Placement for Stream-Processing Systems", 22nd International Conference on Data Engineering (ICDE '06), pp. 49–53, IEEE Computer Society, 2006.
- [45] V. Cardellini, V. Grassi, F. Lo Presti, and M. Nardelli, "Distributed QoS-aware Scheduling in Storm", 9th ACM International Conference on Distributed Event-Based Systems, pp. 344-347, ACM, 2015.
- [46] Y. Xing, S. Zdonik, and J.-H. Hwang, "Dynamic Load Distribution in the Borealis Stream Processor", 21st Int. Conf. on Data Engineering (ICDE '05), pp. 791–802, IEEE Computer Society, 2005.
- [47] M. Hirzel, R. Soule, S. Schneider, B. Gedik, and R. Grimm, "A Catalog of Stream Processing Optimizations", *ACM Computing Surveys*, vol. 46, Mar. 2014, pp 1–34.
- [48] MongoDB MongoDB and MySQL Compare. [Accessed: 27/05/2018] <https://www.mongodb.com/compare/mongodb-mysql>
- [49] L. Sun, M. J. Franklin, S. Krishnan, and R. S. Xin, "Fine-grained partitioning for aggressive data skipping," *SIGMOD*, 2014.
- [50] L. Sun, S. Krishnan, R. S. Xin, and M. J. Franklin, "A partitioning framework for aggressive data skipping," *VLDB*, 2014.
- [51] A. Shanbhag, A. Jindal, S. Madden, J. Quiane, and A. J. Elmore, "A robust partitioning scheme for ad-hoc query workloads," *SoCC*, 2017.
- [52] Y. Lu, A. Shanbhag, A. Jindal, and S. Madden, "Adaptodb: Adaptive partitioning for distributed joins," *VLDB*, 2017.
- [53] D. McPherson, "Managing Compute Resources with OpenShift/Kubernetes," August 2016. Red Hat. <https://blog.openshift.com/managing-compute-resources-openshiftkubernetes/> [Accessed June 2018].
- [54] Mao, H., Netravali, R., & Alizadeh, M. (2017, August). Neural adaptive video streaming with pensieve. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication* (pp. 197-210). ACM.
- [55] Jiang, J., Ananthanarayanan, G., Bodik, P., Sen, S., & Stoica, I. (2018, August). Chameleon: scalable adaptation of video analytics. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication* (pp. 253-266). ACM.

- [56] Rao, J., Bu, X., Xu, C. Z., Wang, L., & Yin, G. (2009, June). VCONF: a reinforcement learning approach to virtual machines auto-configuration. In Proceedings of the 6th international conference on Autonomic computing (pp. 137-146). ACM.
- [57] Tamraparni Dasu and Ji Meng Loh. 2012. Statistical distortion: Consequences of data cleaning. Proceedings of the VLDB Endowment5, 11(2012), 1674–1683.
- [58] Tamraparni Dasu, Theodore Johnson, Shanmugaelayout Muthukrishnan, and Vladislav Shkapenyuk. 2002. Mining database structure; or, how to build a data quality browser. In Proceedings of the 2002 ACM SIGMOD international conference on Management of data. ACM,240–251
- [59] Ziawasch Abedjan, Cuneyt G Akcora, Mourad Ouzzani, Paolo Papotti, and Michael Stonebraker. 2015. Temporal rules discovery for web data cleaning. Proceedings of the VLDB Endowment9, 4 (2015), 336–347.
- [60] Ziawasch Abedjan, Xu Chu, Dong Deng, Raul Castro Fernandez, Ihab F Ilyas, Mourad Ouzzani, Paolo Papotti, Michael Stonebraker, and NanTang. 2016. Detecting data errors: Where are we and what needs to be done? Proceedings of the VLDB Endowment 9, 12 (2016), 993–1004.
- [61] Alireza Heidari, Joshua McGrath, Ihab F Ilyas, and Theodoros Rekatsinas. 2019. HoloDetect: Few-Shot Learning for Error Detection. Proceedings of the 2019 International Conference on Management of Data (2019), 829–846.
- [62] Zhuoran Yu and Xu Chu. 2019. PIClean: A Probabilistic and Inter-active Data Cleaning System. In Proceedings of the 2019 International Conference on Management of Data. ACM, 2021–2024.