Project acronym:	**PrEstoCloud**
Project full name:	**Proactive Cloud Resources Management at the Edge for efficient Real-Time Big Data Processing**
Grant agreement number:	**732339**

D2.2 Requirements for the PrEstoCloud Platform

Deliverable Editor:	**Blaž Novak (JSI)**
Other contributors:	**Birgit Helbig (Software AG), Dimitris Apostolou (ICCS), Giannis Ledakis (UBITECH), Giannis Verginadis (ICCS), Guillaume Urvoy-Keller (CNRS), Iyad Alshabani (ActiveEON), Nenad Stojanovic (Nissatech), Panagiotis Gouvas (UBITECH), Quentin Jacquemart (CNRS), Vincent Kherbache (ActiveEON)**
Deliverable Reviewers:	**Giannis Ledakis (UBITECH)**
Deliverable due date:	**30/06/2017**
Submission date:	
Distribution level:	**Public**
Version:	**1.0**

This document is part of a research project funded by the Horizon 2020 Framework Programme of the European Union

Change Log

Version	Date	Amended by	Changes
0.1	07.06.2017	Blaž Novak	Initial revision
0.2	10.06.2017	Blaž Novak	ToC
0.3	16.06.2017	Blaž Novak	ToC, version 2
0.4	10.07.2017	Blaž Novak	User stories
0.5	13.07.2017	Blaž Novak	System interfaces
0.6	15.07.2017	Blaž Novak	Functional requirements, initial version
0.7	19.07.2017	Blaž Novak	Functional requirements, with descriptions and revisions
0.8	20.07.2017	Blaž Novak	Functional requirements, final
0.9	21.07.2017	Blaž Novak	First draft
0.92	26.07.2017	Birgit Helbig	General proof-reading, and DataProtection / Software AG-related changes
0.95	01.08.2017	Nenad Stojanovic	Editing updates regarding the priority of requirements (all partners)
0.96	04.08.2017	Blaz Novak	Merged review modifications (Yevgeniya, Dimitris, Yannis) and response to comments to draft 1
0.97	21.08.2017	Blaz Novak	Incorporated revisions to draft 2

Table of Contents

Table of Contents 3
List of Tables and Figures 4
1. Executive Summary 5
2. Introduction 6
2.1 Scope and context of the document 6
2.2 Structure of the document 6
3. Methodology 7
3.1 Functional requirements 7
3.2 Non-functional requirements 7
3.3 Requirements gathering step 9
3.4 Requirements analysis step 9
3.5 Known system interfaces 10
4. Collected user stories 11
4.1 Platform owner stories 11
4.2 Data protection officer stories 12
4.3 DevOps stories 13
4.4 Application developer stories 15
5. Functional requirements 17
5.1 General requirements 18
5.2 Configuration requirements 30
5.3 Monitoring requirements 39
5.4 Regulatory requirements 43
5.5 Runtime requirements 44
5.6 Relations 46
5.7 Requirement priorities overview 48
6. Non-functional requirements summary 49
7. System interfaces 51
8. Summary 56
9. References 57

List of Tables and Figures

Table 1. List of stakeholders ---------- 9
Table 2. Platform owner stories---------- 11
Table 3. Data protection officer stories ---------- 12
Table 4. DevOps stories---------- 14
Table 5. Application developer stories---------- 16
Table 6. General capabilities requirements ---------- 29
Table 7. Configuration requirements ---------- 38
Table 8. Monitoring requirements---------- 42
Table 9. Regulatory requirements ---------- 43
Table 10. Runtime requirements ---------- 45
Table 11. Relationships between functional requirements ---------- 47
Table 12. Summary of functional requirement priorities ---------- 48
Table 13. List of non-functional requirements ---------- 50
Table 14. System interfaces known so far---------- 55

Figure 1. Requirements gathering process---------- 7
Figure 2. ISO 25010 software product quality requirements classes.---------- 8
Figure 3. Relationship between user stories and functional requirements. ---------- 17

1. Executive Summary

This deliverable is the result of a PrEstoCloud project task T2.2, "High-level requirements analysis for the PrEstoCloud platform".

In the document, we present the results of the requirements gathering, analysis and documentation process, and a list of currently known system interfaces. The purpose of this document is to provide a starting point for the design of the project architecture, and a reference for the platform software development. The PrEstoCloud platform that will be designed and built will need to comply with the requirements that we have identified. Since the PrEstoCloud platform does not have one specific end-user, we have decided to use an iterative approach, where the requirements list can be modified in the future, when more use cases become apparent, or when new ideas evolve from the development process. Any significant changes to this list will be reported in relevant future deliverables.

According to the description of work, this deliverable collects the initial requirements for the PrEstoCloud platform in terms of functionality, interfaces and information flow by using well-established methodologies for requirements capturing and description. We have decided to move the information flow diagram to the deliverable D2.3, which covers the actual platform architecture and is as such a more appropriate fit.

We have used a well-known approach from the agile software development methodology community where the interface between the end-users and the developers is formalized in a set of user stories that concisely specify user's needs and the rationale behind them. Those user stories are then converted to a set of explicit functional requirements that can be directly used to guide and evaluate the software development process.

The majority of the deliverable consists of the lists of generated user stories, which we collected among the project partners due to the lack of existing external end-users, the functional requirements list, derived from those stories, the analysis of those requirements in terms of thematic groups, priorities and relationships between them, and a non-exhaustive list of the system interfaces that we were able to identify at this point. The list of the system interfaces will grow during the development of the platform, as it will be implemented as a set of interconnected micro-services communicating through well-defined interfaces.

Functional requirements presented in this deliverable are annotated with non-functional categories using the ISO 25010 standard taxonomy, and are linked to the business key performance indicators that were described in the deliverable D7.1.

The requirements collected in this deliverable will be used as the input to the architecture design process.

2. Introduction

In order to support the design of the conceptual platform architecture of the PrEstoCloud project, we have collected a set of high level requirements that describe the expectations and needs of various potential stakeholders involved with the development and the use of the PrEstoCloud platform.

This document presents an aggregated view of these requirements, and will be one of the inputs to the architecture deliverable, D2.3.

We have also collected an initial list of system interfaces, describing some of the interfaces between the platform components and the external world.

2.1 Scope and context of the document

This document is a result of task T2.2, "High-level requirements analysis for the PrEstoCloud platform", in WP2. The main goal of this task is to gather and systemize the requirements of the platform in terms of functionality and interfaces, using established methodologies for software requirement gathering.

Results of task T2.2 will be one of the inputs for task T2.3, "Conceptual Architecture", which deals with the design of the overall software architecture of the platform, which will be developed within the project.

One of the major inputs and dependencies to this document is the deliverable D7.1, "As-is and To-Be Scenarios" [1], which describes the current situation of project use cases, their desired improvements, and summarizes the use-case requirements with a set of business KPIs. We refer to business KPIs collected in D7.1 in the list of functional requirements that we provide in this deliverable, to enable traceability of requirements back to business needs.

2.2 Structure of the document

In the next chapter, we describe the purpose of requirements gathering and our approach to the process.

Chapter 4 lists a subset of the user stories that we collected as a base for functional requirement analysis.

In chapter 5 we list the requirements derived from user stories, grouped by category, their relationships and priorities.

In chapter 6, we summarize the non-functional requirements.

Chapter 7 lists the currently known interfaces between the PrEstoCloud platform and the external world.

Finally, we summarize the document in chapter 8.

3. Methodology

A software requirements specification ("SRS") document is a starting point for developing a software system. It describes the requirements imposed on the software system by the stakeholders, and the fitness of the developed solution can be measured by the extent to which it fulfils the specified requirements.

The software requirements specification is the result of well-defined steps, which comprise a subset of a bigger software development methodology. The choice of the appropriate methodology eases the collection of requirements and improves their quality.

In general, all requirements specification processes consist of three distinct phases: *requirements gathering* followed by *requirements analysis*, and finally *requirements specification*.

Requirement gathering is a process where the software developer interacts with stakeholders and notes their needs and wishes. Requirements are not checked at this stage.

Requirement analysis is the subsequent process of processing and analysing the collected information, to determine the completeness, redundancy, feasibility, conflicts, etc. of the set of the collected requirements. Chosen requirements need to be complete, consistent, clear and actionable in order to be useful for software architecture design.

Requirements specification is the process of documenting the resulting requirements using a well-defined notation.

The entire process is iterative in many software development methodologies, meaning that after an initial set of requirements has been collected, further feedback is collected from the stakeholders, etc.

Figure 1 contains a graphical representation of how requirements specification steps fit together in a generic iterative setting.

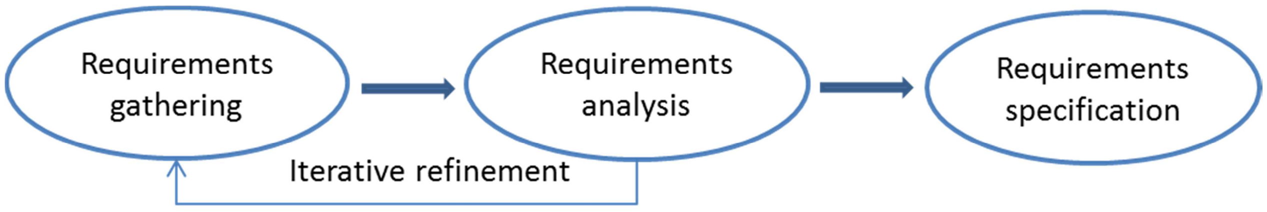

Figure 1. Requirements gathering process

Software requirements can cover various aspects of the system, such as business and market requirements, user interface requirements, functional requirements, non-functional requirements, etc.

3.1 Functional requirements

This document primarily deals with functional and non-functional requirements. Functional requirements describe the list of functionalities required from the system by the stakeholders. They describe *what* the system needs to be able to do, not *how* to implement this functionality, since, according to e.g. [2], the process of identifying, documenting and understanding the problem is separate from the process of solving it, and should as such be treated as a separate step in the development process.

3.2 Non-functional requirements

Non-functional requirements, sometimes called quality requirements, give the constraints on the functional requirements, specifying which kinds of qualitative characteristics are important for certain functions. We have chosen the ISO 25010 standard model of quality characteristics, as it seems to be appropriate and most commonly used. It defines a taxonomy of characteristics and sub-characteristics, as depicted in Figure 2 and described later in this document.

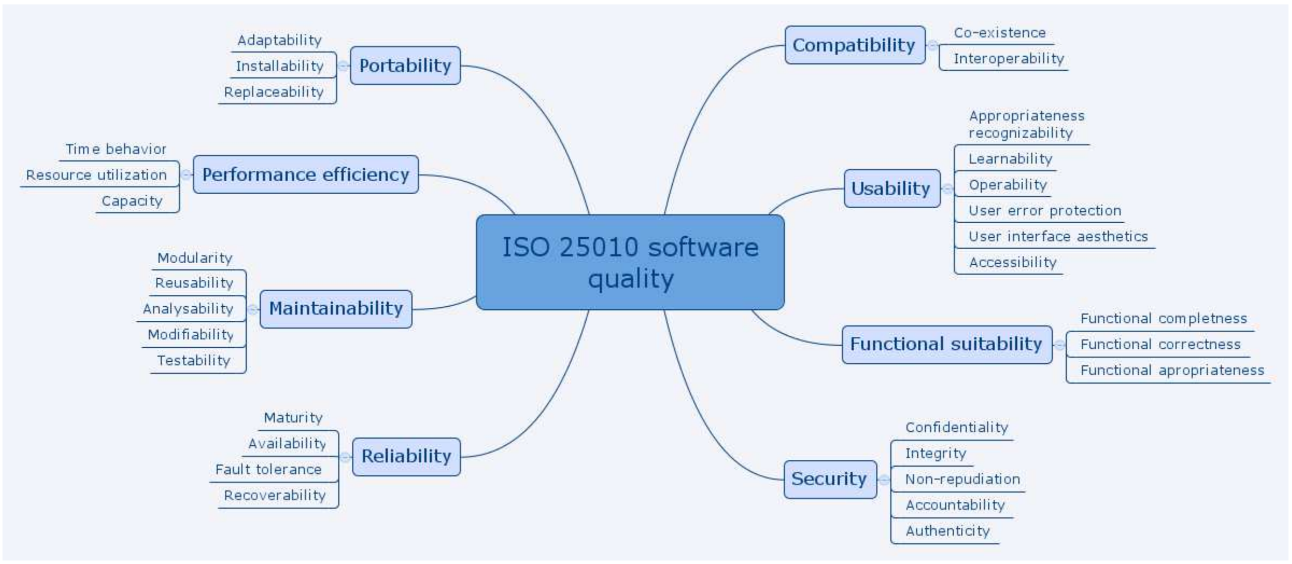

Figure 2. ISO 25010 software product quality requirements classes.

Documenting the collected requirements can be done using various different notations. A commonly used method is the use of "requirements lists" [2], where the requirements are gathered in a table and described with a short description and additional attributes. This is sufficiently expressive, but allows for some non-formality, which is a good match for our initial stage of the platform design.

The process of collecting the requirements begins with identifying the set of stakeholders involved. Since the goal of the PrEstoCloud project is to create a software architecture that will later be used by external users, which are as of yet unknown, we had to modify the usual approach where the real stakeholders are directly interviewed. We held a brainstorming session at a project meeting, where we came up with a list of 12 roles of actors that might interact with the platform.

The next step in the requirements elicitation process involves interaction with the stakeholders. Literature [3,4,5] suggests various approaches that can be used in different situations. Among them are:

- Brainstorming
- Document analysis
- Focus groups
- Interface analysis
- Interviews
- Prototyping
- Workshops
- etc…

A common approach used in agile software development methodologies is the 'user stories' collection format. A user story is a sentence in the form of "As a *[who]*, I want *[what]*, in order to *[why]*". These short "stories" allow the users to express requirements grounded in business needs, and are used to start a discussion on specific requirements. At the same time, requirements are simple to analyze and understand, since they come from single sentences. During the later discussion, the requirements can be confirmed or rejected before they are further analyzed.

3.3 Requirements gathering step

We have chosen a mixture of a workshop format and brainstorming format to collect user stories. During the meeting where we identified the actors, we have generated a small set of exemplary requirements. Afterwards, each project partner considered their planned technology in the context of identified stakeholders and business KPIs identified in D7.1, and produced a set of user stories, and a set of functional requirements based on those stories.

The stories were later grouped by the actor, and the requirements clustered by their topic – some covering configuration time topic, and some to run-time topics.

Every requirement was also annotated by a set of non-functional requirement characteristics described earlier. The characteristics are currently non-quantified, but as we progress through the implementation of the platform, we will decide on their appropriate values in context of project pilots.

We found that only four different types of actors were consistently mentioned in the user stories. These actors are described in Table 1. List of stakeholders

Role	Description
Platform owner	The owner or the manager of the software platform that is to be enhanced using PrEstoCloud technologies. Platform refers to the final product – the bigger cloud application that will be deployed through the PrEstoCloud platform.
Application developer	Software developer that is using the PrEstoCloud platform and integrating it with their system.
DevOps	Contraction of words "development" and "operations", and refers to people who cover the intersection of software development and the day-to-day operations of software deployment, configuration, maintenance, etc.
Data protection officer	Person responsible for regulatory compliance of the developed software solution.

Table 1. List of stakeholders

3.4 Requirements analysis step

During the analysis phase, we listed all the requirements by their topic, merged some requirements into others, and removed some that were either incomplete or irrelevant. We ended up with a list that is presented in chapter 5.

We compared the requirements, and identified which ones talk about the same subject, and the way that they are related. The possible options include one requirement being a superset or a subset of another, of two requirements having some amount of overlap, or the two requirements talking about the same subject in a related way. The result is presented in section 6 of chapter 5.

Since the list of requirements is long, we introduced the notion of the priority of the requirements in order to make the development as feasible as possible. We use the MoSCoW method[6] for defining requirement priorities.

The priorities are typically understood as:

Must have

Requirements labeled as "*Must have*" are critical to the current delivery timebox in order for it to be a success.

Should have

Requirements labeled as "*Should have*" are important but not necessary for delivery in the current delivery timebox. While "*Should have*" requirements can be as important as "*Must* *have"*, they are often not as time-critical or there may be another way to satisfy the requirement, so that it can be held back until a future delivery timebox.

Could have

Requirements labeled as "*Could have"* are desirable but not necessary, and could improve user experience or customer satisfaction for little development cost. These will typically be included if time and resources permit.

Won't have (this time)

Requirements labeled as "*Won't have"* have been agreed by stakeholders as the least-critical, lowest-payback items, or not appropriate at that time. As a result, "*Won't have"* requirements are not planned into the schedule for the next delivery timebox. "*Won't have"* requirements are either dropped or reconsidered for inclusion in a later timebox.

3.5 Known system interfaces

To complete the initial view of the platform that needs to be designed, we enumerated all the already known interfaces between the platform and the external world. The list of interfaces contains a description of the purpose of the interface, the intended user of the interface, the type, and technical parameters, such as frequency of interface access, timing and latency constraints, transfer rates, and security considerations. The list of interfaces is presented in chapter 7. The set of interfaces is not yet fixed, and even the delineation between internal and external interfaces is not yet strongly defined, as we intend to use a micro-services approach, where every interface is effectively external, but the list presented here provides a useful starting point for platform design nonetheless.

4. Collected user stories

As described in the methodology section, the first step in our process was to generate a set of user stories in the form of "As an X, I can Y so I can Z". Stories are organized into sections by the identified stakeholder. The purpose of these stories is to capture the business need of the user and communicate this need, and it's rationale to the platform architect. The last part of the sentence provides the backstory and eases the analysis of requirements. The following set of tables is the original set of those stories that were collected during brainstorming sessions by project partners and not rejected as irrelevant immediately. Functional requirements listed in the next chapter refer to these stories by their IDs. Content of multiple stories may overlap, in which case a single requirement was derived from multiple stories.

4.1 Platform owner stories

User story ID	**UID-1**
Story	As a platform owner I want to be able to register cloud infrastructure and edge resources regardless of their specifities, while providing metadata so they can be used on purpose to deploy and allocate appropriate resources.

User story ID	**UID-2**
Story	As a platform owner I want to monitor the status of all the resources from a single place and define monitoring rules, to be alerted in case of issues.

User story ID	**UID-3**
Story	As a platform owner I can run my application over multiple sites without having to modify the application itself so I can take advantage of higher computing power.

User story ID	**UID-4**
Story	As a platform owner, I can optimize the placement of my tasks so as to improve the utilization of already secured/purchased resources.

User story ID	**UID-5**
Story	As a platform owner I want to take advantage of both cloud and edge resources for the execution of compute intensive operations.

User story ID	**UID-6**
Story	As a platform owner I can register infrastructure resources in a platform independent way and their metadata (common PrEstoCloud model), so I can employ them during installation.

Table 2. Platform owner stories

4.2 Data protection officer stories

User story ID	**UID-7**
Story	As a Data protection officer, I can connect sites in a secure manner in order to ensure data privacy.

User story ID	**UID-8**
Story	As a Data protection officer, I can ensure that in case personal data is involved, collection, storing and analysis of this data is performed according to the General Data Protection Regulation (GDPR) coming into effect 2018, to ensure regulatory compliance and install trust with my end users.

User story ID	**UID-9**
Story	As a Data protection officer I want to be able to apply NFV functionalities in order to protect the compute resources and data.

User story ID	**UID-10**
Story	As a Data protection officer I want to be able to define network rules that can be changed when needed.

Table 3. Data protection officer stories

4.3 DevOps stories

User story ID	UID-11
Story	As a DevOps I want to implement custom scalability policies to adapt the reactivity of the platform depending of the load.

User story ID	UID-12
Story	As a DevOps I want to be able to express a network configuration in order to distribute my applications over multiple sites.

User story ID	UID-13
Story	As a DevOps I want to get information about the current status of the workload running on my infrastructure, so as to optimize the use of my infrastructure.

User story ID	UID-14
Story	As a DevOps I want to get information about the current status of my workload running on public resources, so as to minimize their cost.

User story ID	UID-16
Story	As a DevOps, I can monitor the network use of my application so as to refactor it.

User story ID	UID-17
Story	As a DevOps I want to be able to define constraints, so that the application can be automatically deployed on the cloud infrastructure.

User story ID	UID-18
Story	As a DevOps I want to be able to receive recommendations for initial application and data placement.

User story ID	UID-19
Story	As a DevOps I want to be able to accept recommendations and enact initial application and data placement.

User story ID	UID-20
Story	As a DevOps I want to be able to express constraints and preferences in order to reconfigure my big data intensive application (to maintain the QoS and QoE).

User story ID	UID-21
Story	As a DevOps I want to be able to receive recommendations for application

	placement reconfigurations and data migration based on detected situations that capture the current state of a placement topology (using cloud and edge resources).

User story ID	**UID-22**
Story	As a DevOps I want to be able to accept adaptation recommendations and enact adaptation actions.

User story ID	**UID-23**
Story	As a DevOps, I want to be able to detect interesting situations concerning the used Cloud and Edge resources.

User story ID	**UID-24**
Story	As a DevOps, I want to be able to infer Edge context from relevant event sources.

User story ID	**UID-25**
Story	As a DevOps I want to be able to connect relevant data source to the system in order to enable processing of that data in real-time.

User story ID	**UID-26**
Story	As a DevOps I want to be able to access relevant past data in order to do batch processing.

User story ID	**UID-27**
Story	As a DevOps I want to be able to provide real-time and batch processing in order to fulfill the requirements for data analytics.

User story ID	**UID-28**
Story	As a DevOps i would like to deploy a data intensive workflow (DISG).

User story ID	**UID-29**
Story	As a DevOps I want to create applications that can work using an ad-hoc mesh netwok.

User story ID	**UID-31**
Story	As a DevOps I would like to reduce the load of my cloud resources by engaging edge resources to reduce incoming data streams.

Table 4. DevOps stories

4.4 Application developer stories

User story ID	UID-32
Story	As an application developer I want to define custom selection scripts that combine multiple properties (i.e. based on metadata) to be able to map specific applications to desired set of resources.

User story ID	UID-33
Story	As an application developer I want to be able to design workflows containing custom tasks and logic to properly execute my application.

User story ID	UID-34
Story	As an application developer, I want to restrict which set of data needs to be processed locally in order to minimize cost due to data migration.

User story ID	UID-35
Story	As an application developer, I want to restrict which set of data needs to be processed locally in order to reduce delays due to data transfer or synchronization.

User story ID	UID-36
Story	As an application developer, I want to to be able to express constraints (affinity, anti-affinity, etc…) concerning the placement or workloads on resources.

User story ID	UID-37
Story	As an application developer I want to be able to exploit multi-clouds and edge resources for deploying DIA applications.

User story ID	UID-38
Story	As an application developer I want to be able to configure real-time and batch processing in order to fulfill the requirements for data analytics.

User story ID	UID-39
Story	As an application developer I want to be able to configure the output of real-time and batch processing in order to fulfill the requirements for decision making.

User story ID	UID-40
Story	As an application developer I want to be able to express requirements and preferences (both of which are called constraints) in order to automatically place my big data intensive application.

User story ID	**UID-41**
Story	As an application developer I want to annotate my workflow executors / my application code in order to drive the placement through the interpretation of the common presto model instance.

User story ID	**UID-42**
Story	As an application developer I can register an executable binary in a format which will accept annotations

Table 5. Application developer stories

5. Functional requirements

This chapter lists the actual functional requirements derived from the user stories. We started the process of platform requirement gathering by generating the user stories listed in the previous chapter, and then extracted functional requirements from those stories. There is not necessary a one to one mapping between user stories and functional requirements – in most cases, a single requirement was extracted from one story, but in some cases either multiple or none were extracted. The relationship between user stories and functional requirements is depicted on Figure 3.

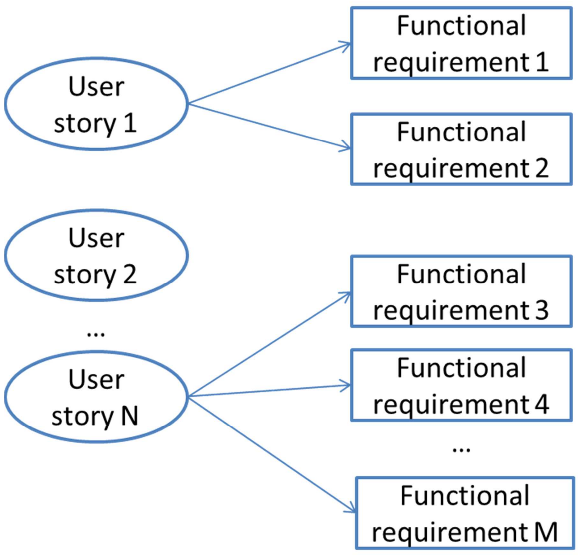

Figure 3. Relationship between user stories and functional requirements.

Every functional requirement is linked to the user story in the previous chapter by the "source story" attribute, and to related business KPIs from deliverable D7.1 by the "related BKPIs" attribute. The 'actor' attribute describes who is the main stakeholder that might have generated such a requirement, and the "non-functional requirements classes" lists an ISO25010 compatible list of qualitative sub-categories that specifies which qualities we will need to pay attention to during the evaluation of the architecture and the resulting platform when considering the requirement.

We grouped the functional requirements into the following categories:

- General capabilities
- Configuration requirements
- Monitoring requirements
- Regulatory requirements
- Runtime requirements

Configuration requirements group contains requirements that pertain to the configurability of the platform, monitoring section covers all of the requirements that deal with monitoring of the behavior and performance of the PrEstoCloud platform, while runtime requirements section similarly contains any requirements that pertain to the already deployed and configured PrEstoCloud platform. All requirements that deal with regulatory compliance are under the *Regulatory requirements* section, while all other are grouped in a non-specific *General requirements* section.

The requirements have been numbered in the same order as they were derived from the user stories. The numerical value of the requirement carries no semantics, so the requirements lists in the following sections aren't sorted by the requirement ID.

5.1 General requirements

ID	FR-6
Requirement	Platform offers a unified view of the sites available
Priority	Must Have
Actor	Platform owner
Source story	UID-3
Description	The platform owner is aware of all the available sites that can be used in order to deploy, upscale, or downscale the infrastructure running its application. The choice of the site is made independently of the list, as a preference (e.g. SLO).
Related business KPIs	Improving service deployment Improving service runtime & maintenance
Non-functional requirement classes	Portability.Adaptability

ID	FR-7
Requirement	Platform offers a unified view of the resources available
Priority	Must Have
Actor	Platform owner
Source story	UID-3
Description	The platform owner is aware of all the currently available resources (i.e. already paid-for), that can be used to further deploy workloads, or that could be freed (e.g. to reduce costs).
Related business KPIs	Improving service deployment Improving service runtime & maintenance
Non-functional requirement classes	Portability.Adaptability

ID	FR-10
Requirement	Uniform resource interaction (harmonized API for different cloud providers)
Priority	Should Have
Actor	Platform owner
Source story	UID-6
Description	The platform owner is able to utilize, manage or interact with different cloud providers by using a uniform interface that is based on an IaaS API harmonization layer.
Related	Simplifying human resources

business KPIs	Improving service runtime & maintenance
Non-functional requirement classes	Portability.Adaptability Maintainability.Reusability Modifiability.Testability Security.Authenticity

ID	**FR-2**
Requirement	Ability to set and attach metadata to resources based on a common description model
Priority	Should Have
Actor	Platform owner
Source story	UID-1
Description	Platform owner shall be able to provide metadata information on the resources he/she owns. This action shall be based on a common model and will be taken under consideration both during the initial placement and during the runtime configuration.
Related business KPIs	Simplifying human resources Improving service runtime & maintenance
Non-functional requirement classes	

ID	**FR-5**
Requirement	Anomaly detection and alerting
Priority	Must Have
Actor	Platform owner
Source story	UID-2
Description	Platform owner should be able to receive alerts about any anomalies during the execution of the stream processing.
Related business KPIs	Simplifying human resources Improving service health monitoring
Non-functional requirement classes	

ID	**FR-12**
Requirement	Platform enables the establishment of secure inter-site channels
Actor	Data protection officer
Source story	UID-7

Description	The platform establishes network connections between the different sites where the application executes. This communication channel offers confidentiality, authentication and integrity of data. In addition, this channel is made available during the whole deployment period.
Related business KPIs	nonspecific
Non-functional requirement classes	Reliability.Recoverability Security.Confidentiality Security.Integrity Security.Authencity

ID	**FR-14**
Requirement	Ability to apply network function virtualization
Priority	Must Have
Actor	Platform owner
Source story	UID-9
Description	Data protection officer shall be able to deploy VNFs so as to guarantee that a specific network functionality is achieved.
Related business KPIs	Nonspecific
Non-functional requirement classes	

ID	**FR-16**
Requirement	Ability to implement custom scalability policies
Priority	Must Have
Actor	DevOps
Source story	UID-11
Description	The DevOps should be able to define custom scalability rules for each infrastructure type (i.e. cloud infrastructure, docker on edge, etc.) by specializing scalability conditions. The main parameters would be: - how much resources to pre-allocate in order to handle load peaks. - how many pending jobs before deploying a new resource. - what are the conditions (i.e. amount of free resource, reduced load prediction, etc.) to release a resource.
Related business KPIs	On demand/dynamic resources management Improving service runtime & maintenance
Non-functional requirement classes	Reliability.Fault tolerance (Reliable recovery) Performance.Capacity (Multiple workflows in parallel) Security.Integrity

	Security.Confidentiality (Data Protection)

ID	**FR-22**
Requirement	Platform supports workload migration
Priority	Must Have
Actor	DevOps
Source story	UID-11
Description	The platform is able to either pause, move, and restart an ongoing workload, or it is able to re-instantiate the same workload (from a template) in another location, and then restart the workload, based on the serialization of the relevant application data.
Related business KPIs	Improving service runtime & maintenance
Non-functional requirement classes	Reliability.Recoverability Performance.Resource utilization

ID	**FR-27**
Requirement	Ability to receive recommendations on initial application placement
Priority	Must Have
Actor	DevOps
Source story	UID-18
Description	Based on annotations of the Data Intensive Application (DIA), the recommender will be able to propose which meaningful fragments (parts) of the DIA should to be deployed on different nodes of a real-time big data processing topology which includes both edge and cloud resources for resilience and performance reasons. Considering different properties like response time, security constraints or other quantitative or qualitative attributes, the recommender should be able to perform matchmaking against available cloud and edge resources and trigger their deployment..
Related business KPIs	On demand/dynamic resources management Improving service infrastructure
Non-functional requirement classes	Reliability.Availability Usability.Appropriateness recognizability (the DevOps should be able to recognize whether the recommendations are appropriate e.g with relative scoring between the recommendations)

ID	**FR-28**
Requirement	Ability to receive recommendations on initial data placement
Priority	Should Have

Actor	DevOps
Source story	UID-18
Description	The initial data placement constitutes a critical decision which often drives the application placement choices. Based on annotations of the Data Intensive Application (DIA), the recommender will be able to propose the network and/or physical location of data storage nodes. In addition, it should be able to identify cloud and edge resources that are in proximity of data sources in stream processing scenarios.
Related business KPIs	On demand/dynamic resources management Improving service infrastructure
Non-functional requirement classes	Reliability.Availability Usability.Appropriateness recognizability (the DevOps should be able to recognize whether the recommendations are appropriate e.g with relative scoring between the recommendations)

ID	**FR-33**
Requirement	Ability to receive recommendations on application placement reconfiguration
Priority	Must Have
Actor	DevOps
Source story	UID-21
Description	Based on the workload predictions, situation details, context of edge resources, variations of Big Data streams, the recommender should propose at the appropriate time the necessary adaptations of the infrastructure that hosts a DIA. These recommendations should be relayed to the Control and Cloud Infrastructure Layers of PrEstoCloud for enacting them. For example a new processing node may be recommended based on a predefined but dynamically changeable pool of alternatives resources. Such resources might include cloud resources or even other resources at the extreme edge of the network that their availability may be declared even at run-time.
Related business KPIs	Improving service infrastructure Improving service deployment
Non-functional requirement classes	Reliability.Availability Usability.Appropriateness recognizability (the DevOps should be able to recognize whether the recommendations are appropriate e.g with relative scoring between the recommendations)

ID	**FR-34**
Requirement	Ability to receive recommendations on data migration
Priority	Should Have
Actor	DevOps
Source story	UID-21
Description	Based on the workload predictions, situation details, context of edge resources,

	variation of Big Data streams the recommender should propose at the appropriate time the necessary migration of data to different nodes of the infrastructure that hosts a DIA. Moreover, the processing load of a single node or even the whole node that deals with a specific type of stream processing (e.g. complex event pattern detection) can be recommended to be offloaded to other more efficient nodes.
Related business KPIs	Improving service infrastructure Improving service deployment
Non-functional requirement classes	Reliability.Availability Usability.Appropriateness recognizability (the DevOps should be able to recognize whether the recommendations are appropriate e.g with relative scoring between the recommendations)

ID	**FR-39**
Requirement	Ability to detect interesting/critical situations that may lead to application reconfigurations or data migration
Priority	Must Have
Actor	DevOps
Source story	UID-23
Description	This requirement refers to the capability to detect / recognise interesting situations that might lead to resources adaptation recommendations or data-intensive application reconfiguration or redeployments. These situations may take into account the following inputs: (i) Big Data streams; (ii) complex event processing data; (iii) usage data; (iv) QoS variations (e.g., due to low bandwidth) and (v) Monitoring data related to the real-time processing networks.
Related business KPIs	Improving service runtime & maintenance Improving service deployment
Non-functional requirement classes	Functional suitability.Functional completeness Functional suitability.Functional correctness (important that extracted context/situations are accurate) Performance Efficiency.Time behavior

ID	**FR-40**
Requirement	Ability to extract high-level context for edge resources based on lower level monitoring data
Priority	Must Have
Actor	DevOps
Source story	UID-24
Description	This requirement refers to the acquisition and analysis of relevant contextual information derived from edge resources that are or will be engaged in either providing data streams or undertaking parts of the processing effort from the data-intensive applications. Analysis refers to the capability to process basic contextual data of edge devices such as their lat and long and infer the high-level status of devices such as that two devices are close to each other,

Related business KPIs	On demand/dynamic resource management
Non-functional requirement classes	Functional suitability.Functional completeness Functional suitability.Functional correctness (important that extracted context/situations are accurate) Performance Efficiency.Time behavior

ID	**FR-41**
Requirement	Ability to send / retrieve events to / from the Communications Broker (e.g. through an API)
Priority	Must Have
Actor	DevOps
Source story	UID-24
Description	This requirement refers to the availability of an API for handling inbound and outbound event messaging to and from the Communication Broker.
Related business KPIs	Nonspecific
Non-functional requirement classes	Reliability.Availability Functional suitability.Functional correctness (important that API calls should result in the correct results) Security.Confidentiality

ID	**FR-44**
Requirement	Enable an efficient and scalable access to past data
Priority	Must Have
Actor	DevOps
Source story	UID-26
Description	It is important to use the value of past experience by applying different data analytics methods on the past data (extraction of added value). However, the data has to be stored in a proper way in order to enable an efficient access (using data analytics methods). This requirement should ensure that the design of the data storage is appropriate to the sources of data that will be sensed (incl. the possible extensions). There should be a proper analysis of the relational and NoSQL databases.
Related business KPIs	nonspecific
Non-functional requirement classes	

ID	**FR-45**
Requirement	Ability to provide new methods for data processing (real-time, batch)
Priority	Must Have
Actor	DevOps
Source story	UID-27
Description	Extraction of the added value from past data is driven by specific data analytics methods. These methods have to be tuned to the characteristics of the data sources and the planned/intended usage of the past data analytics. The interfaces to the data should be clearly defined. New methods should enable past and real-time data processing.
Related business KPIs	nonspecific
Non-functional requirement classes	Usability.Accessibility

ID	**FR-51**
Requirement	Ability to write/design custom selection scripts and apply it to desired tasks
Priority	Must Have
Actor	Application developer
Source story	UID-32
Description	The application developer should be able to easily write and affect selection scripts on desired tasks (a 'job' or a 'workflow' is composed of multiples 'tasks') through a dedicated web portal. The developer needs to master a scripting language, e.g.: bash, groovy, Javascript, python, ruby, or perl. The selection scripts should rely on the metadata described in the common PrEstoCloud model (infrastructure type, qualitative metrics, etc.).
Related business KPIs	Improving service deployment
Non-functional requirement classes	Performance.Time behavior (minimum time between jobs) Security.Authenticity (Remote access) Reliability.Availability (External Service Available) Reliability.Availability (Internal Service Available)

ID	**FR-52**
Requirement	Retrieve desired set of metadata from common PrEstoCloud model
Priority	Should Have
Actor	Application developer
Source story	UID-32

Description	The application developer should be able to retrieve metadata (i.e. the classification of infrastructures, resources peculiarities, available metrics, etc.) in a uniform manner from the shared PrEstoCloud model (i.e. from a dedicated REST AP).
Related business KPIs	Nonspecific
Non-functional requirement classes	Performance.Time behavior (minimum time between jobs) Security.Authenticity (Remote access) Reliability.Availability (External Service Available) Reliability.Availability (Internal Service Available) Compatibility.Interoperability (runtime flexibility)

ID	**FR-55**
Requirement	Ability to write custom tasks based on common script language
Priority	Must Have
Actor	Application developer
Source story	UID-33
Description	The application developer should be able to populate workflows by writing custom tasks using its preferred scripting language (available: bash, python, perl, ruby, JS groovy, dockerFile). The tasks can be written directly from the WEB portal or imported from an XML file (one XML file per workflow). The tasks can be then customized (i.e. by specifying a selection script in order to be executed on the correct set of resources, or by adding controls logic to match specific conditions) before being executed.
Related business KPIs	Improving service deployment Improving service runtime & maintenance Simplifying human resources
Non-functional requirement classes	Reliability.Availability (External Service Available) Reliability.Availability (Internal Service Available) Compatibility.Interoperability (runtime flexibility) Portability.Adaptability (Dynamic configuration) Performance.Time behavior (Minimum deployment time)

ID	**FR-56**
Requirement	Platform understands the notion of locality
Priority	Must Have
Actor	Application developer
Source story	UID-34
Description	The platform is aware that some computing units need to be collocated, either on a hardware-level, or on a site-level (i.e. low-latency).

Related business KPIs	Improving service runtime & maintenance
Non-functional requirement classes	Reliability.Availability

ID	**FR-57**
Requirement	Platform estimates inter-site network cost
Priority	Must Have
Actor	Application developer
Source story	UID-34
Description	The platform is able to estimate the available bandwidth through network measurements techniques (passive and active), in order to estimate the additional operational cost of deploying a workload on multiple sites.
Related business KPIs	Improving service runtime & maintenance
Non-functional requirement classes	Reliability.Availability

ID	**FR-58**
Requirement	Platform estimates inter-site delay
Priority	Must Have
Actor	Application developer
Source story	UID-35
Description	The platform is able to estimate the available bandwidth through network measurement techniques (passive and active), in order to estimate the delay necessary for task synchronisation and/or data exchange for a workload deployed on multiple sites.
Related business KPIs	Improving service runtime & maintenance
Non-functional requirement classes	Reliability.Availability

ID	**FR-61**
Requirement	Ability to guide fragmentation of DIAs using annotations
Priority	Must Have
Actor	Application developer

Source story	UID-37
Description	Based on appropriate models, application developers should be able to perform annotations on DIAs in order to define their meaningful fragments that may be deployed in a distributed way on a hybrid cloud / edge infrastructure.
Related business KPIs	Improving service deployment
Non-functional requirement classes	Functional suitability.Functional completeness (provide a complete mapping between the criteria of the fragmentation and the existing annotations) Functional suitability.Functional correctness (the annotations should correctly map to the criteria of fragmentation

ID	**FR-62**
Requirement	Ability to guide the deployment of DIA fragments over cloud and edge resources using annotations
Priority	Must Have
Actor	Application developer
Source story	UID-37
Description	Based on appropriate models, application developers should be able to perform annotations on DIAs in order to guide how DIA fragments may be deployed in a distributed way on a hybrid cloud / edge infrastructure.
Related business KPIs	Improving service deployment
Non-functional requirement classes	Functional suitability.Functional completeness (provide a complete mapping between the criteria of the fragmentation and the existing annotations) Functional suitability.Functional correctness (the annotations should correctly map to the criteria of fragmentation

ID	**FR-63**
Requirement	Ability to accept recommendations about DIAs fragmentation and deployment
Priority	Must Have
Actor	Application developer
Source story	UID-37
Description	DIA developers should be able to receive intelligent recommendations helping them decide about meaningful annotations about fragmentation and deployment.
Related business KPIs	Improving service deployment
Non-functional requirement classes	Reliability.Availability

ID	**FR-69**
Requirement	Ability to containerize a Data Intensive Application and provide both the configuration layer and the scalability profile
Priority	Must Have
Actor	Application developer
Source story	UID-41
Description	Developer should be able to wrap any executable in a way that is comprehensive by the cloud deployment module. During the wrapping process the configuration aspects and the scalability aspects should be sufficiently covered.
Related business KPIs	Improving service deployment Improving service runtime & maintenance
Non-functional requirement classes	Portability.Adaptability Maintainability.Reusability

ID	**FR-70**
Requirement	Ability to wrap/upload and use a data intensive application in the PrestoCloud platform
Priority	Must Have
Actor	Application developer
Source story	UID-42
Description	A developer should be able to register a workflow consisting of data-intensive applications. All of these applications should comply with one formal model.
Related business KPIs	Improving service deployment Improving service runtime & maintenance
Non-functional requirement classes	Portability.Adaptability Maintainability.Reusability

Table 6. General capabilities requirements

5.2 Configuration requirements

ID	**FR-1**
Requirement	Ability to register cloud infrastructure by providing appropriate credentials and properties
Priority	Must Have
Actor	Platform owner
Source story	UID-1
Description	The platform owner should be able to manage the list of infrastructures available from a WEB portal or using a REST API. He should be able to add/register a new infrastructure or to remove obsolete ones. Among the provided informations must be present the credentials of the destination platform/infrastructure in order to allow dynamic deployment/release of resources.
Related business KPIs	Paid cloud resources Own cloud/premises resources Improving service deployment
Non-functional requirement classes	Portability.Adaptability (Dynamic configuration) Reliability.Availability (External Service Available) Reliability.Availability (Internal Service Available) Maintainability.Modifiability (Run-time adaptation)

ID	**FR-18**
Requirement	Ability to manually update the amount of resources
Priority	Must Have
Actor	Platform owner
Source story	UID-11
Description	The platform owner should be able to deploy new resource(s) from the infrastructure of choice. The new deployed resources can be manually configured to stay running until an explicit release action is performed for example or to inherit from a specific dynamic scaling policy.
Related business KPIs	Improving service runtime & maintenance Paid cloud resources On demand/dynamic resources management
Non-functional requirement classes	Portability.Adaptability (Dynamic configuration) Reliability.Availability (External Service Available) Reliability.Availability (Internal Service Available) Maintainability.Modifiability (Run-time adaptation) Compatibility.Interoperability (Heterogeneous Service Support)

ID	**FR-19**

Requirement	Platform allows to express the network configuration for all workloads
Priority	Must Have
Actor	DevOps
Source story	UID-12
Description	DevOps should be able to specify the wanted network configuration, in terms of IP addresses, so as to avoid possible address conflicts and to enable secure inter site connections..
Related business KPIs	Improving service deployment Improving service runtime & maintenance
Non-functional requirement classes	Compatibility.Interoperability Maintainability.Modularity

ID	**FR-24**
Requirement	Ability to define application deployment constraints
Priority	Must Have
Actor	DevOps
Source story	UID-17
Description	The DevOps should be supported in defining DIA deployment constraints which should be taken into account by the DIA deployment recommender.
Related business KPIs	Improving service runtime & maintenance
Non-functional requirement classes	Compatibility.Coexistence (with the constraints of other applications) Functional suitability.Functional completeness (ensure that at least all necessary constraints have been specified)

ID	**FR-25**
Requirement	Ability to define data placement/storing constraints
Priority	Must Have
Actor	DevOps
Source story	UID-17
Description	The DevOps should be supported in defining data placement / storing constraints which should be taken into account by the DIA deployment recommender.
Related business KPIs	Improving service runtime & maintenance
Non-functional requirement classes	Compatibility.Coexistence (with the constraints of other applications) Functional suitability.Functional completeness (ensure that at least all necessary constraints have been specified)

ID	FR-31
Requirement	Ability to express runtime scalability and qualitative constraints at the level of individual Data-Intensive Applications (DIAs)
Priority	Must Have
Actor	DevOps
Source story	UID-20
Description	The DevOps should be supported in defining scalability and qualitative constraints which should be taken into account by the DIA deployment recommender. For example, DIA A has high scalability requirements while it should be deployed only on resources meeting qualitative attribute B.
Related business KPIs	Improving service runtime & maintenance Improving service deployment
Non-functional requirement classes	Compatibility.Coexistence (with the constraints of other applications) Functional suitability.Functional completeness (ensure that at least all necessary constraints have been specified)

ID	FR-32
Requirement	Ability to express runtime data migration constraints
Priority	Should Have
Actor	DevOps
Source story	UID-20
Description	The DevOps should be supported in ruime migration constraints which should be taken into account by the DIA re-configuration recommender. For example, DIA A has data that should only be migrated from resource B to resource C if QoS is not affected during migration..
Related business KPIs	Improving service runtime & maintenance Improving service deployment
Non-functional requirement classes	Compatibility.Coexistence (with the constraints of other applications) Functional suitability.Functional completeness (ensure that at least all necessary constraints have been specified)

ID	FR-37

Requirement	Ability to use an editor for defining a situation triggering model
Priority	Must Have
Actor	DevOps
Source story	UID-23
Description	The DevOps should have at its disposal a graphical editor assisting the definition of situation models, based on the available topology monitoring data, event streams, possible workload predictions and other relevant entities of the hybrid cloud / edge infrastructure.
Related business KPIs	Simplifying human resources
Non-functional requirement classes	Functional suitability.Functional completeness (all intended situation categories should be included in the editor)

ID	**FR-42**
Requirement	Ability to register data sources (real-time) to the system
Actor	DevOps
Priority	Must Have
Source story	UID-25
Description	New data sources should be registered with the goal to enable feeding the system with new (real-time) data. Characteristics of the sources should be provided
Related business KPIs	On demand/dynamic resources management Improving service infrastructure Improving service runtime & maintenance
Non-functional requirement classes	

ID	**FR-46**
Requirement	Ability to combine registered resources and deployment and runtime constraints in order to deploy and manage the DISG
Priority	Could Have
Actor	DevOps
Source story	UID-28
Description	The platform should be able to use the registered resources in order to deploy a workflow of executors taking under consideration the constraints at the executor level and at the workflow level. These constraints should be 'translated' to a proper placement plan.

Related business KPIs	Improving service runtime & maintenance Improving service deployment
Non-functional requirement classes	

ID	**FR-48**
Requirement	Ability to express runtime constraints at the level of a) individual Data Intensive Applications b) entire Data Intensive Service Graph
Priority	Should Have
Actor	DevOps
Source story	UID-20
Description	Platform should offer a formal editor which will allow the creation of constraints both at the level of the workflow (Data Intensive Service Graph) and at the level of Data Intensive Applications (workflow executors)
Related business KPIs	Improving service runtime & maintenance
Non-functional requirement classes	

ID	**FR-49**
Requirement	Ability to group runtime constraints in the form of SLA objectives (Covering QoS/QoE constraints)
Priority	Could Have
Actor	DevOps
Source story	UID-20
Description	The formal editor that handles constraints should be able to express constraints related to the Quality Of Service and Quality Of Experience of the data intensive workflow
Related business KPIs	Improving service runtime & maintenance Improving service QoS performance Improving service QoE performance
Non-functional requirement classes	

ID	**FR-50**

Requirement	Ability to define infrastructural usage constraints which relate to VCPU, memory, network links, capacity and collocation
Priority	Could Have
Actor	DevOps
Source story	UID-31
Description	The formal editor that handles constraints should be able to constraint infrastructural parameters that relate to the deployment and the operation of the workflow. These parameters should correspond to the virtualization capabilities of the hypervisors.
Related business KPIs	Improving service infrastructure Improving service deployment
Non-functional requirement classes	

ID	**FR-53**
Requirement	Ability to build complex workflows from a WEB interface
Priority	Must Have
Actor	Application developer
Source story	UID-33
Description	The application developer should be able to manage workflows (creation/deletion) and to express the dependencies between the tasks of a workflow from a WEB Portal. The dependencies are mainly used to perform parallelization/sequentialization of the tasks execution. In addition to ‘dependencies’ management, the application developer should be able to express the controls logic of a workflow by adding custom conditions, loop, replicate, etc.
Related business KPIs	Simplifying human resources
Non-functional requirement classes	Portability.Adaptability (Dynamic configuration) Reliability.Availability (External Service Available) Reliability.Availability (Internal Service Available) Maintainability.Modifiability (Run-time adaptation) Reliability.Availability (Toolkit availability) Performance.Time behavior (Time to build the workflows) Security.Integrity (Web user space)

ID	**FR-54**

Requirement	Ability to manage (export/import) custom workflows
Priority	Should Have
Actor	Application developer
Source story	UID-33
Description	The application developer should be able import/export specific workflows from the WEB portal. The workflows are exported into XML format, results in a single XML data file per workflow.
Related business KPIs	Improving service deployment Improving service runtime & maintenance
Non-functional requirement classes	Portability.Adaptability (Dynamic configuration) Reliability.Availability (External Service Available) Reliability.Availability (Internal Service Available) Maintainability.Modifiability (Run-time adaptation) Reliability.Availability (Toolkit availability) Performance.Time behavior (Time to build the workflows) Security.Integrity (Web user space)

ID	**FR-59**
Requirement	Platform accepts placement constraints
Priority	Must Have
Actor	Application developer
Source story	UID-36
Description	The application developer can instruct the platform that some workloads need to be (or cannot be) executed on a specific type of hardware.
Related business KPIs	Improving service deployment
Non-functional requirement classes	Reliability.Recoverability Performance.Time Behavior

ID	**FR-64**
Requirement	Ability to adjust fragmentation and deployment
Priority	Should Have
Actor	Application developer
Source story	UID-37
Description	The application developer should have the means to accept, decline and adjust the automatic fragmentation and deployment recommendations.
Related	Improving service deployment

business KPIs	
Non-functional requirement classes	Functional suitability.Functional completeness (every deployment/fragmentation proposition must be accepted, or a valid alternative must be presented)

ID	**FR-65**
Requirement	Ability to configure methods for data processing (real-time, batch)
Priority	Should Have
Actor	Application developer
Source story	UID-38
Description	Big data processing methods usually work with very complex settings due to different nature of the input data. The system should enable an easy reconfiguration of the data processing methods, usually done in the interaction with users (GUI)
Related business KPIs	Nonspecific
Non-functional requirement classes	

ID	**FR-66**
Requirement	Ability to configure (e.g. context) outcomes of data processing (real-time, batch)
Priority	Could Have
Actor	Application developer
Source story	UID-39
Description	In order to use the outcomes of the big data processing properly, it is very important to put these outcomes in a suitable (business, process) context. This contextualization can be done in two different ways: automatically and semi-automatically (support by a user). This requirement should enable that the big data processing delivers information which can be effectively used in the relevant decision making processes
Related business KPIs	nonspecific
Non-functional requirement classes	

ID	**FR-68**

Requirement	Ability to express and compose a directed acyclic graph (hereinafter data intensive service graph - DISG) that consists of data-intensive-apps (hereinafter DIA) that collaborate each other
Priority	Should Have
Actor	Application developer
Source story	UID-40
Description	The platform should offer a workflow editor that is able to compose workflows of data intensive applications taking under consideration the chainability profile of each workflow executors.
Related business KPIs	Improving service deployment
Non-functional requirement classes	Functional suitability.Functional correctness

Table 7. Configuration requirements

5.3 Monitoring requirements

ID	**FR-3**
Requirement	Ability to monitor resources
Priority	Must Have
Actor	Platform owner
Source story	UID-2
Description	The platform owner should be able to monitor desired resources metrics and define logic to properly reacts on specific events. The corresponding monitoring rule must be submitted to a REST API. The monitoring itself is based on the CEP engine Drools, the metrics are exposed from Sigar into a JMX endpoint.
Related business KPIs	Improving service health monitoring
Non-functional requirement classes	Portability.Adaptability (Dynamic configuration) Reliability.Availability (External Service Available) Reliability.Availability (Internal Service Available) Maintainability.Modifiability (Run-time adaptation) Reliability.Availability (Toolkit availability) Security.Integrity (Integrity of collected metrics) Reliability.Availability (Message Queuing) Portability.Adaptability (Scalability -- Number of nodes managed) Performance.Time behavior (Time to build the workflows) Security.Integrity (Web user space)

ID	**FR-4**
Requirement	Ability to centralize the monitoring in a common view or place
Priority	Should Have
Actor	Platform owner
Source story	UID-2
Description	The platform owner should be able to consult and manage the list of monitoring rules submitted to the platform and decide to enable/disable or modify existing rules through a single and dedicated REST API.
Related business KPIs	Improving service health monitoring
Non-functional requirement classes	Reliability.Availability (Message Queuing) Security.Integrity (Integrity of Data) Security.Confidentiality (Remote Access) Reliability.Availability (Internal Service Available) Portability.Adaptability (Scalability -- Common storage space)

	Portability.Adaptability (Scalability -- Number of Messages) Performance.Time behavior (Time constraints)

ID	**FR-8**
Requirement	Platform reports global resource utilization
Priority	Must Have
Actor	Platform owner
Source story	UID-4
Description	The platform reports in real time (or close to real-time), the utilization in terms of CPU, memory, and network of each virtual resources (either virtual machines, or containers), with a per-site summary (e.g. utilisation of a given private/public cloud or cloudlet).
Related business KPIs	Improving service health monitoring Paid cloud resources
Non-functional requirement classes	Reliability.Availability Portability.Adaptability

ID	**FR-17**
Requirement	Ability to monitor the queue of workflows
Priority	Must Have
Actor	Platform owner
Source story	UID-11
Description	The platform owner should be able to monitor the queue of workflows in real time. He should be able to read the status of each workflow (including the status and the output of each sub-tasks) but also to perform appropriate actions (stop, suspend, resume specific tasks/workflows). This can be done from a WEB portal, a CLI or by directly consuming the dedicated REST API.
Related business KPIs	Improving service health monitoring
Non-functional requirement classes	Reliability.Availability (Message Queuing) Security.Integrity (Integrity of Data) Security.Confidentiality (Remote Access) Reliability.Availability (Internal Service Available) Portability.Adaptability (Scalability -- Number of Messages) Performance.Time behavior (Time constraints)

ID	**FR-20**
Requirement	Platform reports current and recent workload resource consumption
Priority	Must Have

Actor	DevOps
Source story	UID-13
Description	The platform provides historical data concerning the resource consumption of each workload in progress with a summary per site. This data can be used to optimize the usage or resources, e.g., upscaling or downscaling the infrastructure.
Related business KPIs	Improving service health monitoring Paid cloud resources Lowering direct costs of operation
Non-functional requirement classes	Reliability.Availability

ID	**FR-21**
Requirement	Platform reports per site costs
Priority	Could Have
Actor	DevOps
Source story	UID-14
Description	The platforms reports in real time or close to real time the cost of running the application for each site (public or private) based on an adequate cost model, either provided by the public cloud provider or constructed in an ad-hoc manner for the case of a private cloud.
Related business KPIs	Lowering direct costs of operation
Non-functional requirement classes	Reliability.Availability

ID	**FR-23**
Requirement	Platform reports network usage per machine, and per site
Priority	Must Have
Actor	DevOps
Source story	UID-16
Description	Each device running the distributed application provides information about how much data is exchanged. Similarly, each gateway point returns information about inter-site data transfers. Based on this information, a communication matrix can be maintained, and used as a basis for finding network bottleneck and optimizing application deployment, internally within a site, or across sites.
Related business KPIs	Improving service health monitoring
Non-functional	Reliability.Availability

requirement classes	Portability.Adaptability

Table 8. Monitoring requirements

5.4 Regulatory requirements

ID	**FR-13**
Requirement	The ability to comply with the new GDPR (General Data Protection Regulation) regarding the protection of personal data and personal sensitive data.
Actor	Data protection officer
Source story	UID-8
Description	The GDPR coming into effect in May 2018 clearly states how personal data and personal sensitive data is to be stored and/or processed. We have to ensure the ability of the platform to conform with these regulations also after the project runtime in order to support exploitation.
Related business KPIs	nonspecific
Non-functional requirement classes	Security.Confidentiality Security.Integrity Security.Accountability

ID	**FR-15**
Requirement	Ability to modify network rules to maintain regulatory compliance
Priority	Should Have
Actor	Data protection officer
Source story	UID-10
Description	Data protection officer shall be able to modify the security configuration of the cloud infrastructure in order to apply a security policy.
Related business KPIs	nonspecific
Non-functional requirement classes	

Table 9. Regulatory requirements

5.5 Runtime requirements

ID	**FR-26**
Requirement	Ability to enact (e.g. call an API) the execution of the initial deployment
Priority	Must Have
Actor	DevOps
Source story	UID-17
Description	The DevOps should be able to call a PrestoCloud API that will enact the initial DIA deployment generated by himself/herself with the aid of the initial deployment recommender.
Related business KPIs	nonspecific
Non-functional requirement classes	Reliability.Availability Functional suitability.Functional correctness (the called api correctly maps between request and action)

ID	**FR-30**
Requirement	Ability to enact (e.g. call an API) the implementation of the initial data placement
Priority	Should Have
Actor	DevOps
Source story	UID-19
Description	The DevOps should be able to call a PrestoCloud API that will implement the initial data placement generated by himself/herself with the aid of the initial deployment recommender.
Related business KPIs	nonspecific
Non-functional requirement classes	Reliability.Availability Functional suitability.Functional correctness (the called api correctly maps between request and action)

ID	**FR-35**
Requirement	Ability to enact (e.g. call an API) the implementation of the application placement reconfiguration
Priority	Must Have
Actor	DevOps
Source story	UID-22
Description	The DevOps should be able to call a PrestoCloud API that will implement the DIA placement re-configuration as recommended by the re-configuration recommender. For example this might involve horizontal or vertical scaling of the cloud infrastructure or migrating parts of the DIA from resource A to

	resource B.
Related business KPIs	nonspecific
Non-functional requirement classes	Reliability.Availability Functional suitability.Functional correctness(important that applications are correctly reconfigured and data is migrated correctly)

ID	**FR-36**
Requirement	Ability to enact (e.g. call an API) the implementation of data migration
Priority	Should Have
Actor	DevOps
Source story	UID-22
Description	The DevOps should be able to call a PrestoCloud API that will implement the data migration as recommended by the Resource Adaptation Recommender. That is, migrate data processing / storage of the DIA from resource A to resource B.
Related business KPIs	nonspecific
Non-functional requirement classes	Reliability.Availability Functional suitability.Functional correctness(important that applications are correctly reconfigured and data is migrated correctly)

ID	**FR-9**
Requirement	Ability to use cloud and edge compute resources
Priority	Must Have
Actor	Platform owner
Source story	UID-5
Description	
Related business KPIs	Exploiting resources at the extreme edge of the network Paid cloud resources Own cloud/premisses resources
Non-functional requirement classes	

Table 10. Runtime requirements

5.6 Relations

In this subsection we provide a preliminary analysis of the relations between functional requirements. The goal is to enable a better understanding of these relations and create a basis for an efficient development of the corresponding features.

We defined three types of relations:

- linked: meaning the two requirements are a different take on the same matter (e.g. decide and take action), or that the information (data) go together.
- overset/underset: means the LHS FRID (left part) is more general/less general than the RHS (right part) FRID. Eg for underset: a specific case of the general rule.
- overlap: requirements are the same, either for different actors, or for different components.
- partial overlap: requirements are partly the same, but, due to different actor or component, they are not 100% equivalent.

The identified relationships between the requirements are listed in the following table.

FRID_1	FRID_2	RELATIONSHIP	NOTE
FR-24	FR-59	overlap	both are about expressing deployment constraints
FR-26	FR-1	linked	need to register cloud infrastructure to enact
FR-26	FR-2	linked	
FR-27	FR-14	linked	
FR-31	FR-56	overset	runtime scalability possibly include constraints related to locality of data
FR-31	FR-57	linked	scalability constraints are possibly linked to inter-site networking costs
FR-31	FR-16	linked	policies need to be implemented (FR31 = Need to, FR16 = ability to)
FR-31	FR-18	linked	policies need to be enforced (FR31 = Need to, FR18 = dynamic enforcment)
FR-32	FR-57	linked	data migration contraints could be based on networking costs
FR-32	FR-18	linked	ability to dynamically enfore constraints
FR-33	FR-8	linked	need to know resource status to make (and receive) recommendations
FR-35	FR-22	partial overlap	workload migration could be a requisite (in the case of VMs) to enact reconfiguration
FR-35	FR-22	linked	
FR-38	FR-6	underset	topology data is underset of sites available
FR-38	FR-7	underset	topology data is underset of resources available
FR-38	FR-8	underset	topology data is udnerset of global resource status
FR-38	FR-20	linked	topology data can be based on current and recent past workload/status
FR-38	FR-21	linked	topology data contains inter-site networking costs
FR-38	FR-23	linked	topology data contains network usage statistics
FR-1	FR-6	overlap	cloud infrastructure registered <> unified view
FR-2	FR-7	linked	
FR-3	FR-8	linked	monitor resources <> global resource utilization
FR-3	FR-20	linked	monitor resources <> report current and recent workloads
FR-3	FR-23	linked	monitor resources <> report networking statistics
FR-4	FR-6	overlap	centralise monitoring <> unified view of sites
FR-4	FR-7	overlap	centralise monitoring <> unified view of resources

FR-16	FR-59	linked	need placement constraints to implement custom policies
FR-17	FR-20	overlap	
FR-51	FR-19	overset	FR-19 is a specific instance of FR-51
FR-17	FR-20	Overlap	monitor workload <> report workload
FR-48	FR-25	overset	
FR-48	FR-31	overset	
FR-48	FR-32	overset	
FR-48	FR-50	overset	

Table 11. Relationships between functional requirements

5.7 Requirement priorities overview

For the ease of referencing and overview, the following table lists all the requirements grouped by the identified priority.

Priority	List of requirements
Must have	FR-1, FR-10, FR-14, FR-16, FR-17, FR-18, FR-19, FR-20, FR-22, FR-23, FR-24, FR-25, FR-26, FR-27, FR-3, FR-31, FR-33, FR-35, FR-37, FR-39, FR-40, FR-41, FR-42, FR-44, FR-45, FR-5, FR-51, FR-53, FR-55, FR-56, FR-57, FR-58, FR-59, FR-6, FR-61, FR-62, FR-63, FR-69, FR-7, FR-70, FR-8, FR-9
Should have	FR-10, FR-15, FR-2, FR-28, FR-30, FR-32, FR-34, FR-36, FR-4, FR-48, FR-52, FR-54, FR-64, FR-65
Could have	FR-21, FR-46, FR-49, FR-50, FR-66
Won't have	none

Table 12. Summary of functional requirement priorities

6. Non-functional requirements summary

The non-functional requirement categories that were mentioned in the functional requirement list are listed here, along with a description. Non-functional requirements are, at this point, unquantified – we do not specify values for given parameters (e.g. how many requests per second a service will need to be able to handle), but list what attributes will need to be taken into account while designing a piece of functionality.

ISO 25010 non-functional requirement name	Requirement description	Related functional requirements
Compatibility.Coexistence	The ability of the PrEstoCloud platform to run along other pieces of software without interfering with them in a negative way, or being impacted negatively itself	FR-24, FR-25, FR-31, FR-32
Compatibility.Interoperability	The ability of the PrEstoCloud platform to communicate and interact with other software components	FR-52, FR-55, FR-18, FR-19
Functional suitability.Functional completeness	The degree to which the PrEstoCloud platform functionality has been specified in full, to cover all the required tasks	FR-39, FR-40, FR-61, FR-62, FR-24, FR-25, FR-31, FR-32, FR-37, FR-64
Functional suitability.Functional correctness	The degree to which the PrEstoCloud platform functionality performs with correct behavior and results	FR-39, FR-40, FR-41, FR-61, FR-62, FR-68, FR-26, FR-30, FR-35, FR-36
Maintainability.Modifiability	The degree to which the PrEstoCloud platform functionality can be parametrized, configured or modified while still performing correctly	FR-1, FR-18, FR-53, FR-54, FR-3
Maintainability.Modularity	The degree to which the PrEstoCloud platform components can be modified without significantly impacting the design or behavior of other platform components	FR-19
Maintainability.Reusability	The degree to which the PrEstoCloud platform components can be deployed in different scenarios	FR-10, FR-69, FR-70
Modifiability.Testability	The degree to which a PrEstoCloud platform component can be tested to behave correctly	FR-10
Performance.Capacity	The degree of the PrEstoCloud platform and component scalability	FR-16
Performance.Resource utilization	The degree to which the PrEstoCloud platform component uses system resources efficiently	FR-22
Performance.Time behavior	The latency and throughput of PrEstoCloud platform components	FR-39, FR-40, FR-51, FR-52, FR-55, FR-53, FR-54, FR-59, FR-3, FR-4, FR-17
Portability.Adaptability	The extent of the PrEstoCloud platform component's ability to run on different underlying platforms and systems	FR-6, FR-7, FR-10, FR-55, FR-69, FR-70, FR-1, FR-18, FR-53, FR-54, FR-3, FR-4, FR-8, FR-17, FR-23
Reliability.Availability	The degree to which the PrEstoCloud	FR-27, FR-28, FR-33, FR-

	platform and its components are available at the required times	34, FR-41, FR-51, FR-52, FR-55, FR-56, FR-57, FR-58, FR-63, FR-1, FR-18, FR-53, FR-54, FR-3, FR-4, FR-8, FR-17, FR-20, FR-21, FR-23, FR-26, FR-30, FR-35, FR-36
Reliability.Fault tolerance	The degree to which the PrEstoCloud platform can handle errors and faults while still performing correctly	FR-16
Reliability.Recoverability	The ability of the PrEstoCloud platform to recover it's state and data after a fault	FR-12, FR-22, FR-59
Security.Accountability	The ability to trace the actions of actors using the PrEstoCloud platform	FR-13
Security.Authenticity	The ability to correctly identify the entity issuing a request to the PrEstoCloud platform	FR-10, FR-12, FR-51, FR-52
Security.Confidentiality	The degree to which the PrEstoCloud platform insures the privacy of the data	FR-12, FR-16, FR-41, FR-4, FR-17, FR-13
Security.Integrity	The degree to which the PrEstoCloud platform prevents unauthorized access to data	FR-12, FR-16, FR-53, FR-54, FR-3, FR-4, FR-17, FR-13
Usability.Accessibility	The degree to which the PrEstoCloud platform can be used by various types of end-users easily	FR-45
Usability.Appropriateness recognizability	The degree to which various types of end-users can recognize that the PrEstoCloud platform meets their business needs	FR-27, FR-28, FR-33, FR-34

Table 13. List of non-functional requirements

7. System interfaces

The PrEstoCloud architecture will comprise of multiple components collaborating together to accomplish the required task. Some of these component's behaviours are already sufficiently understood by the project partners, so we have collected a list of known interfaces that are either exposed or required by these components. This list is preliminary and not exhaustive. It enumerates some of the ways the platform components will communicate either between themselves or with the outside world.

For each of the interfaces, we give multiple properties.

The *interface consumer* attribute identifies an external actor that is interacting with the interface for the purpose stated in the "*purpose of the interface*" attribute.

The "*type of interface*" can be, for example, a GUI, a REST API, a Java API, a Web Portal, etc.

Technical properties of the interface are given under "frequency of access", "timing requirements", "transfer rates" and "security considerations" fields.

Timing requirements cover the information on when and how the interface will be used, e.g. if it is asynchronous, and if it is called during deployment, development, etc.

Frequency of access gives information on how often we expect this interface will be called.

Transfer rates attribute provides an estimate on the amount of data transferred via the interface.

Security considerations field covers security related issues regarding this interface, specifically the need to use a secure transfer protocol.

The following table lists all the system interfaces we have already identified. Not all fields are known or relevant for all the interfaces, in which cases the values are left empty.

The ProActive Cloud Watch, Cloud Automation and Scheduler that are mentioned in the list are existing software solutions of ActiveEon.

ID	**IF-1**
Interface consumer	DevOps
Purpose of the interface	Specify the multicloud location of the different VMs, so as to obtain a proper IP nubering for the VMs
Type of the interface	API (to be defined, most likely REST)
Frequency of access	N/A
Timing requirements	At original deployment, or at scale time/redeployment
Transfer rates	1 request/deployment
Security requirements	None

ID	**IF-2**
Interface consumer	DevOps
Purpose of the interface	Specify the constraints of execution for sevices/microservices
Type of the interface	API (Java)
Frequency of access	Solver runs every 5 minutes usually

Timing requirements	N/A
Transfer rates	N/A
Security requirements	none

ID	**IF-3**
Interface consumer	Application Developer
Purpose of the interface	The application developer should be able to interact with the PrEstoCloud platform through a graphical user interface. The purpose of this interaction is to allow the application developer to describe and annotate a big data intensive application with all
Type of the interface	GUI
Frequency of access	Everytime a new application needs to be developed, refactored and deployed.
Timing requirements	Asynchronous
Transfer rates	N/A
Security requirements	Secure transfer protocol should be used.

ID	**IF-4**
Interface consumer	Edge devices
Purpose of the interface	Edge devices should be able to communicate with the PrEstoCloud platform through a dedicated communication broker. The purpose of this interaction is to enable PrEstoCloud to aggregate monitoring data that reveal the health status of all the considered edge devices.
Type of the interface	Pub/Sub (any message queuing protocol)
Frequency of access	Depends on the edge device and the metric of interest but it can be as low as in miliseconds.
Timing requirements	Asynchronous
Transfer rates	This refers to regular push-based messages in the size of a few Kbytes
Security requirements	Secure transfer protocol should be used.

ID	**IF-5**
Interface consumer	Edge devices
Purpose of the interface	The PrEstoCloud platform should be able to interact with edge devices through a dedicated communication broker. The purpose of this interaction is to enable PrEstoCloud to instruct edge devices to offload or onload processing jobs. In PrEstoCloud this int
Type of the interface	Pub/Sub (any message queuing protocol)
Frequency of access	Everytime a new application needs to be deployed or reconfigured.

Timing requirements	Asynchronous
Transfer rates	This refers to infrequent push-based messages in the size of a few Kbytes
Security requirements	Secure transfer protocol should be used

ID	**IF-6**
Interface consumer	Cloud resources
Purpose of the interface	Private and Public cloud resources should be able to interact with the PrEstoCloud platform through a dedicated communication broker. The purpose of this interaction is to enable PrEstoCloud to aggregate monitoring data that reveal the health status of al
Type of the interface	Pub/Sub (any message queuing protocol)
Frequency of access	Depends on the platform or application metric used but it can be as low as in miliseconds.
Timing requirements	Asynchronous
Transfer rates	This refers to regular push-based messages in the size of a few Kbytes
Security requirements	Secure transfer protocol should be used

ID	**IF-7**
Interface consumer	Application Developer
Purpose of the interface	The developer uses the JAVA interface of the ubi:chord library in order to create a mesh network topology among the edge devices
Type of the interface	Java API
Frequency of access	N/A
Timing requirements	During development
Transfer rates	N/A
Security requirements	N/A

ID	**IF-8**
Interface consumer	DevOps
Purpose of the interface	For Security Enforcement mechanism DevOp will be responsible to setup the required rules
Type of the interface	Web Portal
Frequency of access	N/A
Timing requirements	During setup or at DevOp will
Transfer rates	N/A
Security requirements	Usage of Secure Transfer Protocol would be beneficial

ID	**IF-9**
Interface consumer	Cloud Resources
Purpose of the interface	Security Enforcement mechanism communicates with the VMs (Cloud Resources) that are responsible to provide NFV rules and functonalities
Type of the interface	REST API
Frequency of access	N/A
Timing requirements	During setup or at DevOp will
Transfer rates	Non applicable
Security requirements	Usage of Secure Transfer Protocol would be beneficial

ID	**IF-10**
Interface consumer	DevOps who wants to use multi-IAAS connector, or monitor the system.
Purpose of the interface	ProActive Could Watch can monitor a system, and detect complex events and then trigger actions according to some rules (i.e. more than one probe are involved).
Type of the interface	REST API
Frequency of access	User defined configuration parameter
Timing requirements	As soon as possible(Asynchronous)
Transfer rates	N/A
Security requirements	None

ID	**IF-11**
Interface consumer	Interface which has need for multiple scheduling, service management etc.
Purpose of the interface	ProActive Cloud Automation (a cloud automation solution for complex multi-VM applications) is a platform able to deal with cloud services management in an autonomic and smart way. It aimes to tackle manageability and self-manageability requirements.
Type of the interface	Web Portal
Frequency of access	User defined
Timing requirements	N/A
Transfer rates	N/A
Security requirements	None

ID	IF-12
Interface consumer	Deployment workflows
Purpose of the interface	Using Proactive Scheduler to deploy tasks on the requested resources
Type of the interface	Web Portal/Command Line
Frequency of access	One time per job
Timing requirements	N/A
Transfer rates	N/A
Security requirements	None

Table 14. System interfaces known so far

8. Summary

Software requirements specification is an important step towards the design of a good software architecture. With full involvement of all project partners, we have collected an extensive set of requirements and interfaces, using a simplified version of an agile software development process, and categorized them into groups, which cover run-time and configuration time aspects. We have identified how requirements fit together, which ones are absolutely necessary, and which one should be considered optional.

All the functional requirements are annotated with the stakeholder, non-functional quality requirement types that will need to be taken into consideration, and are linked to both user stories and business key performance indicators, derived from project use cases. This will allow us to track and evaluate the progress of the development.

We also collected a list of currently known interfaces between the PrEstoCloud platform and the external world.

This document presents the first iteration of the platform requirements. We expect that the requirements set will evolve with the development of the first prototype of the platform, when we will revisit and potentially update them.

9. References

[1] PrEstoCloud Deliverable D7.1, "As-is and To-Be Scenarios", http://prestocloud-project.eu/index.php/deliverables/

[2]"Use Cases – Requirements in Context", Kulak, D., Guiney, E. Addison Wesley, 1st ed., 2000

[3] "Requirements Engineering: A Good Practice Guide", Sommerville, I., and Sawyer, P. (1997), Wiley & Sons, 1997

[4] "Writing Better Requirements", Alexander, I., and Stevens, R. Addison Wesley, 2002

[5] "Software Requirements: Styles and Techniques", Lauesen, S. Addison Wesley, 2002

[6] https://en.wikipedia.org/wiki/MoSCoW_method