



Project Title AN OPEN, TRUSTED FOG COMPUTING PLATFORM
FACILITATING THE DEPLOYMENT, ORCHESTRATION AND
MANAGEMENT OF SCALABLE, HETEROGENEOUS AND SECURE
IOT SERVICES AND CROSS-CLOUD APPS

Project Acronym RAINBOW

Grant Agreement No 871403

Instrument Research and Innovation action

Call / Topic H2020-ICT-2019-2020 /
Cloud Computing

Start Date of Project 01/01/2020

Duration of Project 36 months

D1.1 – RAINBOW Stakeholders Requirements Analysis

Work Package	WP1 – Requirements, Reference Architecture and Use-Cases
Lead Author (Org)	George Kakamoukas (K3Y)
Contributing Author(s) (Org)	Demetris Trihinas (UCY) Zacharias Georgiou (UCY) Moysis Symeonides (UCY) George Pallis (UCY) Marios D. Dikaiakos (UCY) Stefanos Venios (SUITE5) Thomas Pusztai (TUW) Schahram Dustdar (TUW) Vasileios Psomiadis (AUTH) Theodoros Toliopoulos (AUTH) Claudio Casetti (POLITO) Fabiana Chiasserini (POLITO) John Kaldis (AI3) Thanassis Giannetsos (DTU) Panos Gkouvas (UBI) Kostas Theodosiou (UBI) Christina Stratigaki (UBI)
Due Date	30.06.2020
Actual Date of Submission	30.06.2020
Version	V1.0



Dissemination Level

- | | |
|--|--|
| <input checked="checked" type="checkbox"/> | PU: Public (*on-line platform) |
| <input type="checkbox"/> | PP: Restricted to other programme participants (including the Commission) |
| <input type="checkbox"/> | RE: Restricted to a group specified by the consortium (including the Commission) |
| <input type="checkbox"/> | CO: Confidential, only for members of the consortium (including the Commission) |



Versioning and contribution history

Version	Date	Author	Notes
0.1	25.03.2020	K3Y	Initial version
0.2	17.04.2020	SUITE 5	Updated version. Contribution to Section 2
0.3	30.04.2020	UCY, TUW, AUTH, DTU	Updated version. Contribution Section 3
0.4	28.05.2020	K3Y	Updated version. Contribution Section 4 and 5
0.5	18.06.2020	UCY, TUW, AUTH, DTU	Updated version. Contribution Section 6
0.6	19.06.2020	K3Y	Updated version. Contribution Section 7
1.0	29.06.2020	K3Y	Reviewed after comments received from internal reviewers

Disclaimer

This document contains material and information that is proprietary and confidential to the RAINBOW Consortium and may not be copied, reproduced or modified in whole or in part for any purpose without the prior written consent of the RAINBOW Consortium

Despite the material and information contained in this document is considered to be precise and accurate, neither the Project Coordinator, nor any partner of the RAINBOW Consortium nor any individual acting on behalf of any of the partners of the RAINBOW Consortium make any warranty or representation whatsoever, express or implied, with respect to the use of the material, information, method or process disclosed in this document, including merchantability and fitness for a particular purpose or that such use does not infringe or interfere with privately owned rights.

In addition, neither the Project Coordinator, nor any partner of the RAINBOW Consortium nor any individual acting on behalf of any of the partners of the RAINBOW Consortium shall be liable for any direct, indirect or consequential loss, damage, claim or expense arising out of or in connection with any information, material, advice, inaccuracy or omission contained in this document.



Table of Contents

Executive Summary	10
1 Introduction	11
1.1 The RAINBOW Project	11
1.1.1 RAINBOW's Value Propositions	11
1.1.2 RAINBOW Use Cases	11
1.2 Structure of the Deliverable	13
2 Requirements Methodologies	14
2.1 Requirements Elicitation Framework	14
2.2 Techniques for Requirements Elicitation	14
2.2.1 Brainstorming	15
2.2.2 Document analysis	15
2.2.3 Focus Groups	16
2.2.4 Interface analysis	16
2.2.5 Interviews	17
2.2.6 Observation	17
2.2.7 Prototyping	17
2.2.8 Requirements Workshops	18
2.2.9 Survey/Questionnaire	18
2.3 RAINBOW Requirements Collection Methodology	19
3 Technology Analysis and State of the Art	21
3.1 Underlying technologies	21
3.1.1 Fog/Edge Computing	21
3.1.2 Microservices architectural paradigm	22
3.1.3 Service Graph Topology Descriptions	24
3.1.4 Secure service mesh networking (and routing)	25
3.1.5 Scalable trust establishment and attestation	26
3.1.6 Geo-distributed data processing	31
3.2 State of the Art paradigms	32
3.2.1 From literature	32
3.2.2 From Projects	33
3.2.3 From Industry	37
4 Stakeholders and their Goals	39
4.1 The 3 Categories of Stakeholders	39
4.1.1 Interests & Relations between Stakeholder Categories	40
4.1.2 Interest vs Power Matrix	41
4.2 Categories of Applications	44
4.2.1 Mobile Devices Applications and Gaming	44
4.2.2 Infrastructure Applications	44
4.2.3 IoT Device Applications	44
4.2.4 Human Applications	45
4.3 Stakeholder Roles	45
4.3.1 Service Developer	46



4.3.2	Service Operator	46
4.3.3	Infrastructure Provider	46
4.3.4	RAINBOW Developer	46
5	<i>RAINBOW Questionnaire and Stakeholder Interviews</i>	<i>47</i>
5.1	Questionnaire Establishment.....	47
5.1.1	Business Perspective.....	47
5.1.2	Technical Perspective	47
5.2	Questionnaire Recipients.....	48
5.3	Stakeholder Interviews.....	49
5.4	Early Results - Discussion	54
6	<i>System's Functional & Non-Functional Requirements</i>	<i>56</i>
6.1	Types of Requirements	56
6.2	RAINBOW Functional Requirements	57
6.2.1	VP1 - Cloud Service Modelling	57
6.2.2	VP2 - Orchestration algorithms	59
6.2.3	VP3 - Efficient Data Storage, Querying and Processing.....	64
6.2.4	VP4 - Secure Zero-touch configuration.....	66
6.2.5	VP5 - Configuration Integrity Verification.....	68
6.3	User Roles to Functional Requirements Mapping.....	71
6.4	RAINBOW Non-Functional Requirements	73
7	<i>Conclusion.....</i>	<i>84</i>
8	<i>References</i>	<i>86</i>
9	<i>Appendix.....</i>	<i>92</i>



List of tables

Table 1: Requirements engineering processes	14
Table 2: Requirements elicitation techniques	15
Table 3: Classification of different stakeholder types with associated strategies for engagement.....	43
Table 4: Relevance, needs and benefits per application category	53
Table 5: Functional Requirements Relation to User Role.....	73



List of figures

Figure 1: RAINBOW Requirements Collection Methodology	19
Figure 2: Monolithic Legacy Enterprise Architecture vs Micro-service Architecture Approach.....	22
Figure 3: Service Graph Example	24
Figure 4: RAINBOW Trust Protocol Framework	26
Figure 5: Stakeholder Categories	40
Figure 6: Stakeholders and their respective interests	41
Figure 7: Interest vs Power Matrix for the 3 Categories of Stakeholders.....	43
Figure 8: Categories of Applications	45



List of abbreviations

AI	Artificial Intelligence
AIK	Attestation Identity Keys
AODV	Ad hoc On-Demand Distance Vector
API	Application programming Interface
AR	Augmented Reality
AWS	Amazon Web Service
BABOK	Business Analysis Body of Knowledge
CFG	Control-Flow Graph
CNC	Computer Numerical Control
CPU	Central Processing Unit
CoAP	Constrained Application Protocol
DAA	Direct Anonymous Attestation
DAG	Directed Acyclic Graph
DFG	Data Flow graph
DNS	Domain Name System
DSO	Distribution System Operators
DSR	Dynamic Source Routing
DTU	Data Terminating Unit
ECC	Elliptic-Curve Cryptography
FR	Functional Requirement
GaaS	Gaming as a Service
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HWMP	Hybrid Wireless Mesh Protocol
IDC	International Data Corporation
IoT	Internet of Things
IT	Information Technology
MAC	Media Access Control
ML	Machine Learning
MQTT	MQ Telemetry Transport or Message Queuing Telemetry Transport
OCF	Open Connectivity Foundation
OT	Operational Technology
PCR	Platform Configuration Register
QoS	Quality of Service
REST	REpresentational State Transfer
S-ZTP	Secure Zero Touch Provisioning
SDN	Software-Defined Networking
SHA	Secure Hash Algorithm
SLO	Service Level Objective
SME	Subject Matter Expert
SPI	Serial Peripheral Interface
TCG	Trusted Computing Group



TCP	Transmission Control Protocol
TLS	Transport Layer Security
TPM	Trusted Platform Module
TSO	Transmission System Operators
UAV	Unmanned Aerial Vehicle
UDP	User Datagram Protocol
VR	Virtual reality
WiMax	Worldwide Interoperability for Microwave Access
ZB	Zettabytes



Executive Summary

The vision of RAINBOW is to design and develop an open and trusted fog computing platform that facilitates the deployment and management of scalable, heterogeneous and secure IoT services and cross-cloud applications. With RAINBOW, fog computing can reach its true potential by providing the deployment, orchestration, network fabric and data management for scalable and secure edge applications, addressing the need to timely process the ever-increasing amount of data continuously gathered from heterogeneous IoT devices and appliances.

Deliverable D1.1 - RAINBOW Stakeholders Requirements Analysis, hereafter referred to simply as D1.1, presents a stakeholder analysis of the RAINBOW ecosystem. This encompasses the identification of stakeholders as well as the key drivers and the incentives that set the context in which solutions and services of the RAINBOW platform should be offered. Additionally, stakeholder analysis assesses the roles, expectations and potential benefits for different relevant stakeholders to understand how to leverage and engage them. The analysis of the stakeholder requirements and benefits will help the consortium, on the one hand to extract the functional and non-functional requirements of the platform and on the other hand to make future decisions for business plans. This document also provides valuable information to refine the development of RAINBOW platform and define best exploitation strategies.



1 Introduction

1.1 The RAINBOW Project

The idea of RAINBOW is the development of a fog/edge orchestration framework capable of maintaining the required QoS for applications running at the edge (or fog) level of the network, by identifying and optimizing in real time the resources, while satisfying application related constraints. Platform evaluation will be carried out through pertinent and straightforward use cases, that highlight the framework capabilities at the edge/fog level and according to certain constraints.

1.1.1 RAINBOW's Value Propositions

The focal points of RAINBOW project are outlined in the following numbered list (in future referred to as “RAINBOW's Value Propositions”):

1. **Cloud-service modelling** for fog/edge applications: it provides the theoretical framework for elaborating and solving in real-time constraint-satisfaction problems that relate to fog/edge QoS. Such constraints may refer to application/connectivity qualitative parameters (i.e. served requests/s, latency, throughput, jitter, etc.).
2. **Orchestration algorithms** are intended to perform the proper enactment at the orchestration level during runtime in order to maintain a proper QoS. For this purpose, an “on-line” version of the optimization problems, based on the aforementioned modelling, must be solved. In general, on-line multi-objective problems are computation intractable.
3. **Efficient data storage, querying and processing** aims to solve the problem of efficient query planning and query execution in datasets that are physically dispersed and whose (parallel) acquisition cannot be performed with equivalent guarantees.
4. **Secure Zero-touch configuration** of fog nodes built on top of existing transport-layer protocols for mesh networks. The emphasis on this research issue is to cope with the pressing need of secure device management and, more specifically the problem of zero-knowledge/collision-free node addition and/or removal in a mesh environment, comprising heterogeneous types of devices with different configuration characteristics.

Create trust enablers dealing with the **Configuration and Execution Integrity Verification** of fog applications. Such enablers will verify the integrity of any devices and applications during both their initial deployment and (runtime) execution phases.

1.1.2 RAINBOW Use Cases

RAINBOW evaluation will be carried out through three uses cases, that aim to affirm the framework capabilities at the edge/fog level and according to particular constraints.



Use Case 1- Human-Robot Collaboration in Industrial Ecosystems

Reliable indoor positioning enables several innovative location-based services, because such accuracy levels essentially allow for real-time interaction between humans and cyber-physical systems. Activity recognition, machine navigation (e.g., “shelf” level), geo-fencing and automated robotics are among services that yield safety-critical assembly processes and logistics. For safety-critical industrial IoT, for instance, real-time indoor localization services may monitor the flow of objects and detect human worker positioning, collaborating with machinery (e.g. heavy-payloads robots) to prevent collisions and accidents. Specifically, the production process demands the involvement of humans and robots to assembly heavy and complex entities like car engines or power supply units, with robots assisting on carrying these heavy products for assembly.

Use Case 2: Digital Transformation of Urban Mobility

A real-time geo-referenced notification system for vehicles traveling in urban areas about critical situations for the city mobility network, due to any possible cause (e.g., severe weather, failure of road infrastructure, huge congestion, pollution). The notification system will be designed to collect explicit or implicit alert signals issued by vehicles in urban areas. Explicit alert signals refer to those that are either triggered directly (i.e., manually) by the driver, who may want to report a road issue, such as a large pothole or any other surface irregularity not detected by on-board sensors; or, they may be triggered by on-board sensors (e.g., a skid sensor that detects the presence of ice). Implicit alert signals, instead, are the result of sensor fusion processes that may involve multiple cars, of different makers, all sending log data to the cloud, where AI/ML algorithms can infer alert conditions that should be notified (for example, cars turning on fog lights and other vehicles reporting temperatures approaching the dew point can trigger a fog alert). Each alert signal will be delivered with the available geolocalization information, allowing reports to be localized in the areas where the traffic disruption was detected.

Use Case 3: Power Line Surveillance via Swarm of Drones

Power line surveillance is essential for all high and medium power line operators. Today, most of inspections are carried out with aerial methods, with the use of both helicopters and ground patrols. Even if the introduction of drones for power line surveillance is still at an embryotic state, the perspective is at the same time interesting and challenging: a swarm of drones presents the obvious benefit of reducing the total time required to scan the entire power line infrastructure, but it subtends many significant challenges. The foremost challenge stands in drone autonomy, which is critical due to the following constraints: performing high quality image-taking is energy consuming which results in the frequent return of drones to their base station for recharging; in turn, the image analysis is performed offline after drones return to base without any indication if the images are sufficient (if not, the drone must repeat the same flight plan); moreover, although a swarm is used, currently drones do not communicate to coordinate routing alteration, image exchanging, terrain overlapping avoidance, etc.; in addition, surveillance of critical infrastructure, such as power grid in this scenario, requires data protection, high performance, optimized resource allocation, energy reduction and specific restrictions.



The main innovation of the use case is to move data processing on board. Thus, coordination of routing, image exchanges, terrain overlapping avoidance, etc. can lead to higher energy autonomy and monitoring capacity, while reducing overlapping during image gathering process.

1.2 Structure of the Deliverable

The work in this deliverable begins by determining the methodology and respective techniques for requirements elicitation. There is also a section dedicated to the technologies that drive and empower the ecosystem where RAINBOW platform envisages to act in and, at the end of the section, the underlying technologies are elucidated.

Furthermore, D1.1 describes the methodology and the process followed to achieve an in-depth analysis of stakeholders that might be interested in RAINBOW services and provides an early indication of their needs which are extracted through tele-interviews, questionnaires, technical sessions with the demonstrators and literature review. The extracted needs are translated into functional and non-functional requirements at the end of the deliverable.

D1.1, through a detailed analysis, provides the foundation and constraints that will drive the development of the RAINBOW reference architecture (D1.2). The specific needs, requirements and domain-related features of the RAINBOW's Use-Cases (D1.3) are also part of this analysis and, for this reason, a high-level description of each of them will be provided. The stakeholder analysis report is the main input to understand how to leverage and engage RAINBOW's external environment, which is a task that directly affects the communication and dissemination activities of the project (D7.6) and (D7.7).

The structure of the present document is as follows:

- **Section 2** analyses the methodologies and techniques for extracting functional and non-functional requirements for RAINBOW project.
- **Section 3** presents the technological axis and the state-of-the-art paradigms for edge/fog computing.
- **Section 4** identifies the key stakeholders for the project.
- **Section 5** illustrates the rationale behind RAINBOW questionnaire.
- **Section 6** presents the functional and non-functional requirements.
- **Section 7** draws the conclusions.

2 Requirements Methodologies

The intention behind requirement elicitation is to identify quality user requirements that can be implemented into software development projects. This chapter provides a comprehensive description of the available methods for eliciting stakeholder requirements in a defined environment. The section concludes with the requirement elicitation techniques used by RAINBOW project.

2.1 Requirements Elicitation Framework

The requirements engineering method will follow ISO/IEC/IEEE 29148:2018 which describes two main processes or practices:

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 (StRS)
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.	System Requirements Specification (SyRS) Software Requirements Specification (SRS)

Table 1: Requirements engineering processes

2.2 Techniques for Requirements Elicitation

Eliciting requirements is a key task in product development lifecycle and most precisely in business analysis. Because the requirements serve as the foundation for the solution to the business needs, it is essential that the requirements be complete, clear, correct, and consistent.

It is of utmost importance for the person or organization who is responsible to elicit requirements to be in a position to actively engage all affected stakeholders.

Eliciting requirements is not generally considered as a strictly one-off activity. For example, requirements may be elicited in interviews or requirements workshops during the elaboration phase of a project. Later, when those requirements are used to build and verify model(s) or product(s), gaps in the requirements may be discovered. This will then require eliciting details of those newly identified requirements.

BABOK [1] is a collection of standard business analysis practices, which are compiled in a detailed guide published by the International Institute of Business Analysis (IIBA). BABOK establishes a common framework which assists analysts develop an in-depth understanding of core concepts and stay up to date with developments. BABOK lists nine requirements elicitation techniques:

Elicitation Technique	Synonym
Brainstorming	
Document Analysis	Review existing documentation
Focus Groups	
Interface Analysis	External Interface Analysis
Interviews	
Observation	Job Shadowing
Prototyping	Storyboarding, Navigation Flow, Paper Prototyping, Screen Flows
Requirements Workshops	Elicitation Workshop, Facilitated Workshop
Survey/ Questionnaire	

Table 2: Requirements elicitation techniques

2.2.1 Brainstorming

Brainstorming is a technique intended to produce a broad or diverse set of options. Its sessions help answer specific questions such as (but not limited to):

- What options are available to resolve the issue at hand?
- What factors are constraining the group from moving ahead with an approach or option?
- What could be causing a delay in activity 'A'?
- What can the group do to solve problem 'B'?

Brainstorming works by focusing on a topic or problem, and then coming up with many possible solutions to it. This technique is best applied in a group as it benefits from the experience and creativity of all members of the group. To increase creativity, participants are encouraged to use new ways of looking at things and are free to associate them in any direction. Facilitated properly (without censoring ideas) and executed with the right audience (representatives of each group, SMEs, stakeholders), brainstorming can be fun, engaging and productive.

2.2.2 Document analysis

Document analysis is a mean to elicit requirements by studying available documentation on existing and comparable solutions, identifying relevant information. It may include the analysis of business plans, market studies, contracts, requests for proposal, statements of work, memos, existing guidelines, procedures, training guides, competing product literature, published comparative product reviews, problem reports, customer



suggestion logs, and existing system specifications, among others. Identifying and consulting all the potential sources of requirements will result in improved requirements coverage, assuming the documentation is up to date.

While using this technique and without breaking any non-disclosure clauses or copyright laws, one could review documentation from competitors and other industries that have similar systems.

2.2.3 Focus Groups

A focus group is a mean to elicit ideas and attitudes about a specific product, service or opportunity in an interactive group environment where the participants share their impressions, preferences and needs, guided by a moderator. It is composed of pre-qualified individuals whose objective is to discuss and comment on a topic and represents an opportunity for individuals to share their own perspectives and discuss them in a group setting. The approach could lead participants to re-evaluate their own perspectives in light of others' experiences. A trained moderator manages the administrative pre-work, facilitates the session and produces the report. Observers may record or monitor the focus group but cannot participate in it.

A focus group can be utilized during any life-cycle state: exploratory, under development, ready to launch, or in production. If the group's topic is a product under development, the group's ideas are analyzed in relationship to the stated requirements. This may result in updating existing requirements or identifying new ones. If the topic is a completed product that is ready to be launched, the group's report could influence how to position the product into the market. If the topic is a product in production, the group's report may provide direction on the revisions to the next releases of requirements. A focus group may also serve as a means to assess customer satisfaction with a product or service.

The work of a focus group may be similar to that done in a brainstorming session, : one is that a focus group is typically more structured; another difference is that a brainstorming session's goal is to actively seek broad, creative, even exaggerated ideas.

2.2.4 Interface analysis

This technique consists of identifying interfaces between solutions and/or solution components and define requirements that describe how they will interact.

Interface types include:

- User interfaces, including human users directly interacting with the system, as well as reports provided to the user.
- Interfaces to and from external applications.
- Interfaces to and from external hardware devices.

Interface analysis helps to clarify the boundaries of the interfacing applications. It distinguishes which application provides specific functionality along with the input and output data needs. By clearly and carefully separating the requirements for each



application while defining the shared interface requirements, a basis for successful interoperability is established.

Identifying what interfaces are necessary to support an application sets the stage for eliciting a wide variety of requirements. Early identification of interfaces uncovers and confirms the interfacing stakeholders and provides a framework for subsequent analysis of the detailed requirements for each interface. Interface analysis is certainly necessary for a software solution or solution component but can also be useful for a non-software solution, such as when defining requirements for deliverables that will be produced by third parties.

2.2.5 Interviews

In an interview, the interviewer formally or informally directs questions to a stakeholder in order to obtain answers that will be used to create formal requirements. One-on-one interviews are common. In a group interview (with more than one interviewee in attendance) the interviewer must be careful to elicit responses from all attendees. For the purpose of eliciting requirements, interviews are of two basic types:

- **Structured Interview:** where the interviewer has a pre-defined set of questions and is looking for answers.
- **Unstructured Interview:** where, without any pre-defined questions, the interviewer and the interviewee discuss topics of interest in an open-ended way.

2.2.6 Observation

Observation is primarily useful for capturing what's already in existence. It relies on studying people performing their jobs and is sometimes called "job shadowing" or "following people around." For instance, some people have their work routine down to such a habit that they have difficulty explaining what they do or why. The observer may need to watch them perform their work in order to understand the flow of work. In certain projects, it is important to understand the current processes to better assess the process modifications that may be needed. There are two basic approaches for the observation technique:

- **Passive/invisible:** In this approach, the observer observes the user working through the business routine but does not ask questions. The observer records what is observed, but otherwise stays out of the way. The observer waits until the entire process has been completed before asking any questions. The observer should observe the business process multiple times to ensure they understand how the process works today and why it works the way it does.
- **Active/visible:** In this approach, while the observer observes the current process and takes notes they may dialog with the user. When the observer has questions as to why something is being done as it is, they ask questions right away, even if it breaks the routine of the user.

2.2.7 Prototyping

It is frequent that users and business owners don't know what they want until they see it or alternatively see something they don't want. It is where prototyping comes handy.



Prototyping details user interface requirements and integrates them with other requirements such as use cases, scenarios, data and business rules. Stakeholders often find prototyping to be a concrete means of identifying, describing and validating their interface needs. Prototyping can be categorized in two ways:

- **Functional Scope.** A horizontal prototype models a shallow, and possibly wide view of the system's functionality. It typically does not have any business logic running behind the visualization (i.e. a mock-up). A vertical prototype models a deep, and usually narrow slice of the entire system's functionality.
- **Usage Throughout System Development Lifecycle.** A "Throw-away" prototype seeks to quickly uncover and clarify interface requirements using simple tools, sometimes just paper and pencil. As the name suggests, such a prototype is usually discarded when the final system has been developed. The focus is on functionality that is not easily elicited by other techniques, has conflicting viewpoints, or is difficult to understand. An "Evolutionary or Functional" prototype extends the initial interface requirements into a fully functioning system and requires a specialized prototyping tool or language. This prototype produces a working software application.

A prototype cannot typically be developed too early in the project - analysts need to have gathered some ideas of where they are going before a prototype is feasible.

2.2.8 Requirements Workshops

A requirements workshop is a highly productive focused event attended by carefully selected key stakeholders and subject matter experts for a short, intensive period (typically one or a few days). The workshop is facilitated by a team member or ideally, by an experienced, neutral facilitator. A scribe (also known as a recorder) documents the requirements elicited as well as any outstanding issues. A requirements workshop may be used to generate ideas for new features or products, to reach consensus on a topic, or to review requirements. Other outcomes are often detailed requirements captured in models.

2.2.9 Survey/Questionnaire

A survey (may also be referred to as a questionnaire) is a means of eliciting information from many people, sometimes anonymously, in a relatively short period of time.

A survey administers a set of written questions to the stakeholders and subject matter experts. Alternatively, respondents are provided with a series of statements and asked for their level of agreement or endorsement. Their responses are analyzed and distributed to the appropriate parties. Questions in a survey are of two types:

- **Closed:** The respondent is asked to select from available responses. This is useful when the range of user's responses is fairly well understood, but the strength of each response category needs to be determined. The responses to closed questions are easier to analyze than those gained from open-ended questions, because they can be tied to numerical coefficients.
- **Open-ended:** The respondent is free to answer the questions as they wish. Useful when the issues are known but the range of user responses to them is not. The

responses to open-ended questions may provide more detail and a wider range of responses than those gained from closed-ended questions. However, open-ended questions are more difficult to quantify and summarize as they often include qualitative, rather than quantitative, language.

2.3 RAINBOW Requirements Collection Methodology

RAINBOW project employed five different requirement elicitation techniques, namely:

- Brainstorming:
- Document analysis:
- Interviews
- Survey/Questionnaire
- Focus Groups

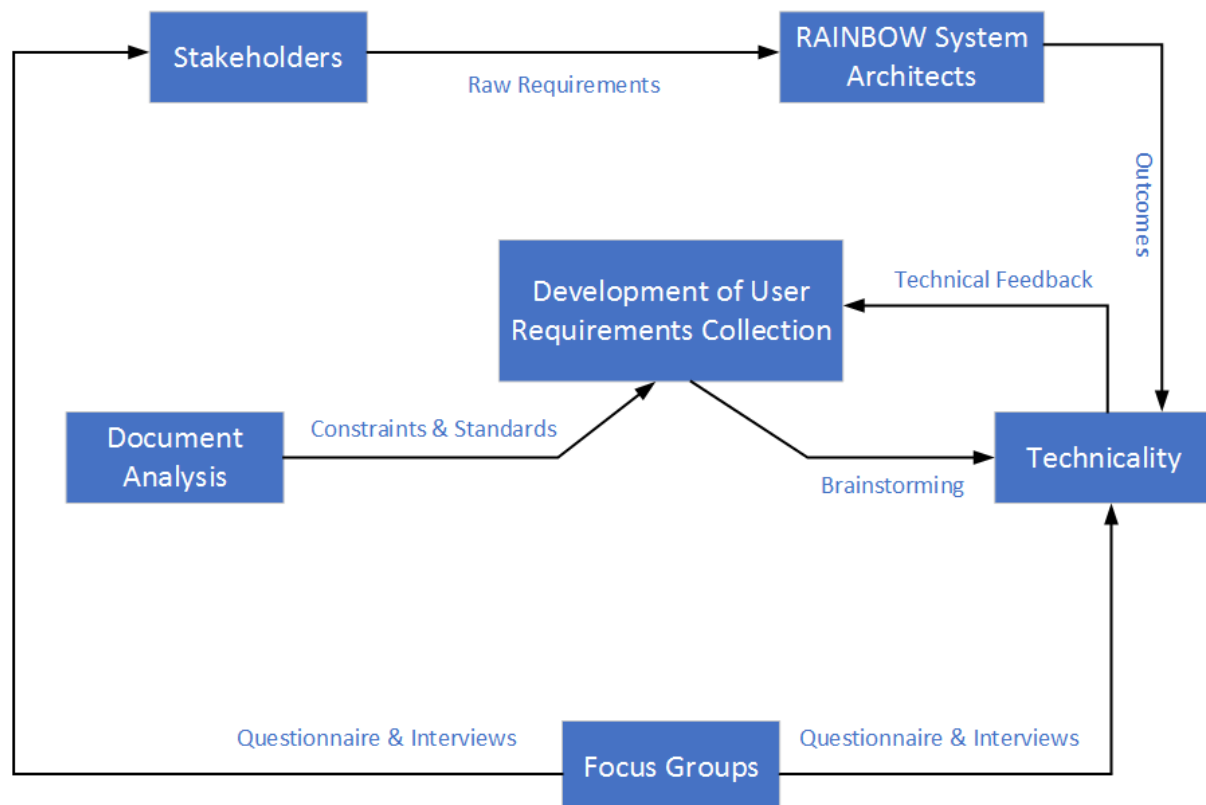


Figure 1: RAINBOW Requirements Collection Methodology

Requirement elicitation focuses on examining and gathering desired requirements and objectives for the system under different stakeholder perspectives. RAINBOW approach for the specification of system requirements starts with interviewing the different types of stakeholders through interviews and questionnaires. Eliciting requirements through these two techniques results in a type of raw requirements. Raw requirements are requirements that have not been analysed and have not yet been written down in a well-formed requirement notation [2]. RAINBOW system architects collect the raw



requirements and then run an iterative internal process in order to produce some outcomes. Thereafter, these outcomes are compared with the technicality of the system and produce the good and necessary requirements for the development of RAINBOW platform. However, the requirements derived from interviews and questionnaires can sometimes be ambiguous since the system architects may misinterpret the user's needs, or the user may not understand a question or lack the technical knowledge to answer it. Therefore, in addition to requirements gathering, a document analysis through a thorough industry and literature review has been conducted by RAINBOW consortium, in order to validate the requirements as well as identify standards and constraints which play an important role in the attempt to receive requirements of high quality.

The aforementioned techniques were constantly reviewed and enriched by the corresponding RAINBOW focus groups. These groups took place in the form of dedicated teleconferences between specific partners or the entire RAINBOW consortium. RAINBOW focus groups also provided valuable feedback on the questionnaire respondents as well as the interview participants since the prosperity of user requirements elicitation depends heavily on the knowledge and maturity of the stakeholders.



3 Technology Analysis and State of the Art

Before proceeding with the stakeholder identification and the requirement collection and analysis process, it is important to identify and elaborate on the key concepts driving the innovative technological axes of the RAINBOW project. The terminology determined in this section will work as a reference guide across all future RAINBOW technical deliverables.

3.1 Underlying technologies

3.1.1 Fog/Edge Computing

In recent years, with the proliferation of the Internet of Things (IoT) and the wide penetration of wireless networks, the number of edge devices and the data generated from the edge have been growing rapidly [13]. According to International Data Corporation (IDC) prediction [14], global data will reach 180 zettabytes (ZB), and 70% of the data generated by IoT will be processed on the edge of the network by 2025. However, with the growing volume of data generated at the logical extremes of the network, and the fact that network bandwidth is simply not scaling at the same speed as with computing power, data mitigation is becoming a bottleneck constraining the cloud computing paradigm for delay-sensitive IoT services. Fog Computing promises lower response times for IoT services while provides the necessary computational resources, closer to the users, on the path between IoT devices to the cloud [3]. Fog Topology is organized hierarchically with IoT devices at the bottom and the cloud on top [4] [5].

Since the Fog computing is a relative new topic, there are many definitions of it. In this document we consider the following definitions of Fog/Edge computing and its three-tier hierarchical architecture, as presented in [13].

Edge/Fog computing is a new paradigm in which the resources of an edge server are placed at the edge of the Internet, in close proximity to mobile devices, sensors, end users, and the emerging IoT. Terms such as “cloudlets,” “micro data centers,” and “fog” have been used in the literature to refer to these types of small, edge-located computing hardware. They all represent counterpoints to the theme of consolidation and massive data centers that have dominated discourse in cloud computing.

Three-Tier Fog Computing Model: By analyzing several representative application scenarios of fog computing we abstract a typical three-tier edge computing model: IoT, edge, and cloud. The first tier is IoT, including drones, sensors in the connected health area, devices and appliances in the smart home, and equipment in the industrial Internet. Multiple communication protocols are used to connect IoT and the second tier, edge. For example, drones can connect to a cellular tower by 4G/LTE, and sensors in the smart home can communicate with the home gateway through Wi-Fi. Edge, including connected and autonomous vehicles, cellular tower, gateway, and edge servers, requires the huge computing and storage capabilities of the cloud to complete complex tasks. The protocols between IoT and the edge usually have the characteristics of low power consumption and

short distance, while the protocols between the edge and the cloud have large throughput and high speed. The Ethernet, optical fibers, and the coming 5G are the preferred communication protocols between the edge and the cloud.

Since a fog is made up of heterogenous devices [3] [6], ranging from low power sensors to powerful servers, the communication technologies used are also diverse, e.g., “ZigBee, Wi-Fi, 2G/3G/4G, WiMax, 6Lowpan and so on” [7]. There are several communication protocols that are used in fog computing, e.g., HTTP, CoAP, or MQTT [8]. The Constrained Application Protocol (CoAP) has been designed for machine to machine applications on devices with high resource constraints. Originally it uses UDP, but extensions allow using TCP, TLS, and WebSockets as well [9] [10]. It targets environments, where a small code size is required or where network bandwidth must be kept as low as possible. It targets environments, where a small code size is required or where network bandwidth must be kept as low as possible. The Open Connectivity Foundation (OCF) [12] provides specifications for allowing IoT devices to communicate, regardless of the technologies they use. They have devised a framework and a bridging specification for connecting third party technologies to this framework and already provide detailed bridging specifications for a variety of technologies, such as AllJoyn, Bluetooth Low Energy, UPlus, or Zigbee.

3.1.2 Microservices architectural paradigm

The evolution of new software development paradigms is following the need for development of applications that adhere to the notions of modularity, distribution, scalability, elasticity and fault-tolerance [15]. A micro-service architectural approach is considered as the resulting set that arises from the decomposition of a single application into smaller pieces (services) that tend to run as independent processes and have the ability to inter-communicate usually using lightweight and stateless communication mechanisms (e.g., RESTful APIs over HTTP) [16]. These (micro-) services are built around business capabilities and are independently deployable by fully automated deployment machinery. For (micro-) services, there is a bare minimum of centralized management and such services may be written in different programming languages and even use different data storage technologies [17].

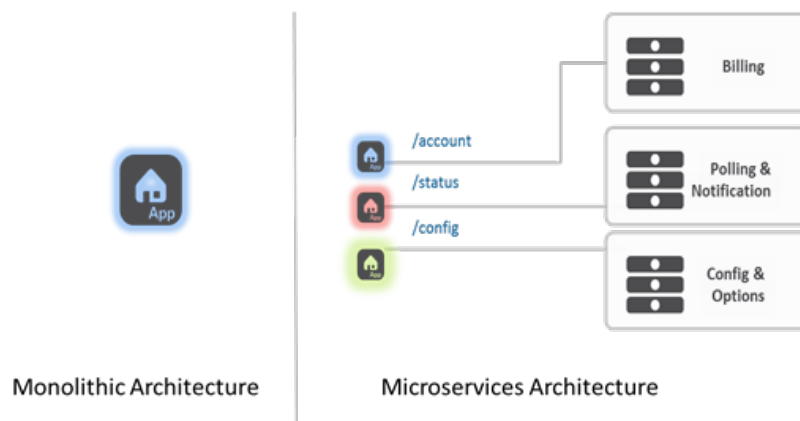


Figure 2: Monolithic Legacy Enterprise Architecture vs Micro-service Architecture Approach



To understand the logic behind a micro-service architectural approach it is useful to compare it to a monolithic approach (Figure 2) where a single executable hosts the entire functional logic of an application, such as in the case of a web service handling HTTP requests while responsible for executing domain logic, database access, and HTML view population. Hence, all logic for handling web requests runs within a single process. However, this approach features a number of disadvantages, often referred to as monolith inhibitors [18]. In particular, feature roll-outs and software code changes are always tied together – even a single change made to a small code segment of the application, requires the entire monolith to be rebuilt and re-deployed. Over time, and as the software stack expands, it becomes evident that a good modular structure is hard to keep, making it difficult to track software code changes that ought to only affect one module within that module. Most importantly, resource capacity provisioning for the software stack requires scaling the entire application rather than only the specific services in real need of additional resources [19].

In contrast to monoliths, micro-services are decomposed into services organised around discrete business capabilities. The boundaries between these units are usually comprised of functional APIs that expose the core capabilities of each service. Large systems are then composed of many (micro-) services, whereby communication between micro-services is a central ingredient. For instance, such is the case of amazon.com, where the different aspects of their e-commerce platform —recommendations, shopping cart, invoicing and inventory management— are split into discrete, scalable and independent (micro-) services [20]. Instead of all being part of one enormous monolith, each business capability is a self-contained service with a well-defined interface. The advantage of this is that separate teams are each responsible for different aspects of the service allowing the team and software core to develop, test, handle failures and scale independently. In turn, continuous delivery is possible as small units are easier to deploy and manage their entire lifecycle.

Finally, decentralized data management is highly evident where each service dealing with a specific function of the business process may manage its own database, either different instances of the same database technology or entirely different database systems, so as to optimize data storage, processing and acquisition to the heterogeneous needs and scale of each business function. As stated by A. Cockcroft, who oversaw Netflix's transition from a monolithic DVD-rental company to a micro-service architecture comprised of many small teams working together to stream content to millions of users, a micro-service with correctly bounded context is self-contained for the purposes of software development [7]. Therefore, one can understand and update the micro-service's code without knowing anything about the internals of its peers, because the micro-services and its peers interact strictly through APIs and therefore there is no need for sharing or exposing (with security threats lurking) data structures, database schemata, or other internal representations of objects. Thus, the commonly understood “contract” between micro-services is that their APIs are stable and forward compatible.

3.1.3 Service Graph Topology Descriptions

Since a microservice architecture is decomposed into small units that depend on each other, it is beneficial to model such an architecture as a graph. The service graph is a topology-aware descriptive model where nodes represent the services composing an application and directed edges represent the relationship(s) and inter-dependencies between services. Nodes can be additionally annotated by a number of attributes denoting service-specific characteristics including configurations, optimizations, constraints and requirements. In turn, directed edges representing relationships and interdependencies denote the service deployment order (e.g., database backend must be bootstrapped before front-end), resource exchange protocols (e.g., ZigBee) and can also be annotated with service-to-service specific characteristics such as the configuration shown in Figure 3 (A/B testing). In this figure, the product service is configured to direct 90% of the incoming requests to the current version of the reviews service while 10% of the traffic is directed to a newer (untested) version. One can also observe that v1 is phased out and although still part of the service graph no incoming requests are directed to that version.

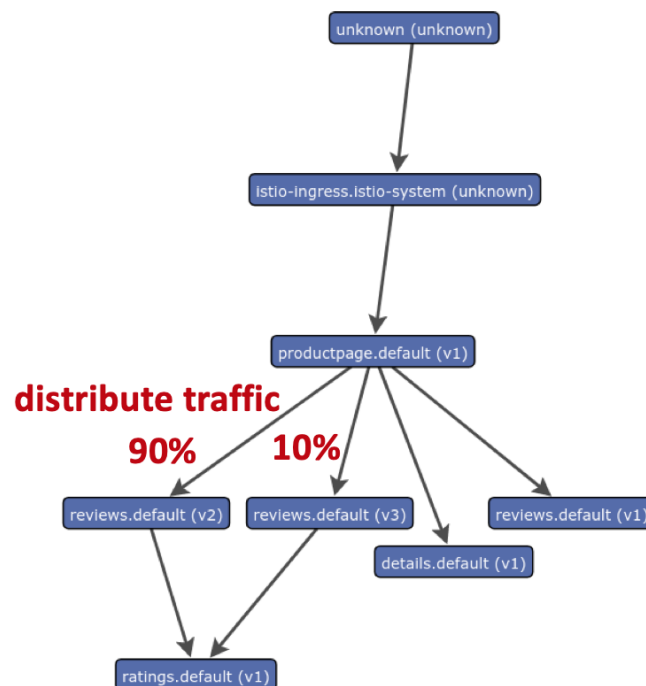


Figure 3: Service Graph Example

The service graph plays a prominent role in RAINBOW as users will be able to graphically describe their application topology, denote intercommunication and dependencies among services, and annotate the description with constraints, optimization policies and QoS requirements, simply denoted as “Configurations”. Decoupling constraint definition from the business logic allows for minimal code intrusion while hiding the complexity behind the operation of each Configuration in the background to free users by not affecting code development and debugging. Most importantly, users are not constantly derailed and can focus on core application development leaving enforcement to the



RAINBOW platform. These are concepts coming from Model-Driven Engineering, which helps architects with designing, documenting, and understanding architectures [3].

In turn, through the service graph modelling one can immediately validate the semantic correctness of the defined Configurations. This task is performed to detect potential problems such as unreachable edges due to antagonizing Configurations, inaccessible nodes, and circular dependencies, leading to a situation in which no valid evaluation order exists, because none of the Configurations in the service graph can be orderly evaluated. This process, while not exhaustive, is an important aspect at a pre-deployment phase, as detecting unforeseen problems can prevent successful deployments before resources are even (billed) and provisioned. A similar approach is taken in [4], where a service dependency graph is used for analysing risky service invocation chains.

3.1.4 Secure service mesh networking (and routing)

The problem of establishing and maintaining a secure mesh network is rather challenging. The reason for that is that in a mesh environment there is no objective root of trust that can be used to verify the integrity and trustworthiness of the participating nodes. In addition, even if all nodes that participate in a mesh were verified in advance from a hypothetical trust anchor, several adversarial scenarios that relate to the manipulation of the routing paths or the managed traffic do exist. Hence, it goes without saying that security considerations are multi-fold in these types of networks.

- Firstly, the routing protocols that support mesh networks differ significantly from the fixed networking protocols. Fixed networking protocols are handled by dedicated centralized routers that expose minimum attack vectors. Centralized routing is not suitable for mesh networks, due to their nature. mesh networks are unpredictable and mobile-by-definition. Therefore, routing paths have to be established on-demand and upon their establishment their validity is questionable upon a short period of time. Dedicated protocols undertake the task to perform the process of route-identification (i.e. path discovery) and route-usage. Such protocols include: AODV [21], DSR [22] or more lately HWMP [23]. It should be clarified, that the scope of these protocols are only to satisfy the operational needs of routing. As such they do not provide security guarantees since this is outside their operational purpose. Indicatively. If someone, tampers the advertised routes of a mesh node s/he could trigger a man in the middle attack and lead other members of the mesh to pass their traffic through the exploited node. It should be clarified, that the scope of these protocols are only to satisfy the operational needs of routing: as such they do not provide security guarantees, since this is outside their operational purpose. Indicatively. If someone, tampers the advertised routes of a mesh node s/he could trigger a man in the middle attack and lead other members of the mesh to pass their traffic through the exploited node.

In order to prevent such scenarios, high-level cryptographic schemes and protocols can be used to make the communication process as immune as possible to these types of attacks. Several overlay protocols have been proposed such as the B.A.T.M.A.N. (better

approach to mobile ad-hoc networking) [24] or the CJDNS [25]. These protocols are specialized on creating multiple secure overlays on top of an existing mesh topology. However, their operation pre-requisites a consistent way to infer the identity of the node and a way to pre-deliver cryptographic keys.

The aforementioned assumptions are not perfectly suited to the functional requirements of RAINBOW since in RAINBOW the way mutual trust will be achieved is a critical issue.

3.1.5 Scalable trust establishment and attestation

RAINBOW will include the provision of secure, robust, and efficient run-time behavioural attestation and verification methods to check the internal state of an untrusted fog-based environment towards establishing its trustworthiness and privacy. The endmost goal is to establish “**fog/edge node communities of trust**”. To do so, RAINBOW will develop a trusted framework for **attestation and system assurance**. At a high level, the framework will enable fog/edge entities to establish and maintain trust during the entire system life-cycle. Typically, this stems from establishing **roots of trust** in components, and using these roots of trust to establish and maintain trust relationships. Once a trusted community is materialised, secure community communications can be established and used to provide trusted community-wide system updates. Thus, using the concept of a trusted community, trusted communities of communities can be created within a fog-based environment.

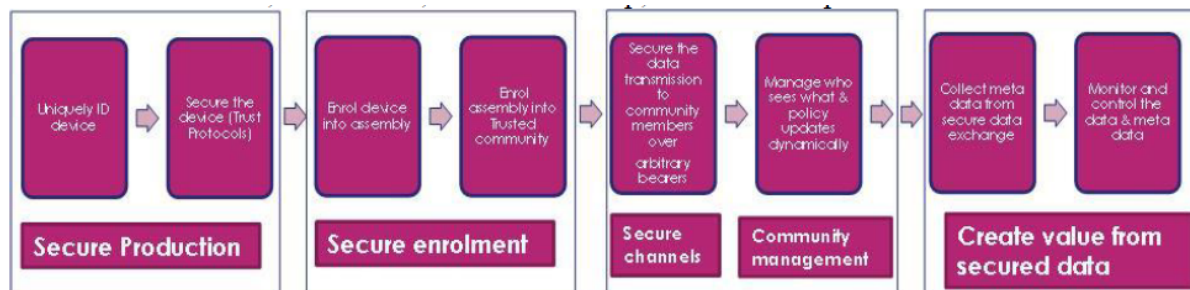


Figure 4: RAINBOW Trust Protocol Framework

This is considered as one of the main goals towards “**security and privacy by design**” solutions, including all methods, techniques, and tools that aim at enforcing security and privacy at software and system level from the conception and guaranteeing the validity of these properties. For privacy, RAINBOW will leverage advanced crypto primitives, namely **Direct Anonymous Attestation (DAA)** [26], whereas for security and operational assurance, it will enable the provision of **Control Flow Attestation**.

The reason behind employing attestation mechanisms as a means of operational assurance is twofold: First of all, one of the main challenges in managing device and network security in today’s heterogeneous and scalable infrastructures is the lack of adequate containment and sufficient trust when it comes to the behaviour of a remote system that generates and processes mission-critical and/or sensitive data. An inherent property in RAINBOW is the codification of trust among computing entities that potentially are composed of heterogeneous hardware and software components, are



geographically and physically widely separated, and are not centrally administered or controlled. By leveraging the artefacts of traditional security infrastructure (such as digital signatures, certificates and assurance statements) coupled with advanced crypto primitives (such as run-time property-based attestation) and building upon emerging trusted computing technologies and concepts, RAINBOW will convey trust evaluations and guarantees for each network entity.

This high level of trustworthiness which will not only include integrity of system hardware and software but also the correctness and integrity of the generated data flows will, in turn, reduce the overall attack vector and allow for the more effective operation of the RAINBOW security framework. This will allow the secure configuration, deployment and operation of distributed, scalable service graph chains.

In the paradigm of Trusted Computing, relevant to the RAINBOW Security and Trust Establishment service, root of trust will be bootstrapped from a small dedicated piece of secured hardware, the **Trusted Platform Module (TPM)**, onto a more complex computer system. Today, TPMs are very widespread as the major computer manufacturers are integrating them into servers, desktop and notebook computers. Still, the idea of using TPMs also in edge/fog devices only recently gained attention.

Due to size requirements some edge/fog devices might emulate the functionality of a TPM in software [27], [28]. This might lead to security risks and a higher power consumption caused by CPU-intensive cryptographic operations.

To construct trusted communities, the following mechanisms are vital in order to establish trust: **a) secure identity provisioning; b) secure key management and distribution** to support secure communication; and **c) attestation methods**, to enable one node to prove its **trustworthiness**, including the software installed, to another node, and **d) secure update mechanism**. In this context, TPMs are mainly used for **secured boot, remote attestation, and secured communication** [29] Secured identities are essential for implementing cryptographic measures used to protect against digital threats. These include measures to prevent unintended software updates on computers in automation systems as well as protection mechanisms for stored data and communication networks. To protect industrial automation systems, secured identities are fundamental for the entire chain of security measures. They apply secret keys and cryptographic processes that use secret keys. As it is rather easy for attackers to extract secret keys from software key stores, several companies provide hardware-based security solutions [30].

TPMs are standardized hardware components that can secure certain critical functionalities by offering the following features:

- **Protected capabilities:** abilities to execute commands with access shielded locations (e.g. protection and reporting of integrity measurements and secured key storage)
- **Integrity measurement, storage, and reporting.**



Typically, the main building blocks a TPM offers are:

- A secured storage of keys in volatile and non-volatile memory,
- Platform configuration registers (PCRs) to store the current hardware and/or software state of the system,
- Cryptographic co-processors (e.g. encryption/decryption), and a
- True random number generator

The capability to record the current system state is a distinctive feature of the TPM. This is achieved by cryptographically hashing a software component and storing the resulting measurement value in a specially-protected Platform Configuration Register (PCR). PCRs are reset at platform boot. PCRs can only be written to via the non-invertible, non-commutative extend operation. For each call, a PCR with index i in state t is extended with measurement x by setting:

$$PCR_i(t + 1) = Hash(PCR_it || x)$$

Thus, the TPM's PCRs can potentially be used to exactly describe the software executed on a machine by following a transitive trust model, in which each software component is responsible for measuring its successor before handing over control [31]. For the TCG's technical realization to work, each calling code needs to compute a hash value of the expected next executable code and to extend a PCR with the result, before control is passed to that subsequent, and thus measured code. In the simplest case, this is done starting from the firmware, covering boot loader, kernel, and system libraries etc., up to application code and configuration files. Ultimately, the exact configuration of the platform is mapped to a set of PCR values; a so-called chain-of-trust is established.

Considering the complexity in edge/fog computing, a TPM cannot guarantee perfect security; however, it can, with the chain-of-trust, provide strong evidence that certain expectations on the edge/fog devices behaviour may be fulfilled. An edge/fog may thus be trusted if the deciding entity accepts the provided evidence (e.g. PCR values signed under a key protected by a certified TCG-compliant TPM) as credible and sufficient for a given situation and environment.

Using these features the following core services can be realized with a TPM and may be considered the basis for most trusted computing usage scenarios.

- **Secured Key Store:** Software functions (e.g. secured communication protocols) can use the TPM as a secured key store. The software executes the cryptographic operations, whereas the hardware stores the keys.
- **Binding:** This operation encrypts data using a key that is managed by a particular TPM. Only one specific TPM is able to decrypt the data.
- **Sealing:** Similar to binding, sealing limits the decryption of data to a specific TPM, and to a specific platform configuration (hash stored in the PCR registers). Thus, only if the systems integrity is as expected at the time of encryption, the data is decrypted.
- **Secured Boot:** In a feature often also referred to as "Secure Boot", a TPM can support a secured boot mode by assessing the integrity of software that is running

on a device. Here, the TPM serves as root of trust for measurement and at each stage of the boot, the TPM-assured current state measurement is compared with the (signed) expected hash of the current system configuration. Only if the calculated hash matches the expected configuration, boot is allowed to continue. This process is extended into the operating system. One way to implement a secured boot mechanism is by employing TPM Sealing at several intermediate steps of the boot process.

- **Measured Boot:** In contrast to secured boot, measured boot does not check the validity of the measured hashes of the current stage, but just stores them in the PCRs of the TPM. After booting the TPM can reliably report this configuration. Then the system can base certain decisions on this information. For example, it could only allow certain functionalities if there is a specific configuration, or it can use the measured boot configuration for remote attestation.
- **Remote Attestation:** Here, a platform authenticates its hardware and software configuration to a remote entity. A remote system can then determine the level of trust based on the identity and integrity of the attestee. Architectures for remote attestation basically consist of integrity measurement and the remote attestation protocol. A TPM can be used to perform and store the integrity measurement and to sign the resulting PCR values with a key which can be related to a so-called Endorsement Key issued by the TPM-manufacturer, thus proving the report comes from an authentic TPM. The decisive feature here is specific signature keys (AIK - Attestation Identity Keys) which are always under the protection of the specific TPM are used. To protect privacy, AIKs can be blinded either by a Privacy Certification Authority (PrivacyCA) [32], with the drawback being that the PrivacyCA needs to be trusted by all users of the Remote Attestation. The advanced cryptographic protocol of Direct Anonymous Attestation [33] replaces the PrivacyCA instance by a cryptographic evidence that a given AIK is from a certain, potentially very large, group of trusted TPMs.

Overall, remote attestation is a means of integrity verification of software running on a remote device. It is a mechanism, typically realised as a challenge-response protocol, which enables a trusted party (verifier) to obtain an authentic, accurate and timely report about the software state of a potentially untrusted remote device (prover). The verifier then checks whether the reported state is trustworthy, i.e., whether only benign software is loaded on the prover.

On the privacy side, **Direct Anonymous Attestation (DAA)** is a platform authentication mechanism that enables the provision of privacy-preserving and accountable authentication services. DAA is based on group signatures, which give strong anonymity guarantees [34]. The key security and privacy properties documented in [35], [36], [37] are:

- **User-controlled Anonymity:** Identity of user cannot be revealed from the signature.
- **User-controlled Linkability:** User controls whether signatures can be linked.
- **Non-frameability:** Adversaries cannot produce signatures originating from a valid TPM.



- **Correctness:** Valid signatures are verifiable, and linkable, where needed.

In a nutshell, DAA is essentially a two-step process where, firstly, the registration of a TPM is executed once, and during this phase the TPM chooses a secret key (SETUP). This secret key is stored in secure storage so that the host cannot have access to it. Next, the TPM talks to the issuer so that it can provide the necessary guarantees for its validity (JOIN). The issuer then places a signature on the public key, producing an AIK *<cre>*. The second step is to use this *<cre>* for anonymous attestations on the platform (SIGN), using Zero-Knowledge Proofs [38]. These proofs convince a verifier that a message is signed by some key that was certified by the issuer, without knowledge of the TPM's DAA key or AIK *<cre>* (VERIFY). Of course, the verifier has to trust that the issuer only issues *<cre>*s to valid TCs. More details on the underpinnings of each one of these phases and various proposed DAA schemes will be documented in D2.2.

On the security side, there exist different kinds of attestation, particularly **static attestation** and **dynamic attestation** [39]. Static attestation allows the attestation of static properties and configuration of a remote platform. The most prominent example is the attestation of the integrity of binaries [40]. As the name implies, dynamic attestation deals with dynamic properties of the runtime. For instance, it is concerned about the execution and data flow of programs, and not the static integrity of binaries. Naturally, attesting dynamic properties is significantly more challenging than attestation of static (already known) properties. Hence, the majority of research has focused on static attestation including industry effort in the Trusted Computing Base introducing secure and authenticated boot loading mechanisms of operating systems. However, given the continuous attacks on dynamic properties such as zero-day exploits which corrupt program's control flows, static attestation alone cannot be considered a viable security solution in the long-run, and needs to be enhanced with advanced dynamic attestation mechanisms.

There does not yet exist a comprehensive design nor an effective as well as efficient implementation to enabling dynamic attestation. The most prominent approach in this context is **Control Flow Attestation** [41], [42]. Control Flow Attestation is one of the most important dynamic properties at the software layer since it captures diverse instantiations of software exploits that hijack a program's control flow. In RAINBOW, we will develop automated and scalable behavioural-based attestation techniques focusing on the attestation of properties of software and hardware for fog-based environments. For this, we plan to adopt and extend static and dynamic attestation techniques so that both static and run-time properties of a remote platform can be attested.

In conclusion, Trusted Computing mechanisms as they are today can, especially when combined, offer strong hardware-based security services to edge and fog computing systems. Managing identities and privacy is a significant challenge to be tackled in the RAINBOW project.

3.1.6 Geo-distributed data processing

Geo-distributed data analytics have gained ground due to their ability to achieve fast response time, high privacy level and easier failure recovery, while capitalizing on resources that are physically located at different places and might be heterogeneous. In these settings, data is being produced and analyzed in data sources and computing nodes that are distributed in locations all over the world, therefore latency, data transmission, fault tolerance and privacy issues arise. Much work has been done towards optimizing the logical and physical plan of such analytics; the current state-of-the-art mostly focuses on minimizing metrics like bandwidth usage, latency, throughput and response time. The minimization of such metrics is important to ensure fast and reliable results with the lower resource consumption (i.e., cost) possible.

A common way of representing/modelling an analytics job is with the help of a DAG (directed acyclic graph), where each node represents an analysis stage and each edge the data transfer between stages (e.g., data shuffling in Spark). A way of optimizing the DAG is through finding the optimal placement of stages to a set of underlying compute nodes. The main idea behind the placement is usually to place the tasks closer to where the data they need is stored or where the previous or next stage will take place. Each stage can be divided to multiple tasks that run in parallel in multiple computing nodes. Due to the NP-hardness of finding the optimal placement of the tasks of the stages to computing nodes, while ensuring the optimality of the whole graph, the problem is usually tackled in a heuristic manner. Some works deal with that by optimizing each stage independently [43]. Using this technique, the optimal placement of a stage can be easily found but as a local optimum does not ensure a global optimum, the final solution may deviate from the optimal one arbitrarily. These techniques usually come with a small computation complexity and cost overhead. Also, the placement of a stage can be found using Linear Programming Formulas [44] [45] where the constraints of the problem (computing nodes' characteristics and network speed) are considered. Having this initial placement, one can further improve it using heuristic algorithms. A family of algorithms that produce an acceptable result is the Local Search one, which includes algorithms like Iterated Local Search [46] and Hill Climbing [47]. Another common technique of finding a placement is by placing the stages in virtual cost spaces and using algorithms like spring relaxation for the mapping to computing nodes [48]. Also, the computing nodes can be presented as a DAG, transforming the problem to a mapping of nodes between two graphs [49].

There are also some works that target the optimization of the analytics job from a different scope. One of them is to avoid moving data that will not participate in the final output of the job [50]. Doing so, we can avoid unnecessary data movement, ensuring lower bandwidth usage. Also, another common technique is using filtering and aggregation techniques [51] or sampling [52] on data to achieve higher throughput. Using these approaches, the volume of data is decreasing. Many works also try to decrease the quality of data in order to find an optimal trade-off between the quality of results and the response time or bandwidth usage [53]. An example of a data quality type this approach can be applied onto is accuracy. This quality degradation must not affect the reliability of the final results so the level at which it is acceptable should be carefully decided.



The focus of these works may be single-objective, where they try to minimize only one metric at a time or multi-objective, where two or more metrics are considered. In the second scenario a single approach is to assign weights to the objectives [31] making the problem a single-objective one. Another approach is to find the optimal trade-offs between the metrics [46]. Finally, some works built more complex cost models including the various metrics.

Some of the most common tools used for geo-distributed analytics include customized flavors of Apache Spark which is used mainly for batch processing, although a streaming extension exists, Apache Storm (which allows for easier DAG creation and set-up) and Apache Flink. However, these solutions, building upon extensions of MapReduce and focusing on massive parallelism, might not be suitable for a fog/edge analytics setting, especially when the computing nodes might be resource-constraint devices. On the other hand, the main effort in fog analytics is still in the area of programming models and appropriate deployment abstractions [54], which prohibit the community to benefit from the big advances in distributed analytics by offering users with the tooling for hiding most of the complexity related to machine scheduling, task coordination, and fault tolerance. Nonetheless, recently a number of frameworks are appearing for deriving analytic insights for edge computing and network telemetry. For example, Edgent (<http://edgent.apache.org/>), Kafka and Sieve [55] are frameworks providing micro-kernel runtimes with small footprints that are particularly tailored to deriving streaming analytics on IoT gateways, network routers and edge devices.

In RAINBOW, we plan to explore current state-of-the-art frameworks for geo-distributed edge analytics and also build upon partner expertise in multi-objective geo-distributed and streaming analytics, e.g. [46] [52] to jointly take into consideration data placement and task analysis to determine which data chunks (e.g., metadata required for orchestration) to move and where to place processing tasks given multiple alternatives.

3.2 State of the Art paradigms

3.2.1 From literature

The modern world is becoming increasingly connected, at an exponential rate. With computing models' evolution having shifted from cloud to fog-based architectures, the demand for the foundation of a sustainable network ecosystem is higher than ever. Thus, was formed the OpenFog Consortium [5]. It was founded on the idea that the future of fog computing is an open fog computing architecture. Through an independent open membership ecosystem of industry, end users and universities, a broad coalition of knowledge can be applied directly to technical and market challenges. OpenFog Reference Architecture (RA) aims for fully interoperable and secure systems, supported by a spectrum of suppliers, as OpenFog functions complementarily to other IoT and technology industry alliance groups alike.



The OpenFog Consortium (which recently merged into the Industrial Internet Consortium) provides a reference architecture, which takes a hierarchical approach with IoT devices being at the bottom and the Cloud being at the top. The fog is made up of the nodes between these two ends. Data processing and storage can occur at any point in the hierarchy. In general, as data flows from the bottom to the top, the volume is reduced. Raw processing occurs mostly at the lower layers of the fog, while resource intensive tasks, like machine learning, are performed towards the top. The OpenFog reference architecture consists of eight pillars that address the most crucial areas of fog computing: security, scalability, openness, autonomy, RAS (reliability, availability, and serviceability), agility, hierarchy, and programmability. Those pillars are essentially a set of core principles on which the entire OpenFog reference architecture is based. They can be characterized as key virtues that a system shall adhere to, in order to claim the OpenFog's definition of an horizontal, system-level architecture.

OpenFog RA is the first step in creating new industry standards to enable interoperability in IoT, 5G, Artificial Intelligence, Tactile Internet, Virtual Reality and other complex data and network intensive applications. It represents an industry commitment toward cooperative, open and inter-operative fog systems to accelerate advanced deployments in smart cities, smart energy, smart transportation, smart healthcare and manufacturing. The OpenFog Consortium considers that without its open architecture, interoperability, reliability and security will be rather limited resulting in slower adoption and limited functionality of the entire system. Therefore, the architecture's open nature is of outmost importance. The OpenFog Reference Architecture is the first step in forming industry standards for fog computing.

3.2.2 From Projects

We are at the beginning of a new technological revolution as disruptive technologies such as cyber-physical systems, machine-to-machine communication, Big Data, AI, and human-machine collaboration aim to transform crucial industries such as energy, manufacturing, agriculture and livestock, industrial automation, retail, etc. . However, the aforementioned domains will reach their true potentials only through the convergence of Operational and Information Technologies (OT & IT). The European Parliament claims that "[...] this convergence will be achieved through the new concept of Fog Computing, which is a logical extension from Cloud Computing towards the edge of the network (where machines are located), enabling applications that demand guarantees in safety, security, and real-time behaviour.". That is the reason why European Commission has allocated significant amount of resources towards this direction. The projects funded in the same call with are listed below:

3.2.2.1 *Accordion*

ACCORDION project [56] aims to establish an opportunistic approach in bringing together edge resource/infrastructures (public clouds, on-premise infrastructures, telco resources, even end-devices) in pools defined in terms of latency, that can support NextGen application requirements. To mitigate the expectation that these pools will be "sparse", providing low availability guarantees, ACCORDION will intelligently orchestrate the compute & network continuum formed between edge and public clouds, using the



latter as a capacitor. Deployment decisions will be taken also based on privacy, security, cost, time and resource type criteria. The slow adoption rate of novel technological concepts from the EU SMEs will be tackled through an application framework, that will leverage DevOps and SecOps to facilitate the transition to the ACCORDION system. With a strong emphasis on European edge computing efforts (MEC, OSM) and 3 highly anticipated NextGen applications on collaborative VR, multiplayer mobile- and cloud-gaming, brought by the involved end users, ACCORDION is expecting to radically impact the application development and deployment landscape, also directing part of the related revenue from non-EU vendors to EU-local infrastructure and application providers.

3.2.2.2 hCloud

H-CLOUD project [57] aims to drive coordination and support activities for the consolidation and growth of the Cloud Computing research and innovation community in Europe, by bringing together innovators, policy makers, cloud computing research, industry and users into an open, participatory and sustainable forum. H-CLOUD will provide a rich set of collaborative content, tools and actions to overcome fragmentation and increase collaboration in Europe and beyond, while aligning on a common direction to help creating a Cloud agenda for the future of Europe. To this end H-CLOUD will lead the definition of the Strategic Innovation and Research Agenda for Cloud Computing that will provide recommendations and strategies to guide the future of European Cloud services and their market regulations. This will encompass the creation of a comprehensive knowledge base, the H-CLOUD LANDSCAPE, including an online catalogue of stakeholders, initiatives, projects, businesses, policies, success stories and best practices that will be made accessible to all H-CLOUD FORUM participants.

3.2.2.3 MORPHEMIC

MORPHEMIC project [58] proposes a unique way of adapting and optimizing Cloud computing applications by introducing the novel concepts of polymorph architecture and proactive adaptation. The MORPHEMIC deployment platform will be very beneficial for heterogeneous deployment in distributed environments combining various Cloud levels including Cloud data centres, edge Clouds, 5G base stations, and fog devices. The project outcomes will be implemented in the form of a complete solution, starting from modelling, through profiling, optimization, runtime reconfiguration and monitoring. Then the MORPHEMIC implementation will be integrated as a pre-processor for the existing MELODIC platform [59] extending its deployment and adaptation capabilities beyond the multi-cloud and cross-cloud to the edge, 5G, and fog.

3.2.2.4 Pledger

Pledger project [60] aims to provide a new architectural model as well as a set of software tools that will prepare the future development of the next generation of edge computing. The project will allow edge computing providers to secure the stability and effective performance of the edge infrastructures. It will also allow edge computing users to understand the nature of their applications, research understandable quality of service



metrics and optimise the competitiveness of their infrastructures. The project intends to introduce the set of tools in the application fields of manufacturing, mixed reality and smart cities.

3.2.2.5 *FogProtect*

FogProtect project [61] aims to deliver new and advanced architectures, technologies, and methodologies for ensuring end-to-end data protection across the computing continuum, from cloud data centres through fog nodes to end devices. The FogProtect solutions are generic and can be used in multiple contexts to support many types of applications and services. FogProtect combines four main technology innovations: (i) secure data container technology for data portability and mobility, (ii) data-protection-aware adaptive service and resource management, (iii) advanced data protection policy management, (iv) dynamic data protection risk management models and tools.

3.2.2.6 *SmartCLIDE*

SmartCLIDE project [62] aims to develop a radically new smart cloud-native development environment, based on the coding-by-demonstration principle, that will support creators of cloud services in the discovery, creation, composition, testing and deployment of full-stack data-centred services and applications in the cloud.

There is also a significant number of older related projects that have been funded.

3.2.2.7 *LighKone*

LighKone project (Lightweight Communication for networks at the edge) [63] aims to develop a validated model for doing general-purpose computation on edge networks. During the project, LighKone consortium presented an edge reference architecture that exploits decentralized lateral data sharing and convergent vertical data semantics across a myriad of different edge resources. Project results will be new programming models and algorithms that advance scientific understanding, implemented in new industrial applications and a startup company, and evaluated in large-scale realistic settings. LighKone offers Application Deployment as well as Intelligent Orchestration and Fog Service Placement, but lacks services such as Adaptive Monitoring, Elastic IoT Applications, Security Services, Trust Enablement, Analytics Services and Data Management, all of which are offered by RAINBOW.

3.2.2.8 *mF2C*

mF2C project [64] aims to design and develop an open, secure, decentralized, multi-stakeholder management framework. It envisages providing innovative programming models, privacy and security, data storage techniques, service creation, brokerage solutions, SLA policies and resource orchestration methods and tools. mF2C offers a variety of tools and services such as extended-GUI, user and service management, resource management framework, telemetry monitoring solution, service orchestrator and more. The services provided by mF2C are identical to the ones provided by LighKone, with the addition of Data Management.

3.2.2.9 *RECAP*

RECAP project [65] aims to develop an innovative concept for optimized provision of cloud services. These services are elastically instantiated and provisioned “on the field”,



near to users that actually need them, via self-configurable cloud computing systems. The project can be subdivided into 4 technological axes: Workload annotation, near real time decision support system for heterogeneous distributed clouds, efficient provisioning of resources and QoS aware orchestration. RECAP offers Application Deployment, Intelligent Orchestration, Adaptive Monitoring and Analytics Services, but lacks Security Services, Elastic IoT Applications, Fog Service Placement, Trust Enablement and Data Management.

3.2.2.10 PrEstoCloud

PrEstoCloud project [66] aims to contribute in the Cloud computing and real-time Big Data technologies. PrEstoCloud ultimate objective is to provide a highly distributed architecture for proactive cloud resources management reaching the extreme edge of the network. The project envisages supporting services regarding real-time big data processing which are going to be deployed and validated in various challenging use cases. Project's technical objectives include network virtualization, dynamic monitoring in real-time processing, real-time mobile stream processing, as well as pro-active cloud computing. PrEstoCloud offers Intelligent Orchestration, Elastic IoT Applications and Data Management.

3.2.2.11 Ditas

DITAS project [67] aims to develop a cloud platform which is going to help developers design data-intensive applications by specifying Virtual Data Containers and constraints/preferences, run them on a mixed cloud/edge environment and execute the distributed application, no matter the total number of different devices, their technical characteristics and the heterogeneity of runtime environments. Moreover, DITAS offers a monitoring system capable of collecting and analysing data related to the application, such as execution status, data movements. Project Ditas enables Application Deployment, Intelligent Orchestration of said project, Fog Service Placement as well as Data Management.

3.2.2.12 Fog Guru

Fog Guru project [68] aims to familiarize software engineers and cloud specialists with the forthcoming Fog Computing technologies. The next generation of fog specialists will be in position to establish and maintain reliable fog networks and applications focused at real-time low-latency interactive services, as well as high-throughput and dependable fog applications. Fog Guru aims to fill the existing gap of research in the fields of resource management for scalable fog networks, adaptation of software to new fog platforms and the development of a solid foundation for future fog-centred software. Fog Guru offers Application Deployment, Intelligent Orchestration, Adaptive Monitoring services, as well as Elastic IoT Applications.

3.2.2.13 Arrow Head

The Arrow Head project [69] aims at increasing global industrial efficiency, adaptability and resilience. The project focuses on energy (production and management) physical infrastructure and electrical vehicles. Arrow Head provides a solid technical foundation for Cooperative automations and IoT applications. The project's focus and goal are a



modern, efficient energy industry, smart cities and intelligently engineered infrastructure. Project Arrow Head allows Application Deployment, Intelligent Orchestration and Fog Service Placement while offering Security Services and Trust Enablement.

3.2.2.14 Unicorn

Unicorn project [70] aims to assist the design and deployment of cloud services and applications. Unicorn aids developers in increasing their productivity by minimizing time required to code and design cloud applications. This alleviation of effort is achieved via the creation of security blueprints, standardizations and code annotations. Cloud service providers shall gain the ability to manage cloud services lifecycle while using a framework established on and build with respect to end-user privacy and data protection. Project Unicorn enables Application Deployment, Intelligent Orchestration, Adaptive Monitoring services, Elastic IoT Applications and offers Security Services.

3.2.2.15 Decenter

Decenter project [71] is a research/innovation project which aims to establish a robust fog computing platform, serving the entire spectrum of the interface between the cloud and physical networked devices (“Things”). Decenter shall provide application-aware resource allocation and management, powered by Artificial Intelligence. The project’s outcomes find applications in various sectors, such as Robotics and logistics, smart cities and the construction industry. Decenter Project offers Application Deployment, Intelligent Orchestration, Elastic IoT Applications, Fog Service Placement, Security Services as well as Data Management, thus falling short compared to RAINBOW only in regard to Adaptive Monitoring, Trust Enablement and Analytics Services.

3.2.3 From Industry

It goes without saying that industrial stakeholders have also allocated a significant amount of resources towards this direction. Some popular solutions are listed below:

3.2.3.1 AWS IoT Greengrass

AWS IoT Greengrass [72] is a platform aimed at extending the AWS platform to edge devices. With said extension, edge devices can act autonomously and thus base decisions on the data they generate, while maintaining connection to cloud for management, analytics, and durable storage. AWS IoT Greengrass allows connected devices to execute commands based on predictions allowed for by machine learning. The platform allows device data to remain in sync. Furthermore, edge-devices can securely communicate with other devices, with no internet connection being necessary.

3.2.3.2 MindSphere

MindSphere [73] by Siemens is an IoT operating system and cloud platform. This system is used in numerous applications such as: automated production and vehicle fleet management. MindSphere functions by collecting, analysing and locally storing various sensor data in real time. Said data is made it accessible to end-users via an open platform, which can then be utilized for both software and hardware automations, management as well as optimizations.



3.2.3.3 *ioFog*

The ioFog platform [74] is a utility aimed at establishing edge-computing networks. ioFog is open source and requires minimal computing power, enabling it to run on machines with limited available computing power. After successful establishment of an edge-computing network, a user can use it as a basis for any microservice automation.

3.2.3.4 *Vapor*

Vapor [75] is a platform aimed at decentralizing internet services and establishing a massive, “city-scale” IoT network. Vapor’s SDN-based platform removes unnecessary latency barriers standing in the way of true edge-computing, thus shaping the next generation of networking infrastructure, where computing, storage and data acquisition is effectively and efficiently decentralized, void of any restrictive latency. Data, application and end-users can thus be brought closer than ever.

3.2.3.5 *Crosser*

Crosser [76] provides a platform, aimed at bringing about the era of edge computing and a transformation to an IoT-enabled industry. Crosser enables real-time processing of streaming or batch data for Industrial IoT, analytics and various automations. The platform is built with predictive maintenance and condition monitoring in mind, as it promises seamless integration and interconnection of industrial machines, sensors and equipment. The platform supports real-time integration and utilization of harvested data and allows for further automation of the industry.

4 Stakeholders and their Goals

This chapter provides a comprehensive description of the RAINBOW platform stakeholder analysis. Stakeholders are divided into three categories which are based on their role in edge/fog ecosystem and their interests and relations are further investigated, in order to identify RAINBOW's key stakeholders. Moreover, the applications and markets that may benefit from RAINBOW outcomes are classified into four categories and are further explored using stakeholder interviews.

4.1 The 3 Categories of Stakeholders

The multiple benefits that edge/fog computing offers are broadening the range of potential stakeholders. These stakeholders can benefit from this technology either as individual users or as organizations from three different positions: (i) use services deployed at the edge, (ii) develop services deployed at the edge or (iii) design and operate edge infrastructure [77]. These positions form the three categories of stakeholders:

- The first category includes the end users of a service. They could be individuals or groups of users who are interested in using edge/fog technology to enhance their approaches, to solve a problem in their market, to outperform competition etc.
- The second category includes the service developers/providers. They could be individuals or companies that possess the technical knowledge to develop and support edge/fog applications. The main motivation for this layer of stakeholders, is to expand their expertise and operations to new markets using state of the art tools and technologies, having revenue as their final goal.
- The third category includes providers of edge/fog computing infrastructure. This category of stakeholders includes large companies operating in Information Technology market, who produce devices throughout the (IT) ecosystem (e.g. network devices, IoT devices, standalone micro-processors etc.) or who provide their infrastructure for communication establishment (e.g. Internet Service Providers, operators of cellular networks, etc.).

Figure 5 illustrates the 3 categories of stakeholders in edge/fog computing.

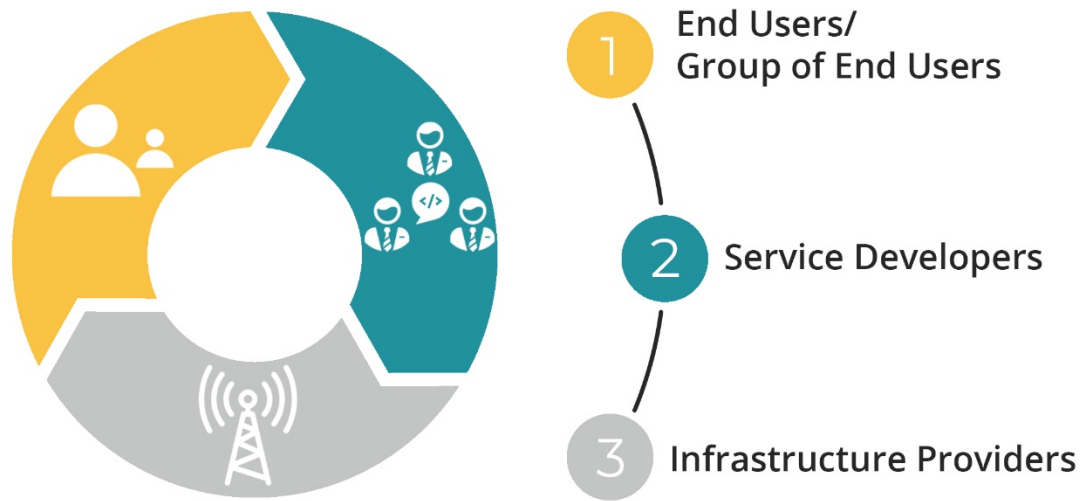


Figure 5: Stakeholder Categories

4.1.1 Interests & Relations between Stakeholder Categories

In order to provide an accurate and comprehensive reflection of the RAINBOW stakeholder landscape, all the interests, dependencies and relations have to be elucidated.

4.1.1.1 Users – Service Developers

Users have certain expectations on the quality of service delivered by the application they use. In order to meet the expectations of end users, service developers have to be at the forefront of edge/fog technology by expanding their knowledge, gaining experience and extending their development toolkit. Moreover, service developers have to establish new strategies that will help them to identify the forthcoming user needs and provide an effective training strategy for the utilization of their products.

4.1.1.2 Users – Infrastructure Providers

If one user expectation shall be taken for granted, this is the high availability of a service. Every user wishes to enjoy a smooth and uninterrupted experience when he uses an application or a service. Infrastructure providers are responsible for ensuring service availability under most of the circumstances. Edge/fog computing can be seen as a great opportunity for infrastructure providers, in order to generate additional revenue. This for example can be achieved by either offering resources at the access network or renting out space for server colocation at those access gateways.

4.1.1.3 Service Developer – Infrastructure Providers

Service developers are closely related to the infrastructure providers since usually they build on top of their infrastructures. What they expect to receive from infrastructure providers are products of high quality along with rich documentation and technical support. On the other hand, infrastructure providers are expecting the feedback of service developers regarding testing and reporting of the underlying product.

Figure 6 summarizes the different stakeholders in edge/fog computing and their respective interests.

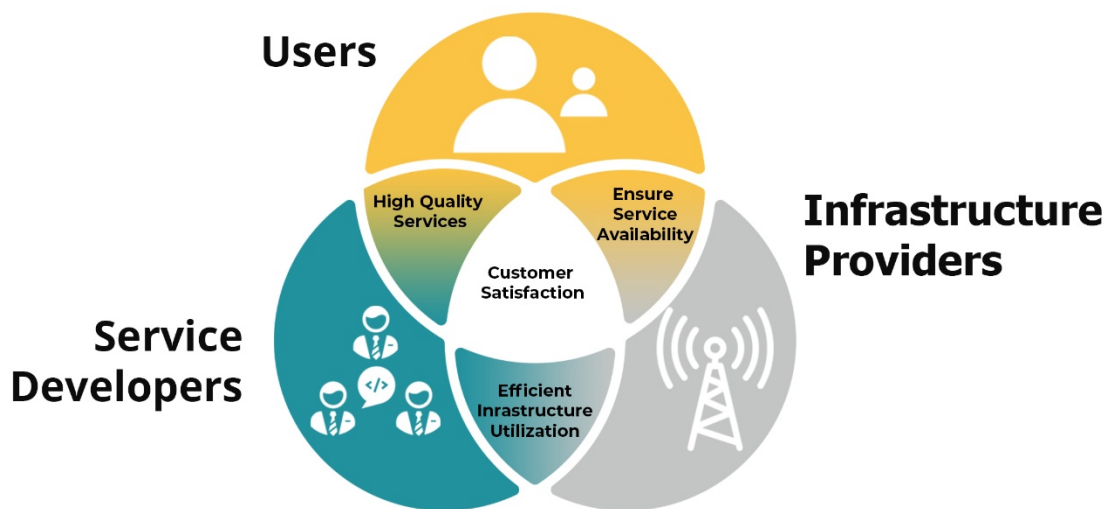


Figure 6: Stakeholders and their respective interests

4.1.2 Interest vs Power Matrix

The process of identifying stakeholders usually results in a long list of individuals and groups. The Interest versus Power Matrix is a structured and comprehensive method to illustrate the classification of each stakeholder into one of the four following categories.

- High Influence & High Interest: Stakeholders that have considerable power to influence the specific market and consequently the project's development and impact and on the other hand, high Interest in the project's outcomes.
- High Influence & Less Interest: Stakeholders that have considerable power to influence the specific market and consequently the project's development and impact, but they are not interested in project's development and outcomes.
- Less Influence & High Interest: Stakeholders that usually follow the market trends and do not have the power to affect it, but they are interested in new tools, such as RAINBOW platform, that will make them more productive and competitive.
- Low Influence & Low Interest: Stakeholders that have neither the power to influence the project's development nor the interest to exploit any of the project's outcomes.



In order to identify the power of each stakeholder a literature review was conducted, and the following questions have been taken into consideration to identify the stakeholders' interests:

- What do stakeholders expect from the project and how do they benefit?
- Are there any conflicting interests that the stakeholder may have with the project?
- How committed is the stakeholder to the project? Is he/she willing to commit tangible resources?
- Are there relationship conflicts between stakeholders that can hinder the project?

In order to have a thorough Interest versus Power analysis, the Interest versus Power methodology was exploited as a recursive process for the RAINBOW project and for the layer of stakeholders that RAINBOW project is most interested in. In other words, the Interest versus Power approach has been evaluated under 2 different perspectives. The first, is the RAINBOW perspective while the second, is the perspective of the stakeholder categories that RAINBOW project is most interested in.

Under the RAINBOW perspective, service developers are the layer of stakeholders that RAINBOW is most interested in. The more service developers use RAINBOW platform to develop new applications the higher the value of the platform. Of course, value can be interpreted in various ways. Infrastructure providers have considerable power to influence the respective ecosystem since they provide all the infrastructure for facilitating the deployment of edge/fog technology. However, infrastructure providers are not directly affected by concepts like RAINBOW. They do not adapt neither their policies nor their products based on an emerging software development platform. Although, they tend to monitor the software development ecosystem especially when a platform becomes very popular. On the contrary, software development platforms such as RAINBOW, have to investigate the infrastructure providers' policies and hardware in order to be more competitive and viable in the long term. As regards the end users, they do not have neither the knowledge to exploit the platform's benefits, nor the power to contribute to the platform's market penetration.

Under the perspective of service developers, users are very important. It goes without saying that the more users use a service the higher the revenue for the service developers. As a result, besides the service developer layer, user is also a key stakeholder for RAINBOW, since user's needs are the those that will trigger the interest of service developers.

Figure 7 illustrates the Interest versus Power Matrix under the two different perspectives. In the first case, under the RAINBOW perspective, service developers are the type of stakeholders that most influences RAINBOW platform prosperity, while in the second case under the service developers' perspective, users is the type of stakeholders that most influences the service developers success and consequently the RAINBOW platform prosperity as well.

Interest vs Power (RAINBOW PERSPECTIVE)

Interest vs Power (SERVICE DEVs PERSPECTIVE)

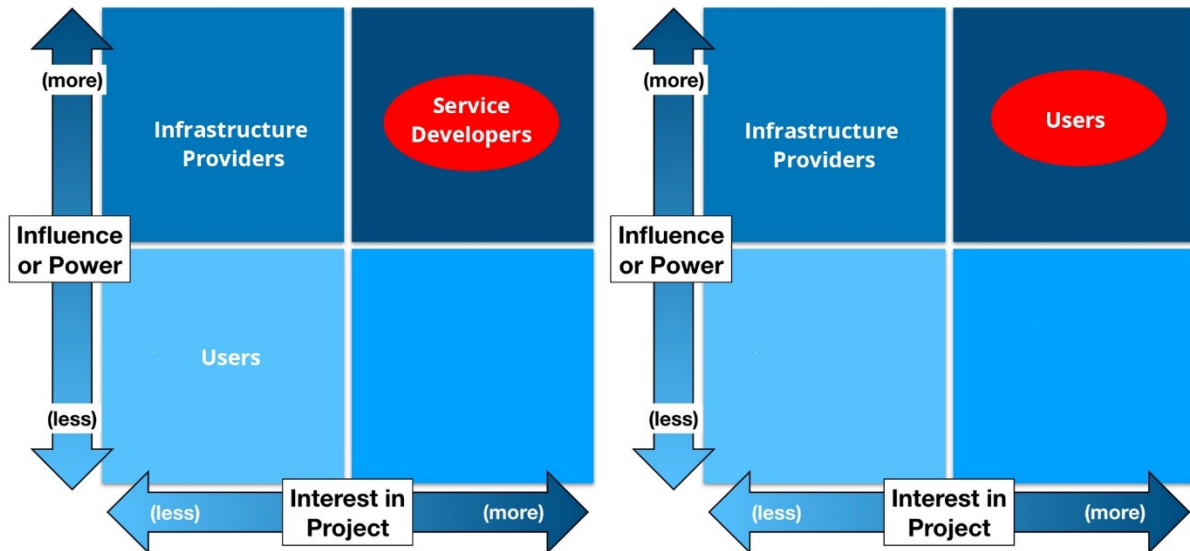


Figure 7: Interest vs Power Matrix for the 3 Categories of Stakeholders

The Interest versus Power approach will also help to establish the proper engagement strategy for each type of stakeholders, an action which will further scrutinized in WP 7. Table 3 classifies the different stakeholder types with associated strategies for engagement.

Level of Importance	Category & classification	Strategy to maximize their engagement
Primary Stakeholders	Key players: High Influence & High Interest	<ul style="list-style-type: none"> – Key players focus effort on this group – Engage and consult regularly – Involve in governance
	Meet their needs: High Influence & Less Interest	<ul style="list-style-type: none"> – Engage and consult in their interest area – Try to increase level of interest – Aim to move into key players
Secondary Stakeholders	Show consideration: Less Influence & High Interest	<ul style="list-style-type: none"> – Make use of interest through involvement in low risk areas – Keep informed and consult on interest area – Potential supporter
	Least important: Low Influence & Low Interest	<ul style="list-style-type: none"> – Inform via general communications: Newsletter, website, etc. – Aim to move into group 3

Table 3: Classification of different stakeholder types with associated strategies for engagement



4.2 Categories of Applications

4.2.1 Mobile Devices Applications and Gaming

The applications which fall into this category are specifically targeted at consumer's mobile devices, such as smartphones, tablets or head-mounted devices. Mobile applications and gaming development are rapidly expanding and they are in a stage when they seek to provide an experience more immersive than ever before, by utilizing new devices such as headsets, smart glasses etc. and state of the art technologies such as Augmented Reality (AR) and Virtual Reality (VR) [78]. Although some of the aforementioned concepts are already in the market, their exploitation and the market acceptance are under consideration since there are not many devices which can support them, because they are really demanding in terms of processing and storage resources. These requirements have led to the emergence of new business and deployment models such as Gaming as a Service (GaaS) [79]. GaaS is a concept that overcomes hardware limitations through application modularization where a demanding functionality is migrated from the mobile device to a server, a concept that resembles the concept of Edge/fog technology. Edge/fog computing can take up the aforementioned challenges and thus it is expected to give a significant push to mobile applications and gaming industry.

4.2.2 Infrastructure Applications

The notion of infrastructure refers to infrastructure as basic services and facilities which the well-being of society depends on. Smart Grids [80], environmental monitoring [81], waste management [82], public safety and emergency response [83], smart transportation [84] and connected cars [85] are huge concepts that involve plenty of requirements such as scalability, real time processing, guaranteed QoS, trust etc. . Edge/fog seems a promising option which can take up the aforementioned challenges. As a result, it can be considered as a key technology in the deployment of concepts around smart cities [86] by facilitating information technology to augment critical infrastructures.

4.2.3 IoT Device Applications

Internet of Things (IoT) refers to objects that are connected and able to interact with each other and extend the Internet to the physical world [87]. IoT is a tremendous concept which can potentially cover every aspect of the human's daily activity. Out of all IoT applications, smart building [88], smart agriculture and livestock [89] and industrial IoT [90] are three applications that are already utilized for enhancing efficiency, productivity, and resource saving. The data volume produced by these applications and the latency requirements they may subtend in some cases, will be likely to be critical in terms of transfer and processing at central clouds [91], [92]. Edge/fog computing can handle the delay sensitive tasks and some of the data volume in order to support such kind of applications.

4.2.4 Human Applications

Human applications include all these applications which can be used to improve the well-being and capabilities of humans. Real time monitoring of human's vital parameters [93] and precision medicine [94] are two types of the human-centric application that are already in the market and fall into the category of the connected health concept. Connected health is a sociotechnical model for healthcare management and delivery by using technology to provide healthcare services remotely which aims to maximize healthcare resources and provide increased, flexible opportunities for individuals to engage with clinicians and better self-manage their care [95]. Moreover, it brings together multidisciplinary technologies to provide preventive or remote treatments by utilizing digital health information structure while at the same time connecting patients and caregivers seamlessly in the loop of the healthcare ecosystem.

Privacy and data security are critical concerns in such kind of applications due to the intimate nature of the data. Today's cloud-based services fail to take up these requirements [96] which implies the need for new technologies such as edge/fog which can deal with issues such as data integrity, authenticity, and confidentiality.

Figure 8 sketches the categories of applications that are expected to capitalize the benefits of the edge/fog technology.



Figure 8: Categories of Applications

4.3 Stakeholder Roles

RAINBOW platform is a complex system which potentially engages multiple stakeholder roles to fully exploit all its components and make it altogether available as a turn-key solution.

Based on the technical analysis of the platform, the following paragraphs introduce the identified user roles for the RAINBOW ecosystem. From this table, we observe that the RAINBOW ecosystem involves four technical roles with diverse responsibilities. Some of these responsibilities may overlap among users of the platform which, at first, may seem



to lead to confusing interpretation of user role duties. However, usually for small software teams, the silver lining between roles in the development team are quite blur, with team members often taking responsibilities spread across different user roles (e.g., service developer and operator). In the following, the Actor terminology and descriptions are designed to clarify and summarize each actor's roles.

4.3.1 Service Developer

The **Service Developer** is a person (or a team) that develops a Fog application or a part of it by using the RAINBOW API, RAINBOW-compliant libraries, deployment description and primitives in order to run on a RAINBOW-compliant Fog execution environment. This role also includes responsibilities such as, unit testing, software features updates and maintenance of fog application's services.

4.3.2 Service Operator

The **Service Operator** (aka Platform Operator) is the person providing the vision for multiple fog applications as a project, gathering and prioritizing user requirements and overseeing the business aspects of deployed applications (e.g. business delivery, functioning and services of the application) in accordance with various criteria (e.g. cost minimization and policy definition like legal constraints). This role is also responsible for the supervision of any hardware failure (e.g., network or fog node fails) and takes the appropriate actions to notify the respective Infrastructure Provider.

4.3.3 Infrastructure Provider

The **Infrastructure (Cloud and Fog) Provider** is the organization that provides Fog services in the form of programmable infrastructure according to a service-level agreement. The Fog Provider is also responsible to operate the Execution Environments in proximity to the end-users and/or IoT devices that will host entirely or partially RAINBOW-compliant Fog Applications.

4.3.4 RAINBOW Developer

The **RAINBOW Developer** is the person (or a team) that creates RAINBOW related (software) components for compliant Fog and Cloud Providers and/or Engineers such as e.g. Monitoring Probes, Modelling Primitives, services utilizing the RAINBOW API.



5 RAINBOW Questionnaire and Stakeholder Interviews

This chapter presents the rationale behind the questionnaire which has been forwarded to the key stakeholders who have been identified in the previous chapter.

5.1 Questionnaire Establishment

RAINBOW partners created a questionnaire to assist in the capture of functional and non-functional requirements from European digital SMEs and start-ups. The highest priority was to make the questionnaire usable and understandable by both technical and non-technical stakeholders. That is the reason why the questionnaire consists of two flows in accordance with the role of the respondent, one flow for the non-technical respondents and one flow for the technical respondents.

The questionnaire comprises 8 different sections and depending on the respondent's answers, some of them will emerge. More specifically, based upon the answer regarding his/her role in "Introductory Questions" section, the respondent will answer either "Business Questions" (section 3) or the technical questions included in sections 4, 5, 6, 7, 8.

- 1 - Edge/fog Computing
- 2 - Introductory Questions
- 3 - Business Questions
- 4 - Application Deployment Modelling
- 5 - Orchestration
- 6 - Data Management and Analytics
- 7 - Secure Remote Asset Management
- 8 - System Configuration Integrity

The estimated completion time of the questionnaire is 15 minutes.

5.1.1 Business Perspective

The business section of the questionnaire aims to elicit why the stakeholder needs the RAINBOW platform, what has happened/is happening inside their organization that may trigger this need or his/her interest, what are the expected business benefits to be delivered by or the challenges/problems to be solved by adopting RAINBOW solution.

5.1.2 Technical Perspective

The technical perspective section aims to elicit the functional, performance, security and other technical system requirements. This section is further divided into five subsections where in each sub-section a different type of requirement is addressed. The selection of the type of requirement was made in accordance with "RAINBOW's Value Propositions".



5.1.2.1 Value Proposition 1 - Cloud-service Modelling

This value proposition is reflected in questionnaire's section 4 - "Application Deployment Modelling". Application deployment modeling is a formal way for describing an application topology. Within the topology description users can denote the intercommunication and dependencies among application services and annotate their description with constraints, optimization policies and QoS requirements.

5.1.2.2 Value Proposition 2 - Orchestration Algorithms

This value proposition is reflected in questionnaire's section 5 - "Orchestration". The orchestrator will be responsible for selecting the nodes on which a deployed application will run, based on the requirements specified in the service graph. Furthermore, it will monitor the running applications and make necessary adjustments to meet the specified QoS goals.

5.1.2.3 Value Proposition 3 - Efficient Data Storage, Querying and Processing

This value proposition is reflected in questionnaire's section 6 - "Data Management and Analytics". Analytics services help businesses convert their historical and real-time data into actionable insights for data-driven decisions. In general, they consist of services for ingesting, storing, processing, and accessing the data created by applications, devices and users.

5.1.2.4 Value Proposition 4 - Secure Zero-touch configuration

This value proposition is reflected in questionnaire's section 7 - "Secure Remote Asset Management". Secure remote asset management refers to the deployment of sufficient services for the real-time supervision, monitoring, and management related to the security of the information and functionalities about the deployed assets; being edge devices, fog nodes or service graphs deployed in the cloud. Access to machines and remote edge devices, edge addition (and/or removal), secure re-programming and/or reconfiguration, etc. must be managed to protect the security posture of the provided services and the underlying infrastructure.

5.1.2.5 Value Proposition 5 - Configuration Integrity Verification

This value proposition is reflected in questionnaire's section 8 - "System Configuration Integrity". Configuration Integrity Verification refers to mechanisms and tools that enable to assess and preserve the integrity of the deployed applications and services, during both deployment and run-time, so that they can assess the trusted state of the host devices.

5.2 Questionnaire Recipients

Generally, a substantial bottleneck of conducting research is finding people who will take part in the research and contribute with their knowledge and experience by filling in a questionnaire or participating in an interview. Since RAINBOW consortium is aware of this fact, it proactively initiated an effort to device a list of digital SMEs, start-ups and research institutes in the domain which interests RAINBOW. Each partner explored and examined its contacts and collaborations, in order to identify potential stakeholders that



(i) may be interested in RAINBOW outcomes and (ii) have the pertinent “know-how” and populated the list. Through this process, RAINBOW expects to receive a high number of responses that will feed the process of eliciting functional and non-functional requirements.

RAINBOW demonstrators play an important role in functional and non-functional requirements elicitation since not only did they helped with the creation of the questionnaire, but they were also among the first responders.

5.3 Stakeholder Interviews

Following the theoretical stakeholder analysis, RAINBOW consortium has explored its network of contacts in order to initiate a discussion with some contacts of these, so as to identify their needs and challenges, as well as the benefits they expect to obtain. This has been accomplished by one-to-one teleconferences that took place at the beginning of March 2020. RAINBOW consortium contacted eleven potential end users and by using an open discussion approach, retrieved interesting insights and gained a deeper understanding of each market. The findings of each discussion were processed and then documented in Table 4 which summarizes the relevance, needs and benefits for each potential end user.



Application Category	Actor	Relevance	Needs and challenges	Benefits
IoT Device Applications	Drone service Providers	Actors that offer a wide range of services using drones. The services may include area surveillance, smart agriculture, track and trace missions etc.	<ul style="list-style-type: none"> – Real time data processing on the drone – Enable collaboration between drones – Terrain overlapping – Routing Alteration with respect to mission parameters 	<ul style="list-style-type: none"> – Reactive validation of mission execution – Increase mission execution efficiency
	Industries and Factories	Actors where employees co-exist with cyber-physical systems and machinery (CNC robots, lathes, wreckers etc.) and human safety is of high importance	<ul style="list-style-type: none"> – High delays in system response – Real time processing – Alleviate the load from a central node – High maintenance costs in terms of human and financial resources – Compliance with the EU regulations – Lack of knowledge 	<ul style="list-style-type: none"> – Guaranteed QoS (rapid in terms of response time and accurate in terms of decision making)
	Agriculture & Livestock	Big players in agriculture domain who participate in large agriculture ecosystems which produce high volume of data (such as Mixed Farming Systems)	<ul style="list-style-type: none"> – Vast area of land – Large volume of multi-collective, heterogeneous data coming from crops, livestock, forestry etc. – Data privacy – Complex concepts of data processing and visualization 	<ul style="list-style-type: none"> – Adopt technology as a mean for sustainable models that improve the efficiency of their ecosystem, while providing a secure and privacy-preserving IoT ecosystem – Formulate rules and quantified metrics for optimum conditions in terms of (animal) behaviour, physiology, food quality, nutrition and agriculture environment – Leverage SoTA technologies such as augmented reality or digital twins



Project No 871403 (RAINBOW)

D1.1 – RAINBOW Stakeholders Requirements Analysis

Date: 30.06.2020

Dissemination Level: PU

Application Category	Actor	Relevance	Needs and challenges	Benefits
IoT Device Applications	Supply Chain (especially for short life span products such as dairy products)	Actors that are involved in the supply chain value either as producers or as intermediaries	<ul style="list-style-type: none"> – Complex concepts of data processing in near real time – Large volume of data 	<ul style="list-style-type: none"> – Avoid over- storage which leads to destruction or disposal at low prices or failure covering potential sales – Perform predictive analytics using production and sales data
Infrastructure Applications	Urban Mobility Service Providers	Actors that offer a wide range of services that leverage various data in order to improve urban mobility.	<ul style="list-style-type: none"> – Huge volume of multi-collective, heterogeneous data coming from (Geo-reference information of high-speed mobile nodes, crowdsourcing reports etc.) – Processing and distributing real-time information – Low latency for disseminating information – Trustworthy crowdsourcing data 	<ul style="list-style-type: none"> – Make the huge volume of data manageable – Alleviate load from centralized nodes (such as traffic control centers) – Guaranteed QoS (availability, accuracy, integrity)
	Smart Building	Actors that focus on smart energy optimisation of buildings	<ul style="list-style-type: none"> – Large volume of multi-collective, heterogeneous data (temperature, humidity, lighting levels, weather conditions, human presence, time of the day, day of the week, electricity/oil/gas price) 	<ul style="list-style-type: none"> – Perform predictive analytics which enable automated heating/cooling/lighting control, as well as more complex concepts such as load balancing including energy harvesting (solar panels)
	Maintenance Operations	Actors that provide maintenance services in	<ul style="list-style-type: none"> – Huge volume of data coming from video streaming 	<ul style="list-style-type: none"> – Improve the maintaining process with distant examination



Project No 871403 (RAINBOW)

D1.1 – RAINBOW Stakeholders Requirements Analysis

Date: 30.06.2020

Dissemination Level: PU

Application Category	Actor	Relevance	Needs and challenges	Benefits
		large systems that can directly affect society (energy and water providers) or smaller systems with high complexity that can threat people's lives (elevators)	<ul style="list-style-type: none"> – Real time data processing – Real time data visualization 	<ul style="list-style-type: none"> – Perform predictive analytics which enable future failures prediction. – Immersive staff training – Reduce operational costs
Infrastructure Applications	Distribution System Operators (DSO)	DSO monitors the medium and low voltage distribution grid, including substations and smart meters used for customer billing.	<ul style="list-style-type: none"> – Huge volume of data generated by their infrastructure – Real time data processing – Real time data visualization 	<ul style="list-style-type: none"> – Provision to avoid failures in the power grid – Predictive maintenance of the infrastructure – Power theft detection
	Transmission System Operators (TSO)	TSO monitors the high and extra-high transmission grid, including substations and the National Energy Management System that oversees the status of the national grid.	<ul style="list-style-type: none"> – Huge volume of data generated by their infrastructure – Real time data processing – Real time data visualization 	<ul style="list-style-type: none"> – Provision to avoid failures in the power grid – Predictive maintenance of the infrastructure
Human Applications	Healthcare industry	Actors that design and develop proactive personalised treatment methodologies	<ul style="list-style-type: none"> – Heterogeneous real-time (from pervasive sensing devices) as well as past data (medical history data based on medical devices inside the hospital) – Data privacy and integrity 	<ul style="list-style-type: none"> – Enabling faster and more efficient health assessment based on inter-domain knowledge exchange – Guaranteed QoS (data privacy, data integrity)



Project No 871403 (RAINBOW)

D1.1 – RAINBOW Stakeholders Requirements Analysis

Date: 30.06.2020

Dissemination Level: PU

Application Category	Actor	Relevance	Needs and challenges	Benefits
Mobile Devices Applications	AR/VR applications, educational applications & Virtual Assistants development	Actors that develop state of the art applications in different mobile platforms such as smartphones and wearables.	– Hardware resource limitation (processing and battery)	– Support resource intensive tasks (video rendering)

Table 4: Relevance, needs and benefits per application category

5.4 Early Results - Discussion

The stakeholder interviews as well as the first findings derived from the questionnaires, especially for the respondents whose large-scale applications have an increased complexity (multiple nodes, heterogeneous data, etc.) showed that describing an application is not a straightforward task. In order to provide a mechanism for the efficient enforcement of service-relevant operations in edge/fog environment, such as application topology, intercommunications and dependencies among services and components, service provision and service management, a completed and precise service specification model is required. Moreover, **constraints, optimization policies and QoS requirements** should also be included in the service specification model as an added competence, since there are a lot of applications (human safety, connected cars, smart grids) that have to comply to specific policies or quality metrics.

Throughout the process of collecting user requirements from the potential stakeholders and in the light of their responses, one thing was ever-present, data. Considering the insights, we aggregated so far, getting access to relevant data, either application behaviour data or performance monitoring data, is the fundamental starting point to enable the respondent's use-cases and meet their business objectives.

Right now, while stakeholders consume plenty of their resources in application behaviour data analysis, they are still invoking pains such as processing delays, data heterogeneity, node mobility, etc. . What they expect is an integrated analytics mechanism that allows data produced by heterogeneous data sources like devices, machines, equipment, pre-processed in real-time closer to where it is created, eliminating delays and preserve the required QoS. There are several technical and business drivers and benefits expected from an edge/fog analytics mechanism, including:

- **Cost Savings:** Significant data reduction by pre-processing and removing dirty and irrelevant data.
- **Increase Data Volume:** Collect more data points from different types of devices. Incorporate more sophisticated data from intelligent devices such as UAVs, connected cars, wearable, etc.) without increasing the cost of acquisition of processing.
- **Transform Raw Data to Insights:** Collect raw data, transform it, analyze it and act on the insights in real-time.
- **Local Intelligence and Automation:** Run local triggers between machines or PLC's with ultra-low latency.

The convergence of OT and IT is another point stressed by the stakeholders. The envisioned RAINBOW platform should allow for both horizontal integration between machines and vertical integration between layers in the OT/IT technology stack. A prerequisite for this integration is that it can be enabled by complementing the existing architecture.

Bring, manage and deploy Machine Learning (ML) models is a requirement that also popped up during the interview sessions with the RAINBOW stakeholders. They expect



to upload trained ML models and relevant scripts (either in Python or R) and then easily reference them when building flows. When these flows are deployed onto edge nodes the system have to make sure that all resources needed are downloaded into the relevant edge nodes.

Last but not least, considering the nature and the scale of the edge/fog applications, data security is also an imperative requirement underlined by the majority of the respondents so far. RAINBOW platform needs to find appropriate methods and techniques to guarantee data integrity and privacy throughout data life cycle.

6 System's Functional & Non-Functional Requirements

Stakeholders' needs and desires cannot be translated into requirements using an automated process. On the contrary, it needs an iterative, evolving strategy in order to transform ambiguous needs into requirements liable to contribute to the architectural design of the system.

6.1 Types of Requirements

Generally, requirements can be classified into two main categories: i) the **functional** and ii) the **non-functional** requirements.

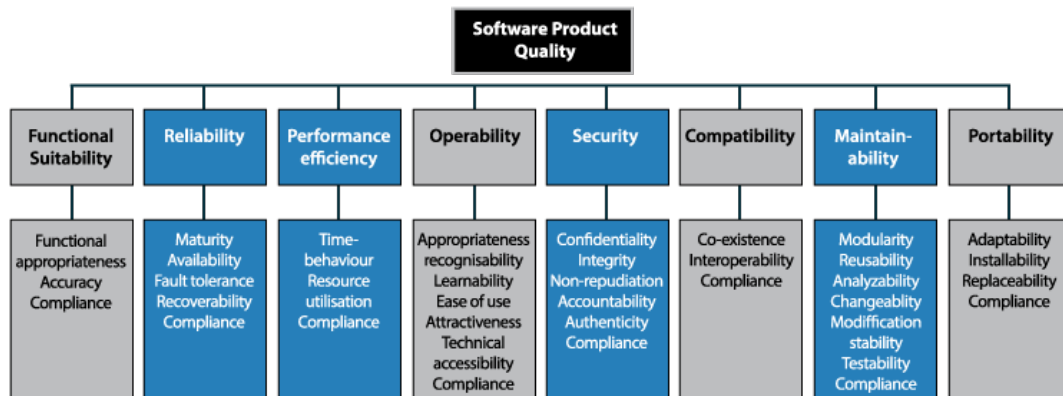
On the one hand, functional requirements are indispensable parts of the product. A functional requirement is used to describe the service that the software must offer to the end users. It can be a business process, a calculation, data manipulation, or a user interaction.

On the other hand, non-functional requirements relate to the desired quality aspects that should be satisfied by the architectural components of the RAINBOW eco-system that, in turn, must satisfy the functional requirements previously introduced. To this end, the widely accepted, by the software and research community, ISO/IEC 25010:2017 software quality assurance model was selected to create a shared conceptualization of the non-technical attributes [97]. The fundamental objective of the ISO/IEC25010:2017 standard is to address some of the well-known human biases that can adversely affect the delivery and perception of a software development project while it also determines which quality characteristics will be taken into account when evaluating the properties of a software product. The ISO/IEC 25010:2017 quality model classifies software quality in a structured set of characteristics and sub-characteristics, as follows:

- **Functional suitability:** It refers to a set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied needs. Indicative sub-characteristics include software functional completeness and functional correctness.
- **Reliability:** It refers to a set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time. Indicative sub-characteristics include software maturity, fault tolerance, recoverability and reliability compliance.
- **Usability:** It refers to a set of attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users. Indicative sub-characteristics include understandability, learnability, operability, attractiveness, and usability compliance.
- **Efficiency:** It refers to a set of attributes that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions. Indicative sub-characteristics include time behaviour, resource utilization, latency, service availability, and efficiency compliance.

- **Maintainability:** It refers to a set of attributes that bear on the effort needed to make specified modifications. Indicative sub-characteristics include: analyzability, changeability, stability, testability and maintainability compliance.
- **Portability:** It refers to a set of attributes that bear on the ability of software to be transferred from one environment to another. Indicative sub-characteristics include adaptability, installability, co-existence with other software, replaceability and portability compliance.
- **Security:** It refers to a set of attributes that define the degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization.
- **Compatibility:** It refers to a set of attributes that define the degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment.

Each quality sub-characteristic (e.g. adaptability) is further divided into attributes. An attribute is an entity which can be verified or measured in the software product. Attributes are not defined in the standard, as they vary between different software products. An overview of the aforementioned characteristics is provided in the following figure.



6.2 RAINBOW Functional Requirements

In the enumerated listings that follow, we make a concrete mapping between “RAINBOW’s Value Propositions” and the functional requirements that they correlate to each. In parallel, a brief description for each functional requirement, as well as the corresponding user role are also provided.

6.2.1 VP1 - Cloud Service Modelling

ID	FR.1
----	------



Title	Graphically describe application topology, denote intercommunication and dependencies among the applications' services and sub-components.
User Roles	Service Operator and Service Developer
Description	The Service Operator and Service Developer should be able to design a Fog application in a graphical manner. The application topology should be described as a directed acyclic graph (DAG), with a node representing a service/component of the application and edges denoting the runtime relationships between them. Each service should be a self-contained binary that includes system libraries, tools, and other files for the fog service developer's executable code, while the relationships between services should denote their communication dependencies at runtime.

ID	FR.2
Title	Annotate the graphical description with constraints, optimization policies and QoS requirements (aka "Configurations") through a unified and abstract service model.
User Roles	Service Operator
Description	<p>The Service Operator should be able to express graphically, user-defined constraints for each fog service in regard to fog node and network link requirements, security and privacy constraints, data and service availability, affinity and monitoring. The Service Operator should be able to define high-level policies for orchestration the Fog services based on monitored data. Annotated policies and configurations can be defined at various application granularity levels:</p> <ol style="list-style-type: none"> 1. Application level: Configurations and policies in regard to the overall fog application 2. Node level: specific policies and requirements for the respective service 3. Edge level: policies configurations for the relationship between services. <p>The configurations and policies model should be extensible to allow users to create custom policies. Users must be able to use the annotated entities without any further modification in the business logic of the under-development application.</p>

ID	FR.3
----	------



Title	Translate high-level service model to a deployment description
User Roles	RAINBOW Developer
Description	The RAINBOW platform should receive the annotated service graph model and produce a valid deployment description. This means that the platform should parse the model to validate its syntax and verify that the optimization policies and constraints are valid. Each node and edge with their associated metadata should be mapped to the format of the underlying orchestrator. The model parser should be extensible and be able to integrate with multiple orchestration technologies.

6.2.2 VP2 - Orchestration algorithms

ID	FR.4
Title	Access multi-level application behaviour and performance monitoring data
User Roles	Service Operator
Description	The RAINBOW platform must provide its users with access to real-time and historical monitoring data via the RAINBOW graphical user interface. The RAINBOW platform should support resource utilization metrics and user-defined metrics about the performance of the underlying infrastructure (e.g., fog nodes, network connections) and the fog application (e.g., fog services).

ID	FR.5
Title	Optimize the collection and aggregation of monitoring data
User Roles	Service Operator
Description	The platform should provide the capability to define adaptive monitoring strategies for monitoring streams of edge devices based on statistical analysis. The monitored data, the granularity level, and the intrusiveness at which monitoring data is collected and logged throughout the entire lifespan of an application should be determined by the user via the provided deployment assembly compiled based on user's preferences.



ID	FR.6
Title	Distributed control divided into local control loops
User Roles	RAINBOW Developer
Description	<p>The orchestration system shall support running localized control loops, which contribute to a distributed system.</p> <p>Each control loop will be responsible for orchestrating the nodes in its vicinity. The orchestration tasks include deploying services on the nodes, collecting monitoring data from the nodes, evaluating the data, as well as planning and executing corrective elasticity actions.</p> <p>The local control loops report aggregated data to their superior control loops, which delegate deployment actions to the local loops.</p>

ID	FR.7
Title	Periodic monitoring data pulling
User Roles	RAINBOW Developer
Description	<p>The platform should be able to periodically pull monitoring data from its assigned fog nodes. The orchestration system shall periodically pull aggregated monitoring data from the RAINBOW platform for all deployed applications.</p> <p>Both components are part of both local and superior control loops.</p>

ID	FR.8
Title	Clustering of low-power nodes
User Roles	RAINBOW Developer, Infrastructure Provider
Description	<p>Low-power fog nodes shall be grouped in a cluster with the cluster head being a more powerful node.</p> <p>This allows the low-power nodes to conserve power and provides a single point of contact for the control loop. If a cluster head fails, a new one shall be elected within the nodes and communicated to the control loop.</p>

ID	FR.9
Title	Data collection within a low-power cluster
User Roles	RAINBOW Developer, Infrastructure Provider



Description	<p>The cluster head shall passively collect monitoring data from its nodes. The RAINBOW platform should provide a mechanism capable of pulling monitoring data from the cluster head only.</p> <p>This allows the low-power nodes to conserve power and reduces the amount of communication that is necessary.</p>
-------------	--

ID	FR.10
Title	Deployment of a new service
User Roles	RAINBOW Developer, Service Operator
Description	<p>The orchestration system shall deploy newly submitted services in the fog. The target nodes must be chosen according to the SLOs of the service and the capabilities of the fog nodes.</p>

ID	FR.11
Title	SLO adherence
User Roles	RAINBOW Developer
Description	<p>The orchestration system shall maintain the SLOs for each deployed application.</p> <p>This requirement is further decomposed into evaluation of monitoring data and corrective elasticity actions.</p>

ID	FR.12
Title	Definition of supported SLOs
User Roles	Service Developer
Description	<p>The list of supported SLOs shall be defined by service developers (in the frame of QoS definition).</p> <p>The service developers know what capabilities their services have and how to adjust their deployments if SLOs are not met. Thus, they must formalize these SLOs, such that the RAINBOW orchestrator can maintain them. Each SLO shall contain a specification of what data needs to be monitored, how it is evaluated, and how to react in case of a violation of the SLO.</p>

ID	FR.13
----	-------



Title	Customization of SLOs
User Roles	Service Operator
Description	Service operators, who want to deploy a service, shall be able to customize parameters for the SLOs. The RAINBOW orchestrator will then be responsible for maintaining them.

ID	FR.14
Title	Evaluation of monitoring data
User Roles	RAINBOW Developer
Description	The orchestration system shall validate the SLOs for each deployed application against the monitoring data. This entails checking whether the current monitoring data satisfies the constraints specified in the SLOs.

ID	FR.15
Title	Corrective elasticity actions on violated SLOs
User Roles	RAINBOW Developer
Description	The orchestration system shall plan and execute corrective elasticity actions for violated SLOs. If the evaluation of monitoring data yields one or more violated SLOs, the orchestration system shall plan corrective elasticity actions, based on the definitions of the SLOs and instruct the appropriate target nodes to execute them.

ID	FR.16
Title	Available and used resources tracking
User Roles	RAINBOW Developer
Description	The RAINBOW platform should provide a resource supervision mechanism able to keep track of the available and used resources on all assigned fog nodes. This is important for planning new deployments and executing corrective actions on existing deployments.



ID	FR.17
Title	Target fog nodes selection for action execution
User Roles	RAINBOW Developer
Description	The resource supervision mechanism shall choose which fog nodes are best suited for hosting a new service deployment or carrying out a corrective action on an existing one. These decisions must take the required SLOs, as well as the available resources and locations of the fog nodes into account.

ID	FR.18
Title	Local sidecar agent on fog nodes
User Roles	RAINBOW Developer, Infrastructure Provider
Description	The edge-related orchestration actions shall be undertaken by an agent that is deployed on each fog node (sidecar approach). This includes e.g., the deployment of a new service on the fog node or the stopping of a running service.

ID	FR.19
Title	Resource reporting by fog node sidecar agent
User Roles	RAINBOW Developer, Infrastructure Provider
Description	The sidecar agent deployed on the fog nodes shall be able to report the available and used resources on its fog node to the resource supervision mechanism.

ID	FR.20
Title	User management
User Roles	RAINBOW Developer, Service Operator
Description	The RAINBOW orchestrator shall provide the facilities to manage the registered users of the platform and their permissions.



ID	FR.21
Title	Infrastructure catalogue
User Roles	RAINBOW Developer, Infrastructure Provider, Service Operator
Description	The RAINBOW platform shall provide a catalogue of the available infrastructure, which is maintained automatically by the orchestrator. Fog nodes shall be grouped into categories, based on their hardware capabilities. Service operators can browse through these categories to see what kind of infrastructure is available.

6.2.3 VP3 - Efficient Data Storage, Querying and Processing

ID	FR.22
Title	Compile and Execute of analytic insights through a high-level and descriptive query model
User Roles	Service Operator
Description	The platform should provide the user with a high-level query language for composing fog service analytics based on monitoring metrics. The platform should support descriptive and summary statistics (possibly ML, NNs, Graph). The user can provide hints (fog-related constraints and optimization) about the execution of the analytics computations. The platform should map the high-level analytics description to a distributed analytic job, optimized and adhering to user-defined constraints (e.g., data-movement, latency).

ID	FR.23
Title	Analytics execution mode (batch, streaming, offline)
User Roles	Service Operator and RAINBOW Developer



Description	<p>Support batch execution mode</p> <ul style="list-style-type: none"> Support efficient data retrieving from the underlying storage for historical analysis and predictive analytics (e.g., training ML models) <p>Support streaming execution mode</p> <ul style="list-style-type: none"> Efficient data-structures and algorithms for accessing data in near-real time. Support streaming operators (e.g., window, accumulated operations)
-------------	--

ID	FR.24
Title	Optimization through scheduling policies (latency guarantees) and restrictions (e.g., data movement)
User Roles	Service Operator
Description	<p>The RAINBOW platform must be able to find a near-optimal plan for executing analytic jobs based on user's preferences and constraints (e.g., data restrictions, and QoS requirements)</p> <p>The platform should be able to adapt the execution of the analytic jobs in near-real time based on infrastructure and application workload monitoring.</p>

ID	FR.25
Title	Efficient data storage and placement based on restrictions
User Roles	Service Operator and RAINBOW Developer
Description	<p>The RAINBOW platform must store data efficiently based on restrictions (e.g. data movement, node resources).</p> <p>The data placement should take into account the dynamic environment and adapt to the changes of the nodes.</p>

ID	FR.26
Title	Specify monitoring data for storage based on analytic queries and orchestration SLOs
User Roles	Service Operator



Description	<p>The RAINBOW platform must store monitoring data specified by the SLOs that the orchestrator will have to adhere to.</p> <p>The stored monitoring data can also be specified based on the analytics that will be available by the platform.</p>
-------------	---

6.2.4 VP4 - Secure Zero-touch configuration

ID	FR.27
Title	Trust-Aware Service Graph Chain Composition
User Roles	RAINBOW Developer, Service Operator, Infrastructure Provider
Description	<p>One key feature of RAINBOW is the establishment, monitoring and maintenance of the trusted state of the overall service graph chain, comprising multiple fog/edge nodes, in a dynamic networking environment where the network topology constantly changes (addition and/or removal of nodes) and data packet routing does not rely on static routing tables. Thus, the platform should support the secure update of the network topology and service graph composition by allowing fog/edge nodes to seamlessly join the network using an almost zero-touch configuration. The RAINBOW Orchestrator will introduce a new capability called Secure Zero Touch Provisioning (S-ZTP) which allows the automatic and secure establishment of trust, called enrolment, between new fog/edge nodes joining a network, without human intervention. Specific functionalities/requirements include:</p> <ul style="list-style-type: none"> • Eliminate the need of trust on “first node sight” or out-of-band trust establishment schemes, which, in practice can be very unreliable from the perspectives of trust model, organization and cost. • Operate in tandem with mesh networking protocols so that a secure and trusted network addressing scheme can be provided. RAINBOW platform will enhance the CJDNS protocol (from Layer 3 and upwards) by leveraging the root of trust capabilities of the fog/edge node for attesting its trust state and establishing secure and privacy-preserving communication channels materialized by the issue of TLS certificates. <p>The RAINBOW platform will augment the decision on the data routing policy (extracted from CJDNS), by factoring in the trust state of each node, in addition to securely extracted network-related parameters (i.e., dynamic load balancing).</p>

ID	FR.28
Title	Secure Remote Asset Management



User Roles	RAINBOW Developer, Service Operator, Infrastructure Provider
Description	<p>The platform should provide the capability of “remote upgrade” by enabling the secure update of micro applications, services and safety-critical functions running at the fog end without affecting the trusted state of the overall service graph chain. This will leverage the root of trust capabilities of each fog/edge node when it comes to secure boot, remote attestation and configuration integrity verification (Section 6.2.5). The Resource supervision mechanism, who is responsible for managing this process, will also be acting as the “verifier” for attesting the correct execution of the over-the-air (software & firmware) update and patching, enabled by the RAINBOW Sidecar agent acting as the “prover”. Required characteristics of RAINBOW’s secure remote asset management are:</p> <ul style="list-style-type: none"> • Minimized downtime in case of a safety failure or integrity violation (as a result of a cyber-attack) of a fog/edge node; • Increased efficiency by continuously and securely monitoring the health of a remote fog/edge node; • Enhanced security and trustworthiness by providing the necessary guarantees of the correct configuration and execution of a service running at the fog end.

ID	FR.29
Title	Secure Communication, Data Privacy and (User-controlled) Anonymity
User Roles	RAINBOW Developer, Service Provider and Service Operator
Description	<p>One key aspect of RAINBOW is the security and privacy guarantees of the user data exchanged (and stored) between the data and control planes. Thus, the platform should provide the necessary security and privacy extensions, as part of the Trust Overlay Mesh Network (FR.27), for ensuring data integrity, confidentiality and privacy preservation (during data routing) against network adversaries even in the case of a compromised fog/edge node. Secure communication is needed to retrieve security-related data and to steer local inspection and enforcement tasks. This will leverage the root of trust capabilities of the fog/edge node for enabling protection from data leakages, attacks and retaliations:</p> <ul style="list-style-type: none"> • Enable the protection of sensitive information; • It should be hard for an adversary to learn the secret information required for any action (e.g., authentication, encryption, anonymization, etc.); • Credentials should be stored on edge/fog node and must be protected from eavesdropping/leakage.



ID	FR.30
Title	Cryptographic primitives supported for Trust-Aware Service Graph Chains
User Roles	RAINBOW Developer
Description	<p>The RAINBOW platform must provide advanced security and trust establishment services (FR.31, FR.32, and FR.33), through secure cryptographic functions including secure authentication, encryption and signing functions, towards the creation of “communities of trusted fog/edge nodes”. Such primitives form the basic security and privacy-preserving functionalities that support the more complex operations of Remote Asset Management, Service & Execution Integrity Verification and Layered Attestation Orchestration. The platform will, therefore, include both symmetric cryptography (used for tasks such as verifiable computing) and asymmetric cryptography (also known as public key cryptography and used for establishing secure connection over the CJDNS mesh networking protocol). Namely this includes:</p> <ul style="list-style-type: none"> • Key generation and storage functionalities; • Hash functions; • MAC; • Symmetric encryption; • Digital (anonymous) signatures; • Public key encryption and key exchange; • Direct Anonymous Attestation (DAA) for enhanced data privacy and (user-controlled) anonymity (FR.29).

6.2.5 VP5 - Configuration Integrity Verification

ID	FR.31
Title	Service Configuration Integrity Verification
User Roles	RAINBOW Developer, Service Operator, and Service Developer
Description	<p>One of the main functionalities of the RAINBOW framework is related to software integrity and correctness of the services deployed over the target fog/edge nodes; both during the service deployment and execution phases. Thus, the platform should be able to verify (by providing runtime verifiable evidence) that the software building blocks of a service, running on a fog/edge node are trustworthy and have not been tampered with by intruders or malware. This dynamic assessment and integrity preservation will leverage the RAINBOW attestation enablers (FR.33)</p>



	<p>and should take place either locally in the fog node (RAINBOW Sidecar Agent) or remotely (managed by the Resource supervision mechanism). The root of trust hardware or software, in the fog node, must offer the following functionalities:</p> <ul style="list-style-type: none"> • Support software measurement extraction and secure measurement reporting, using the RAINBOW crypto primitives (FR.30); • Supporting remote attestation functionalities (FR.33); • Supporting sealing and binding operations.
--	--

ID	FR.32
Title	Service Execution Integrity Verification
User Roles	RAINBOW Developer, Service Operator, and Service Developer
Description	Besides the verification of configurational properties of deployed services (FR.31 towards ensuring the integrity of service binaries), the RAINBOW platform will also target the verification of low-level behavioural execution properties during the execution of a service or safety-critical function. Such behavioural properties (i.e., execution paths to specific memory regions, ports and network interfaces, etc.) capture a service's runtime behaviour which is reflected as a Control-Flow Graph (CFG). The platform will leverage the runtime control- and data-flow attestation enablers (FR.33) for verifying the dynamic state of a fog/edge node so that it can capture control-flow attacks at the binary level of a service while coping with the strict scalability, performance, robustness and reliability requirements of distributed fog-based environments.

ID	FR.33
Title	Service Behavioural Analysis through Layered Attestation Orchestration
User Roles	RAINBOW Developer, Service Operator
Description	The RAINBOW platform should be able to provide runtime behavioural attestation services, targeting both the software and hardware layers and covering all phases of a fog/edge node's execution; from the trusted boot and integrity measurement of a node, enabling the generation of static, boot-time or load-time evidence of the system's components correct configuration (FR.31), to the runtime behavioral attestation of those safety-critical components of a system providing strong guarantees on the correctness of the <i>control</i> - and <i>information-flow</i> properties (FR.32), thus, enhancing the performance and scalability when composing secure service graphs from potentially insecure nodes.



ID	FR.34
Title	Recovery from Compromised Service Graphs
User Roles	Service Operator, Infrastructure Provider
Description	Even with strong service integrity verification and execution correctness attestation services in place, it might be possible that an unknown threat or vulnerability might lead to service graph integrity violations. In case of critical services, it is vital to quickly recover and return to a safe state. The RAINBOW platform will provide the necessary security extensions so that when a compromised service graph is detected, the Resource supervision mechanism will initiate the replacement of the compromised components with cleans instances to return to a safe and trusted condition. Replacement of a compromised function will only possible under specific conditions. It is quite simple for stateless functions (as in case of many network functions), but it is more difficult when a state must be managed.

ID	FR.35
Title	Local Security Processing and Programmability
User Roles	RAINBOW Developer
Description	One important security feature of RAINBOW's platform is the provision of scalable (layered) attestation services which requires correlation of attestation data and policies and local security processing for improved efficiency. Thus, the platform will support the deployment of local agents to all fog/edge nodes which will be responsible (among other things) for the runtime data and execution stream monitoring and introspection towards the efficient tracing of the control- and information-flow execution paths need by the RAINBOW attestation and configuration integrity verification services. The RAINBOW Sidecar agent will provide dynamic tracing functionalities, as programmable components, enabling the continuous monitoring and the configurational and behavioural execution properties of interest. This provides the trusted anchor with the compiled control- and information-flow graphs (CFGs & DFGs) that represent the runtime state of a remote fog/edge node to be attested and verified during runtime. Programmability will provide the option to change the monitoring processes at runtime.

ID	HW.1
----	------



Title	Physical Requirements for Hardware-based Attestation Policy Enablers
User Roles	RAINBOW developer (hardware)
Description	The hardware developer should integrate a hardware token or device to meet the following features (as also required for the provision of the secure attestation services – FR.33). It should support two supply voltage levels, i.e. 1.8V or 3.3V. Furthermore, the hardware should also include a low standby power consumption mode where it consumes less than 120µA. For the communication to the host controller, SPI up to 43MHz with bus encryption should be used, and an internal memory should be supported. The volatile memory should have space for three asymmetric keys and non-volatile memory up to seven keys. The temperature range of the field of application should be from -40°C...+105°C. Common Criteria EAL4+ shall be fulfilled. The integrated device should be tamper-resistant and include shielding and sensors against physical and logical attacks.

ID	HW.2
Title	Functional Requirements for Hardware-based Attestation Policy Enablers
User Roles	RAINBOW developer (hardware)
Description	<p>The hardware should support following functional requirements and crypto primitives (FR.30) as an enabler for hardware-based attestation policy. It should support following cryptographic algorithms:</p> <ul style="list-style-type: none"> • RSA-1024 and RSA-2048 • SHA-1 and SHA-256 • ECC NIST P256 • ECC BN256 <p>The hardware should enable secure boot mechanisms and include sealed storage. The hardware should support the protocols of attestation, integrity measurement and DAA.</p>

6.3 User Roles to Functional Requirements Mapping

In the table that follows, we make a concrete mapping between user roles and the functional requirements that they correlate to.

User Role	Functional Requirements
-----------	-------------------------



Service Operator	<p>FR.2 Annotate the graphical description with constraints, optimization policies and QoS requirements (aka “Configurations”) through a unified and abstract service model.</p> <p>FR.4 Access multi-level application behaviour and performance monitoring data</p> <p>FR.5 Optimize the collection and aggregation of monitoring data</p> <p>FR.10 Deployment of a new service</p> <p>FR.13 Customization of SLOs</p> <p>FR.20 User management</p> <p>FR.22 Compile and Execute of analytic insights through a high-level and descriptive query model</p> <p>FR.23 Analytics execution mode (batch, streaming, offline)</p> <p>FR.24 Optimization through scheduling policies (latency guarantees) and restrictions (e.g., data movement)</p> <p>FR.25 Efficient data storage and placement based on restrictions</p> <p>FR.26 Specify monitoring data for storage based on analytic queries and orchestration SLOs</p> <p>FR.27 Trust-Aware Service Graph Chain Composition</p> <p>FR.28 Secure Remote Asset Management</p> <p>FR.29 Secure Communication, Data Privacy and (User-controlled) Anonymity</p> <p>FR.31 Service Configuration Integrity Verification</p> <p>FR.32 Service Execution Integrity Verification</p> <p>FR.33 Service Behavioural Analysis through Layered Attestation Orchestration</p> <p>FR.34 Recovery from Compromised Service Graphs</p>
RAINBOW Developer	<p>FR.1 Graphically describe application topology, denote intercommunication and dependencies among the applications' services and sub-components.</p> <p>FR.3 Translate high-level service model to a deployment description</p> <p>FR.6 Distributed control divided into local control loops</p> <p>FR.7 Periodic monitoring data pulling</p> <p>FR.8 Clustering of low-power nodes</p> <p>FR.9 Data collection within a low-power cluster</p> <p>FR.10 Deployment of a new service</p> <p>FR.11 SLO adherence</p> <p>FR.14 Evaluation of monitoring data</p> <p>FR.15 Corrective elasticity actions on violated SLOs</p> <p>FR.16 Available and used resources tracking</p> <p>FR.17 Target fog nodes selection for action execution</p> <p>FR.18 Local sidecar agent on fog nodes</p> <p>FR.19 Resource reporting by fog node sidecar agent</p> <p>FR.20 User management</p> <p>FR.23 Analytics execution mode (batch, streaming, offline)</p>

	FR.25 Efficient data storage and placement based on restrictions FR.27 Trust-Aware Service Graph Chain Composition FR.28 Secure Remote Asset Management FR.29 Secure Communication, Data Privacy and (User-controlled) Anonymity FR.30 Cryptographic primitives supported for Trust-Aware Service Graph Chains FR.31 Service Configuration Integrity Verification FR.32 Service Execution Integrity Verification FR.33 Service Behavioural Analysis through Layered Attestation Orchestration FR.35 Local Security Processing and Programmability
Service Developer	FR.1 Graphically describe application topology, denote intercommunication and dependencies among the applications' services and sub-components. FR.12 Definition of supported SLOs FR.29 Secure Communication, Data Privacy and (User-controlled) Anonymity FR.31 Service Configuration Integrity Verification FR.32 Service Execution Integrity Verification
Infrastructure Provider	FR.8 Clustering of low-power nodes FR.9 Data collection within a low-power cluster FR.18 Local sidecar agent on fog nodes FR.19 Resource reporting by fog node sidecar agent FR.27 Trust-Aware Service Graph Chain Composition FR.28 Secure Remote Asset Management FR.34 Recovery from Compromised Service Graphs Cloud and Fog providers should be able to register and manage their infrastructure and service offerings Monitor cloud/fog offering allocation and consumption

Table 5: Functional Requirements Relation to User Role

6.4 RAINBOW Non-Functional Requirements

In the enumerated listings that follow, we make a concrete mapping between the core quality model attributes and the functional requirements that they correlate to. In parallel, for each non-functional requirement, a brief description of the RAINBOW ecosystem relevant characteristics is also provided.

NFR.1	Functional Suitability
--------------	-------------------------------



Description	<p>This characteristic represents the degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions. This characteristic is composed of the following sub-characteristics:</p> <ul style="list-style-type: none"> • Functional completeness. Degree to which the set of functions covers all the specified tasks and user objectives. • Functional correctness. Degree to which a product or system provides the correct results with the needed degree of precision. • Functional appropriateness. Degree to which the functions facilitate the accomplishment of specified tasks and objectives.
Functional Requirements	<p>FR.3 Translate high-level service model to a deployment description FR.6:Distributed control divided into local control loops FR.7 Periodic monitoring data pulling FR.10 Deployment of a new service FR.11 SLO adherence FR.14 Evaluation of monitoring data FR.15 Corrective elasticity actions on violated SLOs FR.17 Target fog nodes selection for action execution FR.18 Local sidecar agent on fog nodes FR.22 Compile and Execute of analytic insights through a high-level and descriptive query model FR.24 Optimization through scheduling policies (latency guarantees) and restrictions (e.g., data movement) FR.28 Secure Remote Asset Management FR.31 Service Configuration Integrity Verification FR.32 Service Execution Integrity Verification FR.33 Service Behavioural Analysis through Layered Attestation Orchestration</p>

NFR.2	Performance Efficiency
Description	<p>This characteristic represents the performance relative to the amount of resources used under stated conditions. This characteristic is composed of the following sub-characteristics:</p> <ul style="list-style-type: none"> • Time behavior. Degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements.



	<ul style="list-style-type: none"> • Resource utilization. Degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements. • Capacity. Degree to which the maximum limits of a product or system parameter meet requirements.
Functional Requirements	<p>FR.3 Translate high-level service model to a deployment description</p> <p>FR.4 Access multi-level application behaviour and performance monitoring data</p> <p>FR.5 Optimize the collection and aggregation of monitoring data</p> <p>FR.6 Distributed control divided into local control loops</p> <p>FR.7 Periodic monitoring data pulling</p> <p>FR.8 Clustering of low-power nodes</p> <p>FR.9 Data collection within a low-power cluster</p> <p>FR.12 Definition of supported SLOs</p> <p>FR.13 Customization of SLOs</p> <p>FR.15 Corrective elasticity actions on violated SLOs</p> <p>FR.16 Available and used resources tracking</p> <p>FR.17 Target fog nodes selection for action execution</p> <p>FR.18 Local sidecar agent on fog nodes</p> <p>FR.19 Resource reporting by fog node sidecar agent</p> <p>FR.22 Compile and Execute of analytic insights through a high-level and descriptive query model</p> <p>FR.23 Analytics execution mode (batch, streaming, offline)</p> <p>FR.24 Optimization through scheduling policies (latency guarantees) and restrictions (e.g., data movement)</p> <p>FR.25 Efficient data storage and placement based on restrictions</p>

NFR.3	Security Services Performance and Cost-Effectiveness
Description	<p>Besides the enhanced security and privacy levels that need to be achieved for the deployed fog/edge nodes, fog-based environments have to also consider the impact that the integrated security protocols will have on performance and the additional cost of implementation. In RAINBOW, there is the inherent assumption that each fog/edge node is equipped with a Root-of-Trust and more particularly a Trusted Platform Module (TPM) for providing hardware-based crypto acceleration functionalities. In this context, the cryptographic primitives and attestation/configuration integrity verification protocols implemented, based on the use of TPMs, must be efficient to be used in practice. In order for TPMs to support a wide range of devices and implementations (particularly for smaller embedded devices such as those used in fog-based environments), efficiency is a</p>

	<p>core concern. This will also have an economic impact on, for example, fog devices manufacturing costs. Nevertheless, security should not be compromised so the specific algorithms and protocols developed in RAINBOW must balance security and functionality. Specifically:</p> <ul style="list-style-type: none"> • It should be feasible to implement the (layered) attestation algorithms and tracing mechanisms on platforms with restricted memory, while providing an acceptable performance; • The selected crypto algorithms can be implemented securely on an identified platform, e.g., x86 for firmware TPM, relatively constrained 32-bit CPUs, etc. • The use of RAINBOW to attest (during runtime) services and code snippets, running in a fog/edge node, should be similar or better than current static vulnerability analysis mechanisms (also benchmarked against the use of Trusted Execution Environments).
Functional Requirements	<p>FR.27 Trust-Aware Service Graph Chain Composition FR.28 Secure Remote Asset Management FR.29 Secure Communication, Data Privacy and (User-controlled) Anonymity FR.30 Cryptographic primitives supported for Trust-Aware Service Graph Chains FR.31 Service Configuration Integrity Verification FR.32 Service Execution Integrity Verification FR.33 Service Behavioural Analysis through Layered Attestation Orchestration FR.34 Recovery from Compromised Service Graphs FR.35 Local Security Processing and Programmability</p>
NFR.4	Compatibility
Description	<p>Degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment. This characteristic is composed of the following sub-characteristics:</p> <ul style="list-style-type: none"> • Co-existence. Degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.



	<ul style="list-style-type: none"> • Interoperability. Degree to which two or more systems, products or components can exchange information and use the information that has been exchanged. <p>The need to implement new paradigms for fog-based environments cannot break existing (security) practice and processes. In this respect, it is important to facilitate interoperability and, most of all, integrability with existing tools. The RAINBOW run-time components should be, architectural-wise and implementation-wise, close to the industry. For this reason, RAINBOW will provide support to a number of commonly used standards, standard syntax, APIs, widely available tools, technologies, methodologies and best practices. The system should support abstractions which will hide from developers and their applications details regarding the system and application infrastructure. RAINBOW will also support uniform service descriptions such as SLA offerings with clear policies and guidelines.</p> <p>Furthermore, RAINBOW will provide a REST API for providing full control of its security services, including the ability to : i) upload, remove, start, stop and configure all security algorithms and protocols, ii) add and remove tracing and inspection programmable agents, and iii) configure authentication, authorization and access control.</p>
Functional Requirements	<p>FR.1 Graphically describe application topology, denote intercommunication and dependencies among the applications' services and sub-components.</p> <p>FR.2 Annotate the graphical description with constraints, optimization policies and QoS requirements (aka "Configurations") through a unified and abstract service model.</p> <p>FR.3 Translate high-level service model to a deployment description</p> <p>FR.6 Distributed control divided into local control loops</p> <p>FR.12 Definition of supported SLOs</p> <p>FR.17 Target fog nodes selection for action execution</p> <p>FR.18 Local sidecar agent on fog nodes</p> <p>FR.30 Cryptographic primitives supported for Trust-Aware Service Graph Chains</p> <p>FR.33 Service Behavioural Analysis through Layered Attestation Orchestration</p> <p>FR.34 Recovery from Compromised Service Graphs</p> <p>FR.35 Local Security Processing and Programmability</p>
NFR.5	Usability



Description	<p>Degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. This characteristic is composed of the following sub-characteristics:</p> <ul style="list-style-type: none"> • Appropriateness recognizability. Degree to which users can recognize whether a product or system is appropriate for their needs. • Learnability. degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use. • Operability. Degree to which a product or system has attributes that make it easy to operate and control. • User error protection. Degree to which a system protects users against making errors. • User interface aesthetics. Degree to which a user interface enables pleasing and satisfying interaction for the user. • Accessibility. Degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use. <p>Taking into consideration all the above characteristics of usability, the RAINBOW platform will support automatic and seamless deployment making it very easy to use and learn. The development platform and tools will be hosted on the fog and will be accessible through a web browser. RAINBOW will have all the content and user interface organized logically and it will provide a presentation interface (e.g., menu and navigation, reporting, user controls etc.)</p>
Functional Requirements	<p>FR.1 Graphically describe application topology, denote intercommunication and dependencies among the applications' services and sub-components.</p> <p>FR.2 Annotate the graphical description with constraints, optimization policies and QoS requirements (aka "Configurations") through a unified and abstract service model.</p> <p>FR.4 Access multi-level application behaviour and performance monitoring data</p> <p>FR.12 Definition of supported SLOs</p> <p>FR.13 Customization of SLOs</p> <p>FR.20 User management</p> <p>FR.28 Secure Remote Asset Management</p>



	FR.29 Secure Communication, Data Privacy and (User-controlled) Anonymity FR.31 Service Configuration Integrity Verification FR.32 Service Execution Integrity Verification FR.33 Service Behavioural Analysis through Layered Attestation Orchestration FR.34 Recovery from Compromised Service Graphs
NFR.6	Reliability
Description	<p>Degree to which a system, product or component performs specified functions under specified conditions for a specified period of time. This characteristic is composed of the following sub-characteristics:</p> <ul style="list-style-type: none"> • Maturity. Degree to which a system, product or component meets needs for reliability under normal operation. • Availability. Degree to which a system, product or component is operational and accessible when required for use. • Fault tolerance. Degree to which a system, product or component operates as intended despite the presence of hardware or software faults. • Recoverability. Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system. <p>Within the context of RAINBOW, specific mechanisms will be architecturally defined and implemented that guarantee that any application can be securely deployed.</p>
Functional Requirements	FR.6 Distributed control divided into local control loops FR.7 Periodic monitoring data pulling FR.11 SLO adherence FR.14 Evaluation of monitoring data FR.15 Corrective elasticity actions on violated SLOs FR.17 Target fog nodes selection for action execution FR.22 Compile and Execute of analytic insights through a high-level and descriptive query model FR.23 Analytics execution mode (batch, streaming, offline) FR.24 Optimization through scheduling policies (latency guarantees) and restrictions (e.g., data movement) FR.25 Efficient data storage and placement based on restrictions FR.27 Trust-Aware Service Graph Chain Composition FR.28 Secure Remote Asset Management

	<p>FR.29 Secure Communication, Data Privacy and (User-controlled) Anonymity</p> <p>FR.31 Service Configuration Integrity Verification</p> <p>FR.32 Service Execution Integrity Verification</p> <p>FR.33 Service Behavioural Analysis through Layered Attestation Orchestration</p> <p>FR.34 Recovery from Compromised Service Graphs</p>
--	--

NFR.7	Security
Description	<p>The degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization. This characteristic is composed of the following sub-characteristics:</p> <ul style="list-style-type: none"> • Confidentiality. Degree to which a product or system ensures that data are accessible only to those authorized to have access. • Integrity. Degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data. • Non-repudiation. degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later. • Accountability. Degree to which the actions of an entity can be traced uniquely to the entity. • Authenticity. Degree to which the identity of a subject or resource can be proved to be the one claimed.
Functional Requirements	<p>FR.1 Graphically describe application topology, denote intercommunication and dependencies among the applications' services and sub-components.</p> <p>FR.2 Annotate the graphical description with constraints, optimization policies and QoS requirements (aka "Configurations") through a unified and abstract service model.</p> <p>FR.3 Translate high-level service model to a deployment description</p> <p>FR.4 Access multi-level application behaviour and performance monitoring data</p> <p>FR.6 Distributed control divided into local control loops</p> <p>FR.7 Periodic monitoring data pulling</p> <p>FR.11 SLO adherence</p> <p>FR.12 Definition of supported SLOs</p> <p>FR.13 Customization of SLOs</p> <p>FR.17 Target fog nodes selection for action execution</p>

	<p>FR.27 Trust-Aware Service Graph Chain Composition</p> <p>FR.28 Secure Remote Asset Management</p> <p>FR.29 Secure Communication, Data Privacy and (User-controlled) Anonymity</p> <p>FR.30 Cryptographic primitives supported for Trust-Aware Service Graph Chains</p> <p>FR.31 Service Configuration Integrity Verification</p> <p>FR.32 Service Execution Integrity Verification</p> <p>FR.33 Service Behavioural Analysis through Layered Attestation Orchestration</p> <p>FR.34 Recovery from Compromised Service Graphs</p> <p>FR.35 Local Security Processing and Programmability</p>
--	---

NFR.8	Maintainability
Description	<p>This characteristic represents the degree of effectiveness and efficiency with which a product or system can be modified to improve it, correct it or adapt it to changes in environment, and in requirements. This characteristic is composed of the following sub-characteristics:</p> <ul style="list-style-type: none"> • Modularity. Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components. • Reusability. Degree to which an asset can be used in more than one system, or in building other assets. • Analysability. Degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified. • Modifiability. Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality. • Testability. Degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met.
Functional Requirements	<p>FR.2 Annotate the graphical description with constraints, optimization policies and QoS requirements (aka “Configurations”) through a unified and abstract service model.</p> <p>FR.6 Distributed control divided into local control loops</p>



	FR.10 Deployment of a new service FR.12 Definition of supported SLOs FR.13 Customization of SLOs FR.18 Local sidecar agent on fog nodes FR.27 Trust-Aware Service Graph Chain Composition FR.28 Secure Remote Asset Management FR.29 Secure Communication, Data Privacy and (User-controlled) Anonymity
NFR.9	Portability
Description	<p>Degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another. This characteristic is composed of the following sub-characteristics:</p> <ul style="list-style-type: none"> • Adaptability. Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments. • Installability. Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment. • Replaceability. Degree to which a product can replace another specified software product for the same purpose in the same environment.
Functional Requirements	FR.1 Graphically describe application topology, denote intercommunication and dependencies among the applications' services and sub-components. FR.2 Annotate the graphical description with constraints, optimization policies and QoS requirements (aka "Configurations") through a unified and abstract service model. FR.3 Translate high-level service model to a deployment description FR.10 Deployment of a new service FR.12 Definition of supported SLOs FR.13 Customization of SLOs FR.18 Local sidecar agent on fog nodes FR.22 Compile and Execute of analytic insights through a high-level and descriptive query model FR.31 Service Configuration Integrity Verification FR.32 Service Execution Integrity Verification



	FR.33 Service Behavioural Analysis through Layered Attestation Orchestration
NFR.10	Security Protocol and Algorithm Agility
Description	<p>This characteristic represents the <i>flexibility</i> that needs to be offered by the RAINBOW platform to service providers and operators regarding the ability to choose among various static and dynamic software vulnerability analysis mechanisms, attestation enablers, cryptographic primitives and protocols that provide a specific security and privacy-preserving service, depending on the different types of security, privacy, trust and safety requirements of specific services. Specific sub-characteristics of interest include:</p> <ul style="list-style-type: none"> • Support for legacy security primitives/protocols. Degree to which legacy security mechanisms and protocols can be easily integrated and activated within the RAINBOW platform. • Ability to allow vendors and service operators to implement and integrate (or remove) new security protocols, should they prove to be cryptographically secure (or weaker), without the need of revisiting the RAINBOW security specifications. This way, for instance, if a security protocol is weakened by cryptanalysis in the future, it can be removed and replaced without changing the overall specification.
Functional Requirements	<p>FR.27 Trust-Aware Service Graph Chain Composition FR.28 Secure Remote Asset Management FR.30 Cryptographic primitives supported for Trust-Aware Service Graph Chains FR.31 Service Configuration Integrity Verification FR.32 Service Execution Integrity Verification FR.33 Service Behavioural Analysis through Layered Attestation Orchestration FR.34 Recovery from Compromised Service Graphs</p>
NFR.11	Support for additional application areas
Description	<p>Finally, we list some desirable functionalities for emerging application areas, in the context of fog-based environments (and beyond the envisioned use cases), such as remote voting, anonymous communications, enhanced data sharing through the use of Blockchains and IoT.</p>



- Support for a broader range of access policies.
- Functionality for key translation (i.e., re-encrypting provided data under a different key) towards enhanced secure communications and (encrypted) data search elasticity.
- Use RAINBOW Configuration Integrity Verification operations in Blockchains and other services such as verifiable data access.
- Secure logging of access to security operations in a decentralized and highly dynamic fog-based environment (ability to provide accountable decryption and similar constructs).
- Secure key backup and recovery.

7 Conclusion

The COVID-19 pandemic had consequences beyond the spread of the disease itself and efforts to deal with it. As the SARS-CoV-2 virus has spread around Europe, remote work become a one-way road in the few businesses that continued their operations. Business priorities changed drastically while the work-force dispersion lead to severe businesses structural issues. It is particularly evident that if the internal business communication was difficult, the sufficient inter-business communication was almost impossible.

Under such circumstances, communication between RAINBOW partners and their contacts was not feasible during the period of quarantine, which persisted for two and a half months (from mid-March until end of May) in most European countries. This situation had a severe effect in i) the interviews that RAINBOW consortium was planning to have with some of the potential stakeholders, and ii) the dissemination of questionnaire. The outcome has been that RAINBOW consortium did not receive a satisfactory number of replies from the interviews and questionnaire in time. In order to deal with this situation, RAINBOW consortium decided to include in this deliverable only the functional and non-functional derived from RAINBOW's demonstrators, the first demonstrator - "Human-Robot Collaboration in Industrial Ecosystems" and the third demonstrator - "Power Line Surveillance via Swarm of Drones". The second demonstrator is located in Italy and is still (mid-June) in suspension.

This deliverable includes the results of the stakeholder analysis which was conducted during the first 6 months of the RAINBOW project as part of the Task 1.1 "RAINBOW Requirements Analysis and Stakeholders' Identification". Therefore, the edge/fog computing landscape, the RAINBOW stakeholders and actors as well as the categories of applications and use cases were explicitly defined, in order to meet the scope of this deliverable which was the extraction of the functional and non-functional requirements and the definition of the stakeholders.



This deliverable aimed to specify in an explicit and coherent manner the functional and non-functional requirements, considering the stakeholder's needs and following the ISO/IEC/IEEE 29148:2011, in order to pave the way towards the definition of RAINBOW platform architecture (D1.3). However, COVID-19 pandemic created unforeseen delays in the questionnaire replies, thus the functional and non-functional requirements presented in this deliverable extracted from a part of questionnaire recipients and from the RAINBOW demonstrators. An updated list of RAINBOW's functional and non-functional requirements will be documented in D1.2.

8 References

- [1] “A Guide to the Business Analysis Body of Knowledge (BABOK Guide) is a standard for the practice of business analysis created and maintained by IIBA (International Institute of Business Analysis),” [Online]. Available: <https://www.iiba.org/standards-and-resources/babok/>.
- [2] D. Pandey, S. U. and A. K. Ramani, “An effective requirement engineering process model for software development and requirements management,” in *International Conference on Advances in Recent Technologies in Communication and Computing, IEEE*, 2010.
- [3] F. Bonomi, R. Milito, J. Zhu and S. Addepalli, “Fog Computing and Its Role in the Internet of Things,” in *Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing*, 2012.
- [4] C. C. Byers, “Architectural Imperatives for Fog Computing: Use Cases, Requirements, and Architectural Techniques for Fog-Enabled IoT Networks,” *IEEE Communications Magazine*, vol. 55, no. 8, 2017.
- [5] OpenFog Consortium Architecture Working Group, “OpenFog Reference Architecture for Fog Computing,” OpenFog Consortium, 2017.
- [6] S. Yi, Z. Hao, Z. Qin and Q. Li, “Fog Computing: Platform and Applications,” in *2015 Third IEEE Workshop on Hot Topics in Web Systems and Technologies (HotWeb)*, 2015.
- [7] Y. Shi, G. Ding, H. Wang, H. E. Roman and S. Lu, “The fog computing service for healthcare,” in *2015 2nd International Symposium on Future Information and Communication Technologies for Ubiquitous HealthCare (Ubi-HealthTech)*, 2015.
- [8] J. Dizdarević, F. Carpio, A. Jukan and X. Masip-Bruin, “A Survey of Communication Protocols for Internet of Things and Related Challenges of Fog and Cloud Computing Integration,” *ACM Comput. Surv.*, 2019.
- [9] Z. Shelby, K. Hartke and C. Bormann, *RFC 7252 – The Constrained Application Protocol (CoAP)*, 2014.
- [10] C. Bormann, S. Lemay, H. Tschofenig, K. Hartke and B. Silverajan, *RFC 8323 - CoAP (Constrained Application Protocol) over TCP, TLS, and WebSockets*, 2018.
- [11] International Standards Organization/IEC JTC 1, *ISO/IEC 20922:2016 Information Technology - Message Queuing Telemetry Transport (MQTT) v3.1.1*, 2016.
- [12] “Open Connectivity Foundation,” [Online]. Available: <https://openconnectivity.org>. [Accessed 04 2020].
- [13] W. Shi, G. Pallis and Z. Xu, “Edge computing,” in *Proceedings of the IEEE*, 2019.
- [14] M. Zwolenski and L. Weatherill, “The digital universe: Rich data and the increasing value of the Internet of Things,” *Austral. J. Telecommun. Digit. Econ.*, vol. 2, no. 3, p. 47, 2014.
- [15] J. Thones, “Microservices,” *IEEE Softw.*, vol. 32, no. 1, p. p. 116, Jan. 2015.
- [16] “Lori MacVittie, Micorservices and Microsegmentation, “<https://devcentral.f5.com/articles/microservices-versus-microsegmentation>.” 2015.”.



- [17] “Martin Fowler, “Microservices a definition of this new architectural term.” [Online]. Available: <https://martinfowler.com/articles/microservices.html>.”.
- [18] “Eric S. Raymond, “The Art of UNIX Programming.” 2013.”.
- [19] D. Trihinas, A. Tryfonos, M. Dikaiakos and G. and Pallis, “Devops as a service: Pushing the boundaries of microservice adoption.”, *IEEE Internet Computing*, pp. pp.65-71, 2018.
- [20] S. M. Fulton, “What Led Amazon to its Own Microservices Architecture,” 2015.
- [21] C. Perkins, E. Belding-Royer and S. Das, “Ad hoc On-Demand Distance Vector (AODV) Routing,” 2003.
- [22] D. Johnson, Y. Hu and D. Maltz, “The Dynamic Source Routing Protocol (DSR),” 2007.
- [23] K. Yang, J.-f. Ma and Z.-h. Miao, “ Hybrid Routing Protocol for Wireless Mesh Network,” 2009.
- [24] A. Neumann, C. Aichele, M. Lindner and S. Wunderlich, “Better approach to mobile ad-hoc networking (BATMAN),” 2008.
- [25] C. J. DeLisle, V. Lorentz, K. Boulain, E. Yilmaz, R. Rankin and R. Mindalev, “CJDNS Whitepaper,” 2019.
- [26] E. F. Brickell, J. Camenisch and L. Chen, “Direct anonymous attestation,” in *ACM Conference on Computer and Communications Security (CCS)*, 2004.
- [27] N. Aaraj, A. Raghunathan, Jha and a. N. K., “Analysis and Design of a Hardware/Software Trusted Platform Module for Embedded Systems,” in *ACM Transactions on Embedded Computing Systems (TECS)*, 2008.
- [28] M. Strasser and H. Stamer, “Software-Based Trusted Platform Module Emulator,” in *International Conference on Trusted Computing*, 2008.
- [29] C. Shepherd, G. Arfaoui, I. Gurulian, R. P. Lee and a. K. Markantonakis, “Secure and Trusted Execution: Past , Present and Future A Critical Review in the Context of the Internet of Things and Cyber-Physical Systems,” in *IEEE Trustcom/BigDataSE/ISPa*, 2016.
- [30] J. Haid and M. Klimke, “Hardware-based Secure Identities for machines in smart factories,” 2016.
- [31] L. Liu, Z. Chang, X. Guo, S. Mao and T. Ristaniemi, “Multiobjective optimization for computation offloading in fog computing,” *IEEE Internet of Things Journal*, 2017.
- [32] Pirker, Martin, Toegl, Ronald, Hein, Daniel, Danner and Peter, A PrivacyCA for Anonymity and Trust., Vols. 101-119. 10.1007/978-3-642-00587-9_7, 2009.
- [33] E. Brickell, J. Camenisch and a. L. Chen, “Direct anonymous attestation,” in *11th ACM conference on Computer and communications security*, Washington DC, USA, 2004.
- [34] J. Whitefield, L. Chen, T. Giannetsos, S. Schneider and H. Treharne, “Privacy-Enhanced Capabilities for VANETs using Direct Anonymous Attestation,” in *IEEE Vehicular Networking Conference (VNC)*, 2017.
- [35] E. Brickell, L. Chen and J. Li, “Simplified security notions of direct anonymous attestation and a concrete scheme from pairings,” in *International Journal of Information Security*, 2009.

- [36] J. Camenisch, L. Chen, M. Drijvers, A. Lehmann, D. Novick and R. Urian, “One TPM to Bind Them All: Fixing TPM 2.0 for Provably Secure Anonymous Attestation,” in *IEEE Symposium on Security and Privacy (S&P)*, 2017.
- [37] J. Camenisch, M. Drijvers and A. Lehmann, “Anonymous Attestation with Subverted TPMs,” in *Advances in Cryptology - CRYPTO*, 2017.
- [38] S. Goldwasser, S. Micali and C. Rackoff, “The knowledge complexity of interactive proof systems,” *SIAM Journal on computing*, 1989.
- [39] K. E. Defrawy, G. Holland and G. Tsudik, “Remote Attestation of Heterogeneous Cyber-Physical Systems: The Automotive Use Case,” in *ESCAR*, 2015.
- [40] R. Sailer, X. Zhang, T. Jaeger and L. v. Doorn, “Design and implementation of a TCG-based Integrity Measurement Architecture,” in *13th USENIX Symposium*, 2004.
- [41] N. Asokan, F. Brasser, A. Ibrahim, A. Sadeghi, M. Schunter, G. Tsudik and C. Waschmann, “SEDA: Scalable Embedded Device Attestation,” in *22nd ACM SIGSAC Conference on Computer and Communications Security*, 2015.
- [42] N. Koutroumpouchos, C. Ntantogian, S. Menesidou, K. Liang, P. Gouvas, C. Xenakis and T. Giannetsos, “Secure edge computing with lightweight control-flow property-based attestation,” in *IEEE Conference on Network Softwarization*, 2019.
- [43] Q. Pu, G. Ananthanarayanan, P. Bodik, S. Kandula, A. Akella, P. Bahl and I. Stoica, “Low latency geo-distributed data analytics,” *ACM SIGCOMM Computer Communication Review*, 2015.
- [44] P. Li, S. Guo, T. Miyazaki, X. Liao, H. Jin, A. Y. Zomaya and K. Wang, “Traffic-aware geo-distributed big data analytics with predictable job completion time,” *IEEE Transactions on Parallel and Distributed Systems*, 2016.
- [45] L. Gu, D. Zeng, S. Guo, Y. Xiang and J. Hu, “A general communication cost optimization framework for big data stream processing in geo-distributed data centers,” *IEEE Transactions on Computers*, 2015.
- [46] M. Anna-Valentini and G. Anastasios, “Bi-objective traffic optimization in geo-distributed data flows,” *Big Data Research*, 2019.
- [47] I. Stanoi, G. Mihaila, T. Palpanas and C. Lang, “Whitewater: Distributed processing of fast streams,” *IEEE transactions on knowledge and data engineering*, 2007.
- [48] P. Pietzuch, J. Ledlie, J. Shneidman, M. Roussopoulos, M. Welsh and M. Seltzer, “Network-aware operator placement for stream-processing systems,” in *22nd International Conference on Data Engineering*, 2006.
- [49] M. Nardelli, V. Cardellini, V. Grassi and F. L. Presti, “Efficient operator placement for distributed data stream processing applications,” *IEEE Transactions on Parallel and Distributed Systems*, 2019.
- [50] A. Foto, D. Shlomi, S. Shantanu and U. J. D., “Meta-MapReduce: A technique for reducing communication in MapReduce computations,” in *Proceedings of the 17th International Symposium on Stabilization, Safety, and Security of Distributed Systems*, 2015.
- [51] A. Rabkin, M. Arye, S. Sen, V. S. Pai and M. J. Freedman, “Aggregation and degradation in jetstream: Streaming analytics in the wide area,” in *11th {USENIX} Symposium on Networked Systems Design and Implementation*, 2014.

- [52] Z. Georgiou, M. Symeonides, D. Trihinas, G. Pallis and M. D. Dikaiakos, “Streamsight: A query-driven framework for streaming analytics in edge computing.,” in *2018 IEEE/ACM 11th International Conference on Utility and Cloud Computing*, 2018.
- [53] B. Heintz, A. Chandra and R. K. Sitaraman, “Trading timeliness and accuracy in geo-distributed streaming analytics.,” in *Proceedings of the Seventh ACM Symposium on Cloud Computing*, 2016.
- [54] P. Patel, M. I. Ali and A. Sheth, “On using the intelligent edge for IoT analytics.,” *IEEE Intelligent Systems*, 2017.
- [55] J. Thalheim, A. Rodrigues, I. E. Akkus, P. Bhatotia, R. Chen, B. Viswanath, L. Jiao and C. Fetzer, “Sieve: Actionable insights from monitored metrics in microservices.,” *arXiv preprint arXiv:1709.06686*, 2017.
- [56] “Accordion,” [Online]. Available: <https://www.accordion-project.eu/>.
- [57] “H-CLOUD,” [Online]. Available: <https://www.h-cloud.eu/>.
- [58] “Morphemic,” [Online]. Available: <http://morphemic.cloud/>.
- [59] “Melodic,” [Online]. Available: <https://melodic.cloud/>.
- [60] “Pledger,” [Online]. Available: <http://www.pledger-project.eu/>.
- [61] “FogProtect,” [Online]. Available: <https://fogprotect.eu/>.
- [62] “SmartClide,” [Online]. Available: <https://smartclide.eu/>.
- [63] “LightKone,” [Online]. Available: <https://www.lightkone.eu/>.
- [64] “mF2C,” [Online]. Available: <https://www.mf2c-project.eu/>.
- [65] “RECAP,” [Online]. Available: <https://recap-project.eu/>.
- [66] “PrEstoCloud,” [Online]. Available: <https://prestocloud-project.eu/>.
- [67] “Ditas,” [Online]. Available: <https://www.ditas-project.eu/>.
- [68] “Fog Guru,” [Online]. Available: <http://www.fogguru.eu/>.
- [69] “Arrow Head,” [Online]. Available: <http://www.arrowheadproject.eu/>.
- [70] “Unicorn,” [Online]. Available: <https://unicorn-project.eu/>.
- [71] “Decenter,” [Online]. Available: <https://www.decenter-project.eu/>.
- [72] “AWS IoT Greengrass,” [Online]. Available: <https://aws.amazon.com/greengrass/>.
- [73] “MindSphere,” [Online]. Available: <https://siemens.mindsphere.io/en>.
- [74] “ioFog,” [Online]. Available: <https://iofog.org/>.
- [75] “Vapor,” [Online]. Available: <https://www.vapor.io/>.
- [76] “Crosser,” [Online]. Available: <https://www.crosser.io/>.
- [77] J. Gedeon, M. Stein, J. Krisztinkovics, P. Felka, K. Keller, C. Meurisch, L. Wang and a. M. Mühlhäuser, “From cell towers to smart street lamps: Placing cloudlets on existing urban infrastructures,” p. pp. 187–202, Oct. 2018.
- [78] J. Steuer, “Defining virtual reality: Dimensions determining telepresence,” *J. Commun.*, vol. 42, no. 4, p. pp. 73–93, Dec. 2010.
- [79] W. Cai, M. Chen and a. V. C. M. Leung, “Toward gaming as a service,” *IEEE Internet Comput.*, vol. 18, no. 3, p. pp. 12–18, May/Jun. 2014.

- [80] S. Howell, Y. Rezgui, J. Hippolyte, B. Jayan and a. H. Li, “Towards the next generation of smart grids: Semantic and holonic multi-agent management of distributed energy resources,” *Renew. Sustain. Energy Rev.*, vol. 77, p. pp. 193–214, Sep 2017.
- [81] Y. Zheng, F. Liu and a. H. Hsieh, “U-Air: When urban air quality inference meets big data,” in *in Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2013.
- [82] T. Nuortio, J. Kytöjoki, H. Niska and a. O. Bräysy, “Improved route planning and scheduling of waste collection and transport,” *Expert Syst. Appl.*, vol. 30, no. 2, pp. pp. 223–232, 2006..
- [83] C. Chung, D. Egan, A. Jain, N. Caruso, C. Misner and a. R. Wallace, “A cloud-based mobile computing applications platform for first responders,” in *in Proc. 7th IEEE Int. Symp. Service-Oriented Syst. (SOSE)*, 2013.
- [84] B. Ghazal, K. ElKhatib, K. Chahine and a. M. Kherfanin, “Smart traffic light control system,” in *Proc. 3rd Int. Conf. Elect., Electron., Comput. Eng. Appl. (EECEA)*, Apr. 2016.
- [85] Laberteaux and H. Hartenstein, “A tutorial survey on vehicular ad hoc networks,” *IEEE Commun. Mag.*, vol. 46, no. 6, p. pp. 164–171, Jun. 2008.
- [86] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson and a. A. Oliveira, “Smart cities and the future Internet: Towards cooperation frameworks for open innovation,” in *Future Internet Future Internet Assembly 2011: Achievements and Technological Promises. Berlin, Germany: Springer*, p. pp. 431–446., 2011.
- [87] L. Atzori, A. Iera and a. G. Morabito, “The Internet of Things: A survey,” *Comput. Netw.*, vol. 54, no. 15, p. pp. 2787–2805, Oct. 2010..
- [88] M. Casini, “Internet of Things for energy efficiency of buildings,” *Int.Sci. J. Archit. Eng.*, vol. 2, no. 1, pp. pp. 24–28, 2014..
- [89] G. Kakamoukas, P. Sarigiannidis, G. Livanos, M. Zervakis, D. Ramnalis and V. Polychronos, “A Multi-collective, IoT-enabled, Adaptive Smart Farming Architecture,” in *2019 IEEE International Conference on Imaging Systems and Techniques (IST). IEEE, 2019..*
- [90] J. Wan, B. Chen, S. Wang, M. Xia, D. Li and a. C. Liu, “Fog computing for energy-aware load balancing and scheduling in smart factory,” *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, p. pp. 4548–4556.
- [91] McElhannon and J. Pan, “Future edge cloud and edge computing for Internet of Things applications,” *IEEE Internet Things J.*, vol. 5, no. 1, p. pp. 439–449, Feb. 2018.
- [92] N. M. Sandar, S. Chaisiri, S. Yongchareon and a. V. Liesaputra, “Cloud-based video monitoring framework: An approach based on software-defined networking for addressing scalability problems,” in *in Proc. Web Inf. Syst. Eng. (WISE) Workshops*, 2014.
- [93] J. Sherman and D. Nafus, “Big data, big questions| this one does not go up to 11: The quantified self movement as an alternative big data practice,” *Int. J. Commun.*, vol. 8, no. 11, p. pp. 1784–1794, 2014.
- [94] M. Zorzi and B. N., “Health care applications: A solution based on the Internet of Things,” in *in Proc. 4th Int. Symp. Appl. Sci. Biomed. Commun. Technol. (ISABEL)*, 2011.



- [95] Z. Gang and W. Honggang, “Connected Health,” *IEEE Internet Computing*, vol. 24, no. 2, pp. 5 - 7, 2020.
- [96] H. Fereidooni, T. Frassetto, M. Miettinen, A. Sadeghi and M. Conti, “Fitness trackers: Fit for health but unfit for security and privacy,” in *in Proc. IEEE/ACM Int. Conf. Connected Health, Appl., Syst. Eng. Technol. (CHASE)*, Jul. 2017.
- [97] ISO/IEC, "ISO/IEC 25010: 2011 Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE)--System and software quality models., (2011).



9 Appendix

30/6/2020

RAINBOW Questionnaire

RAINBOW Questionnaire

The following questionnaire comprises 8 different sections and depending on your responses you will answer only some of them:

- 1 - Edge/fog Computing
- 2 - Introductory Questions
- 3 - Business Questions
- 4 - Application Deployment Modelling
- 5 - Orchestration
- 6 - Data Management and Analytics
- 7 - Secure Remote Asset Management
- 8 - System Configuration Integrity

Estimated completion time: 15 min.

RAINBOW is a platform that simplifies the deployment and management of scalable, heterogeneous and secure IoT services.

With RAINBOW, fog computing can reach its true potential by providing the deployment, orchestration, network fabric and data management for scalable and secure edge applications.

RAINBOW addresses the need to timely process the ever-increasing amount of data continuously gathered from heterogeneous IoT devices and appliances.

More information: <https://rainbow-h2020.eu/rainbow-platform/>

Does your company consider moving into edge/fog computing?

Edge/fog computing optimizes internet devices and web applications by bringing computing and storage closer to the data source. This minimizes the need for long distance communications between client and server, which reduces latency and bandwidth usage.

1. Does your company consider moving into edge/fog computing? *

☐ Yes. We already have

☐ Yes, or probably yes. In the near future

☐ Maybe. Still Evaluating

☐ No or probably not

☐ I don't know what it is

https://docs.google.com/forms/d/1kV3Tg9PT2_-lbVHJCkWZM-9bcCN9aLH5X2kLGOE_cNE/edit?ts=5efa7b8e

1/17



30/6/2020

RAINBOW Questionnaire

Why is your company not considering moving into edge/fog computing?

2. Why is your company not considering moving into edge/fog computing?

Why is your company not considering moving into edge/fog computing?

3. In which sector/market does your company/organisation operate? *

- ☐ Smart Cities
- ☐ Agriculture/Farming
- ☐ Manufacturing/Industry
- ☐ Energy Grids and Critical Infrastructure Grids (water, sewage, power, telco, gas)
- ☐ Logistics and Supply Chain
- ☐ Transport (public – private)
- ☐ Electronics / Hardware provider & OEM
- ☐ IoT & Fog/Cloud Integrators
- ☐ General ICT Developers
- ☐ ICT supportive services (Consulting, Training etc.)
- ☐ Telecom
- ☐ Data Analytics, AI and Machine Learning
- ☐ E-commerce
- ☐ SaaS cloud
- ☐ Financial services
- ☐ Health, e-Health and/or social assistance services
- ☐ Travel/Tourism
- ☐ Education/Learning services
- ☐ Retail trading

Άλλο: ☐ _____



30/6/2020

RAINBOW Questionnaire

4. Which best describes your role in your organisation/company? *

Να επισημαίνεται μόνο μία έλλειψη.

- ☐ Upper Management
- ☐ Lower-Mid Management
- ☐ Chief Technology Operator
- ☐ Chief Architect
- ☐ DevOps Engineer
- ☐ Software Programmer/Designer
- ☐ System/Network Admin
- ☐ IT Support Staff
- ☐ Graphics Designer
- ☐ Data/Business Analyst
- ☐ Marketing / Sales
- ☐ Άλλο: _____

5. Please describe your company / organization. *

	1-15	16-50	51-99	100+
Company Size (employees):	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

6. Please describe your company / organization. *

	1-5	6-15	15-49	50+
IT department (employees):	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>



30/6/2020

RAINBOW Questionnaire

7. Please describe your company / organization. *

	0-5	5-10	11-19	20+
Years in Business:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

8. Please describe your company / organization. *

	1	2-5	6-10	11+
Countries of Operation:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

9. How would you describe your knowledge / previous experience in the following key technologies of RAINBOW? *

	Only Heard it exists	Basic- Medium knowledge	Solid understanding (theory and little practice)	Strong working experience on actual projects
Fog / Edge Computing	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Microservices	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Big data processing	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Containers	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Containers Orchestration (e.g., Kubernetes)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
IT Security & Privacy	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>



30/6/2020

RAINBOW Questionnaire

10. Please, provide a short description of the applications/services your company is currently offering and key technologies used. (A few lines are enough)

Business Questions

11. Does your company face at least one of these challenges as it strives to offer its services? *

- ☐ Multiple machine protocols in interconnected systems
- ☐ Heterogeneous input data format
- ☐ Cloud connectivity restrictions
- ☐ Waste of cloud resources because of junk data
- ☐ Data reformatting
- ☐ Data buffering
- ☐ Delay intolerant applications
- ☐ Bandwidth restrictions
- ☐ Data access restrictions

Άλλο: ☐ _____



30/6/2020

RAINBOW Questionnaire

12. When considering adopting an IT platform (like RAINBOW on edge/fog computing), potential customers and business users pay attention to key Features before “embracing” it. Please select the 3 most important among the features below in your opinion. *

- ☐ Easy and friendly to use and also to modify
- ☐ Independent of existing technologies (no vendor lock-in)
- ☐ Compliant with regulations, standards, etc. (Bureaucracy friendly)
- ☐ Dependable, robust, no breakdowns, good 24/7 support
- ☐ Complemented by expert support, helpdesk and consulting
- ☐ Extensive training and documentation material
- ☐ Can run remotely on demand (as a Service)
- ☐ Respecting privacy, security, sensitive and confidential data
- ☐ Scalable. Can support huge datasets, instances, probes, sensors etc.
- ☐ Extendable and able to host an ecosystem of customized applications and code
- ☐ Data and Bandwidth Efficiency

13. Which of the following stakeholders would be most probable to persuade you to adopt a platform like RAINBOW on edge/fog computing? Select the 3 most important. *

- ☐ Internal IT department
- ☐ Other Internal Departments
- ☐ External ICT/IoT integrator
- ☐ Existing Vendors / Suppliers
- ☐ Existing Customers
- ☐ Existing Collaborators and Partners
- ☐ Research Centres / Universities
- ☐ Consortiums, Clusters or similar joint "collaborations"
- ☐ Advertising and Marketing
- ☐ Governmental / EC policy and legal bodies



6/30/2020

RAINBOW Questionnaire

14. If you were to provide data to a platform like RAINBOW on edge/fog computing, their sources would probably be: *

- ☐ Humans (health parameters, location etc)
- ☐ Mobile devices (phones, tablets, AR/VR etc.)
- ☐ IoT devices and sensors (industrial, smart building, agriculture, etc.)
- ☐ Infrastructure (smart grids, transportation, energy etc)

Άλλο: ☐ _____

15. How do you envision that you would potentially use the RAINBOW platform? *

- ☐ As a Business end-user (No coding. Only use ready services developed by others.)
- ☐ As a Service Operator (does not add new features to RAINBOW. Only parametrizes)
- ☐ As a Service Developer (Develop applications using Rainbows API)
- ☐ As a Rainbow Developer (Contributor. Develops and enriches/adds new features to RAINBOW platform)
- ☐ As an Infrastructure Provider (Offers cloud or fog resources e.g., on-demand VMs)

Application Deployment Modelling

Application deployment modeling is a formal way for describing an application topology. Within the topology description users can denote the intercommunication and dependencies among application services and annotate their description with constraints, optimization policies and QoS requirements.



30/6/2020

RAINBOW Questionnaire

16. How do you describe your cloud/fog application topology? Do you use any formal notation or technology? (Tick all that apply) *

- ☐ Kubernetes
- ☐ Docker-Compose
- ☐ TOSCA
- ☐ UML
- ☐ Eclipse MicroProfile
- ☐ Istio
- ☐ Ansible
- ☐ Rancher
- ☐ Puppet
- ☐ Nomad

Άλλο: ☐ _____

17. What type of policies (i.e., QoS and performance requirements) are necessary for your Fog application that you currently use or expect to include in the application topology description? (Tick all that apply) *

- ☐ Data policies (e.g., restrict access, sharing data)
- ☐ Application performance policies (e.g., response time)
- ☐ Resource utilization policies (e.g., cpu usage, memory capacity)
- ☐ Network resources policies (e.g., limit bandwidth)
- ☐ Cost policies (e.g., cap the hourly cost)
- ☐ Security policies
- ☐ Not applicable

Άλλο: ☐ _____



30/6/2020

RAINBOW Questionnaire

18. Currently, how do you measure performance and QoS of your Fog application? *

- ☐ Developed in-house tools
- ☐ Use of existing monitoring solutions (e.g., Prometheus, netdata, etc.)
- ☐ Integrated in-house tools on existing monitoring solutions.
- ☐ No we don't have a monitoring solution
- ☐ Άλλο: _____

19. How do you expect to specify the performance requirements and policies for a fog service, which will be used to decide how the service will be deployed? (Tick all that apply) *

- ☐ Graphical User Interface
- ☐ Domain Specific Language
- ☐ Code annotations (e.g., stereotypes within UML diagrams)

Άλλο: ☐ _____

20. What challenges do you face when describing QoS and performance policies of your application ? (Tick all that apply) *

- ☐ Lack of tools for describing QoS and performance policies
- ☐ Lack of technical expertise (Learning curve)
- ☐ Lack of time
- ☐ No specific challenge
- ☐ Not applicable

Άλλο: ☐ _____

Orchestration

The RAINBOW orchestrator will be responsible for selecting the nodes, on which a deployed application will execute, based on the requirements specified in the service graph. Furthermore, it will monitor the running applications and make necessary adjustments to meet the specified QoS goals.



30/6/2020

RAINBOW Questionnaire

21. What parts of your Fog/Edge application do you monitor? *

- ☐ All application services
- ☐ Only critical services of the application
- ☐ We don't have a monitoring solution
- ☐ Άλλο: _____

22. What is being measured by your monitoring solution about your Fog/Edge application? (Tick all that apply) *

- ☐ Infrastructure Resource Utilization (e.g., CPU usage, memory, storage, network)
- ☐ Response time, network latency
- ☐ Application-specific metrics
- ☐ Application errors and logs
- ☐ We don't have a monitoring solution

Άλλο: ☐ _____

23. If you have a monitoring solution, please provide a brief description about what is or what will be measured by your monitoring solution about your Fog/Edge application?



30/6/2020

RAINBOW Questionnaire

24. If the desired performance or QoS of your fog/edge application is not met, how are reconfiguration/scaling decisions taken? *

- ☐ Application identifies the problem and notifies developers to take an action
- ☐ Application is self-configured (e.g., auto scaling actions)
- ☐ No reconfiguration/scaling

Άλλο: ☐ _____

25. How do you run your services in the cloud? (Tick all that apply). *

- ☐ Virtualization (VMs)
- ☐ Containerization (Docker)
- ☐ Bare metal
- ☐ None

Άλλο: ☐ _____

26. How do you run your services on fog nodes? (Tick all that apply) *

- ☐ Virtualization (VMs)
- ☐ Containerization (Docker)
- ☐ Bare Metal
- ☐ None

Data
Management
and
Analytics

Analytics services help businesses convert their historical and real-time data into actionable insights for data-driven decisions. In general, they consist of services for ingesting, storing, processing, and accessing the data created by applications, devices and users.



30/6/2020

RAINBOW Questionnaire

27. What analytics do you extract (or consider to extract) from your application ?
What is the type of these analytics? *

- ☐ Descriptive/Summary statistics
- ☐ Video/image Analysis
- ☐ Machine Learning
- ☐ Neural Networks
- ☐ Graph Analytics

Άλλο: ☐ _____

28. How are your analytics being used? (Tick all that apply) *

- ☐ For online decision-making (immediate actions)
- ☐ For offline decision-making
- ☐ Not applicable

29. What analytics frameworks are currently used, or you consider to use in the future? (Tick all that apply) *

- ☐ Tensorflow/keras
- ☐ Spark ecosystem
- ☐ Flink ecosystem
- ☐ Hadoop ecosystem
- ☐ Python Conda ecosystem
- ☐ R ecosystem

Άλλο: ☐ _____



30/6/2020

RAINBOW Questionnaire

30. What are the major barriers preventing you from extracting and processing your application's analytics? *

- ☐ Lack of tools for data extraction
- ☐ Lack of tools for data processing
- ☐ Lack of technical expertise
- ☐ Lack of time
- ☐ Privacy Restrictions
- ☐ Size of data
- ☐ Variety of data
- ☐ Not Applicable

Άλλο: ☐ _____

31. How is your data ingested into your analytics processing framework? *

- ☐ Directly from sources (streamed each data point to the processing engine)
- ☐ Batch processing (from a data storage to the processing engine)
- ☐ Not applicable

32. Which type of data storage solutions do you use at your company? *

- ☐ SQL (postgres,mysql etc)
- ☐ Document based (mongodb, elastic etc)
- ☐ HDFS files
- ☐ Time-series databases
- ☐ Key-value stores (Redis, memcached)

Άλλο: ☐ _____

30/6/2020

RAINBOW Questionnaire

33. What communication protocol(s)/technologies between fog nodes and IoT are used to exchange data? *

- ☐ MQTT
- ☐ HTTP
- ☐ TCP/UDP
- ☐ RPC
- ☐ Kafka Messages
- ☐ RabbitMQ
- ☐ Not applicable

Άλλο: ☐ _____

Secure
Remote
Asset
Management

Secure remote asset management refers to the deployment of sufficient services for the real-time supervision, monitoring, and management over the security of the information and functionalities about deployed assets; being edge devices, fog nodes or service graphs deployed in the cloud. Access to machines and remote edge devices, edge addition (and/or removal), secure re-programming and/or reconfiguration, etc. must be managed to protect the security posture of the provided services and the underlying infrastructure.

34. Is Secure Remote Asset Management, of deployed devices (i.e., sensors, routers, switches. etc ?), a requirement for your services ? If so what are the security requirements of interest (e.g., Secure Re-Programming, Secure Reconfiguration, Firmware Updates, etc.)? *

35. Are you leveraging any specific tools for Remote Asset Management ? Do they provide automatic re-configuration or manual intervention if required from the system administrator ? *

- ☐ CISCO Live
- ☐ DIGI Remote Manager
- ☐ Nebbiolo Fog Computing Platform
- ☐ APAT
- ☐ Άλλο: _____



30/6/2020

RAINBOW Questionnaire

36. Zero touch provisioning removes the need for all manual intervention in the provisioning, testing and activation of demarcation equipment, thus, zeroing the deployment and configuration management of the network edge. Do you see the use of network edge devices featuring zero touch provisioning as an important enabler for your provided services ? *

System
Configuration
Integrity

RAINBOW envisions to deal with a diverse set of mixed-criticality applications (e.g., safety-critical, hard and soft real-time applications, and non-critical best-effort applications) distributed over multiple resources, i.e., CPS (edge) devices, high-end computing resources hosting multiple apps of mixed-criticalities, or in the cloud. In such environments, runtime system integrity, correctness and operational assurance are fundamental security aspects especially against a wide attack landscape. Configuration Integrity Verification refers to mechanisms and tools that enable to assess and preserve the integrity of the deployed applications and services, during both deployment and run-time, so that they can assess the trusted state of the host devices.

37. What are the core security and trust requirements that need to be met in your services? *

- ☐ Confidentiality, Integrity, Authentication
- ☐ Network Security and Access Control
- ☐ Secure Data Storage
- ☐ Data Privacy, User Privacy, Location Privacy (please underline the one that is most relevant)
- ☐ Operational Assurance
- ☐ Functional Safety
- ☐ All of the above

38. What type of mixed-criticality functions comprise your provided services? Are there any safety-critical functions with strict safety requirements in terms on time dependencies, synchronisation, etc. ? *



30/6/2020

RAINBOW Questionnaire

39. For any safety-critical functions, running at the edge, do you envision any requirements regarding « verifiable computing » ; i.e., offload the computation of a function to other (perhaps) untrusted fog nodes, while maintaining verifiable results ? (If YES, please justify your answer with examples of safety-critical functions)

40. Are you employing any tools or mechanisms for secure (shared) and private data computation at the edge? If so, are they software- or hardware-based?

41. The concept of Trusted Computing and Remote Attestation has been proposed for detecting unauthorized changes to the configuration and execution of a remote fog/edge device through either hardware enhancements or software modifications. In your services, what type of (possibly heterogeneous) devices (e.g., sensors/actuators, PLCs, etc.) are being used and what are their characteristics (i.e., processing power, multi-core, etc.) ?



30/6/2020

RAINBOW Questionnaire

42. Would it be acceptable to add specialized hardware to the deployed devices for enhancing their security posture ? What impact would this have on resources, constraints, application requirements ? (Please justify your answer - a few lines would suffice) *

End of
Questionnaire -
thank you for
participating!

Thank you very much for taking the time to complete the questionnaire.
Your response is highly valued. In the next question form, you may
optionally provide an email address.

43. Email address (optional):
