D4.2 Non-functional properties analysis and constraints specification

Version 1.0

Document Information

Contract Number	825473
Project Website	https://elastic-project.eu/
Contractual Deadline	M6, May 2019
Dissemination Level	PU
Nature	R
Author(s)	ISEP
Contributor(s)	IKL, TRT
Reviewer(s)	IKL, THALIT, GEST, FLO

Notices: *The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No "825473".*

Change Log

Version	Author	Description of Change
V0.1	ISEP	Initial Draft
V0.5	ISEP, IKL	Second Draft, including the contributions from ISEP and IKL
V0.6	ISEP, IKL, TRT	Third Draft, including the contributions from ISEP, IKL and TRT
V0.7	ISEP, IKL, TRT	Final Version, including revision
V1.0	BSC	Release for submission to EC
		(Final Change Log entries reserved for releases to the EC)

Table of contents

1 Executive Summary

This deliverable covers the work done during the first phase of the project within WP4. The deliverable spans 6 months of work and handles the work done in Task 4.1 " *Non-Functional System Properties*" to reach milestone MS1.

Concretely, this document provides the analyses and consolidation of the requirements related to non-functional use-case properties, i.e., time, energy, communication quality and security. This consolidation includes both the requirements emanating from the use case requirements extraction from task 1.1, as well as from related application domains (smart manufacturing, avionics and automotive).

To enable evaluation of the project work, this document will also specify the set of concrete criteria and metrics for evaluation.

The analysis of these requirements enables to identify the technical constraints imposed to the software architecture, providing both the specification for the software components to be developed in tasks 4.2 and 4.3, as well as identifying where these will be required in the overall software architecture.

The first milestone of Task 4.1 has been carried out successfully and all objectives of MS1 have been reached and documented in this deliverable.

2 Introduction

ELASTIC addresses the challenge of extreme-scale analytics, considering the necessity of fulfilling the non-functional properties inherited from the system, such as real-time, energy efficiency, communication quality or security. In a smart system, such as smart cities, large volumes of data are collected from distributed sensors, transformed, processed and analysed, through a range of hardware and software stages conforming the so-called compute continuum, i.e., from the physical world sensors (commonly referred to as edge computing), to the analytics back-bone in the data-centres (commonly referred to as cloud computing).

This complex and heterogeneous layout presents several challenges, of which an important one refers to non-functional properties inherited from the application domain including real-time, energy-efficiency, quality of communications and security:

- Real-time data analytics is becoming a main pillar in industrial and societal ecosystems. The combination of different data sources and prediction models within real-time control loops, will have an unprecedented impact in domains such as smart cities. Unfortunately, the use of remote cloud technologies makes infeasible to provide real-time guarantees due to the large and unpredictable communication costs on cloud environments.
- Mobility shows even increased trade-offs and technological difficulties. Mobile devices are largely constrained by the access to energy, as well as suffering from unstable communication, which may increase random communication delays, unstable data throughput, loss of data and temporal unavailability.
- Security is a continuously growing priority for organization of any size, as it affects data integrity, confidentiality and potentially impacting safety too. However, strict security policy management may hinder the communication

among services and applications, shrinking overall performance and real-time guarantees.

Overall, while processing time and energetic cost of computation is reduced as data analytics is moved to the cloud, the end-to-end communication delay and the performance of the system (in terms of latency) increases and becomes unpredictable, making not possible to derive real-time guarantees. Moreover, as computation is moved to the cloud, the required level of security increases to minimise potential attacks, which may end up affecting the safety assurance levels, hindering the execution and data exchange among edge and cloud resources.

It is thus necessary that the ELASTIC architecture includes mechanisms which allow the specification of the required level of non-functional properties (denoted QoS parameters), the offline analysis of these parameters to determine an appropriate system configuration which enables their fulfilment, and an online monitoring and analysis capability which is able to trigger configuration changes upon detection of level violations.

2.1 Purpose and objectives

The purpose of this document is to analyse the non-functional system properties and technical constrains imposed to the ELASTIC software architecture, both from the requirements emanating from the ELASTIC use cases, as well as from related application domains (smart manufacturing, avionics and automotive).

The specific objectives of this document are:

1) Analyse the use cases of the ELASTIC project and related application domains, in terms of non-functional requirements, namely time, energy, communication and security.
2) List the relevant non-functional requirements which need to be addressed in the ELASTIC software architecture, associated with the evaluation metrics and means of verification.
3) Identify the technical constraints imposed to the ELASTIC software architecture in order to provide the required non-functional properties to executed services.

2.2 Relationship with other WPs

Deliverable	Task	Relation
D1.1	T1.1	This document receives inputs from the description of the use-cases and the related requirements.
D2.5	T2.1	This document receives inputs from the requirements of the data analytics platform.
D3.1	T3.1	This document receives inputs, and provides outputs, to the Software Architecture.
D5.1	T5.1	This document receives inputs, and provides outputs, to the Fog Computing Ecosystem.

Table 1. Relationship with other WPs

2.3 Document structure

This document is organized in 7 sections:

- Section 1 provides an Executive Summary of the document
- Section 2 introduces briefly the context and gives a main view of the structure of the document.
- Section 3 gives a general overview of the applications that drive the non-functional requirements of the ELASTIC architecture, starting with a summary of the ELASTIC use cases as defined in WP1, and related applications in the domains of smart manufacturing, automotive and avionics.
- Section 4 provides the detailed non-functional requirements identified for the ELASTIC architecture, including the criteria and means for verification.
- Section 5 describes the main components of the ELASTIC architecture related to guaranteeing non-functional properties quality of service, and the constraints that these impose in the other components of the ELASTIC architecture.
- Section 6 gives a brief description of how the performed work relates to the project activities.
- Section 7 provides a summary and conclusion of the document.

3 Overview of applications with elasticity requirements

Elasticity aims at matching the amount of resources allocated to a service with the amount of resources it actually requires, avoiding over- or under-provisioning. Elasticity is a defining characteristic that differentiates cloud computing from previously proposed computing paradigms, such as grid computing. This section presents the motivations, concepts, typical elasticity patterns, and cost consideration of elastic applications, considering the project’s use cases and related application domains.

3.1 ELASTIC Use-cases

The ELASTIC use-cases will be deployed within the Florence city tram service. In this context, three specific use cases with elasticity requirements were defined:

- positioning and obstacle detection;
- predictive maintenance and energy consumption; and
- interaction between the public and private transport.

This section summarises these use cases, which are described in detail in D1.1 [1].

In order to gather the non-functional requirements specific to each use case, a questionnaire (Annex A) was developed in WP4, and used by use case providers to guide the requirement identification process.

3.1.1 Positioning and obstacle detection

This use case considers two different applications to increase the safety of the operations of the tramway system: autonomous localisation of a tram vehicle and obstacle detection to assist the vehicle driver.

The safe operation of tramway systems traditionally relies on the ability of individually localising the circulating stock. While traditional solutions adopt sensors along the tracks to detect the position of tram vehicles, the Next Generation Autonomous Positioning (NGAP) application enables the tram vehicles to autonomously localise themselves, using sensors (e.g. GPS receivers, accelerometers, RADAR) integrated in a computing system onboard the vehicle. The generated information is transmitted to other devices connected on the cloud for use of the tramway control system.

At the same time, the current trends to increase the safety of circulating vehicles is to assist the driver with systems that automatically detect and alert for potential hazard situations. The Advanced Driving Assistant System (ADAS) concerns the obstacle detection and the collision avoidance techniques. Again, this system comprises specific sensors (e.g. video cameras, RADAR, LIDAR) integrated in a computing system to detect obstacles, determine the potential hazard and alert the driver or even take automatic actions to avoid collision. Information about detected events may be useful on a broader perspective and transmitted to the cloud.

These systems have either local (i.e. at the edge node) or end-to-end (i.e. edge-to-cloud) requirements that define the boundaries of their useful operation. Local timing requirements can be critical (e.g. to timely brake the tram vehicle in order to avoid a collision), but also can be end-to-end timing requirements (e.g. informing the control centre about the current localisation of the tram vehicle). Communication requirements have also an important role when these systems are connected and interact with the cloud. These requirements are fundamental to ensure the performance on the edge/cloud environment, as bandwidth, reliability and security have also impact on the timing and system integrity requirements.

3.1.2 Predictive maintenance and energy consumption

This use case consists of two different applications: monitor rail track to support its predictive maintenance and monitor tram vehicle energy consumption profiles.

Defects on the rail track have an impact on the performance and operations of the tram system. The early detection of these defects can reduce the impact of maintenance interventions on the rail tracks. Monitoring the rail tracks comprises specific sensors (e.g. video camera, RADAR, LASER) that will acquire significant amounts of data. These data must be processed and transmitted to the maintenance management teams, to help planning interventions on the track.

The activity of monitoring of the tram vehicles energy consumption is expected to provide means to potentially improve the electricity energy saving. This will be quite relevant if sections of the rail track will be catenary-free, and thus vehicles will have to rely on battery autonomy. This activity may also generate considerable amounts of data that must be processed and delivered so it can be analysed.

The expected large amounts of data generated by these use cases raises the relevance of timing and communication requirements. Processing data on the edge potentially reduces the amount of data that has to be delivered to the cloud; however, the cloud has more computing power to perform data analysis, which impacts on the communication requirements.

3.1.3 Interaction between the public and private transport

The public and the private transport share a common space and consequently interactions between them necessarily occur. The coexistence of both transports in shared spaces is normally regulated by traffic signs and traffic lights. However, failing to follow these regulations potentially leads to accidents that are threatening to pedestrians or passengers of cars and trams; additionally, it causes a negative impact on the performance of the transportation network.

Monitoring the interaction between the public and the private transport can potentially increase the safety of both pedestrians and public/private passengers, as traffic can be regulated in real-time, in ways to reduce the risk of accidents. The city of Florence has access to a wide variety of intelligent transportation systems (ITS) that allow to monitor and control the public and the private traffic. Data is thus acquired in a distributed way. These data can be used at the edge level (e.g. to prevent a potential hazard) or at the cloud level (e.g. to adapt the strategy management). Consequently, the timing requirements will depend on the response time required for the expected reaction.

Making a coordinated use of the acquired data requires processing, storage and dispatching of traffic data and events. Communication requirements are fundamental to establish where data should be processed.

3.2 Related application domains

3.2.1 Smart Manufacturing

Smart manufacturing is a broad concept. It is a combination of various technologies and solutions that collectively, if implemented in a manufacturing ecosystem, help in optimizing the entire manufacturing process and thus increasing the overall profits.

The European Commission DG Communications Networks, Content & Technology defined Smart Manufacturing as real-time workflow application systems assembled from selected data management, modelling, analysis, display, and interface application, in which all information is available when it is needed, where it is needed, and in the form it is most useful, enabling infusion and integration of network based data and information throughout the lifecycle of design, engineering, planning, and production [2].

Figure 1 below shows the current classical approach to industrial automation, which is deemed by industrial key players and RTD experts to be inadequate to cope with the current manufacturing trends and the needs to consequently evolve [3]. This leads towards new ICT architectures that make possible a transparent integration of the elements of control, communication and processing.

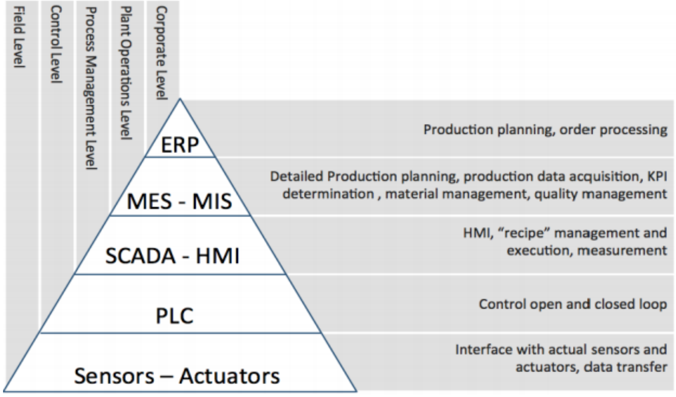

Field Level
Control Level
Process Management Level
Plant Operations Level
Corporate Level
ERP
MES - MIS
SCADA - HMI
PLC
Sensors – Actuators
Production planning, order processing
Detailed Production planning, production data acquisition, KPI determination , material management, quality management
HMI, "recipe" management and execution, measurement
Control open and closed loop
Interface with actual sensors and actuators, data transfer

Figure 1: Traditional automation pyramid [3]

Examples of Smart Manufacturing applications in both process and discrete manufacturing are as follows:

- Real time tracking of supply chain/warehouse management processes such as demand management, order fulfilment, manufacturing flow management and return management.
- Maintenance of the production line: real-time control of performance, durability, and safety of the products they produce.
- Manufacturing Operations removal of manual errors and mitigation of the risk associated with major quality issues.
- Production & Asset Management
- Real-time access to inventory, production and shipment data
- Traceability & Logistic
- Etc.

Making manufacturing smart implies that on one hand all devices in factory have to become smart, or smarter, and furthermore that they need to cooperate in order to provide smart functionalities. The development of such smart manufacturing environment depends on the Internet of Things (IoT) capabilities in an industrial context and demands a better integration between IoT and cloud domain.

The digital environment must be able to handle several data streams from different inputs (such as sensors and actuators, human input, etc.) and be able to store these data into local or distributed cloud environments that are able to communicate effectively with each other. This applies to data generated in the asset itself as well as data from multiple assets combined.

Industrial internet of things (IIoT) is nothing but an ecosystem where every device, machine and/or process is connected through data communication systems. Each machine and piece of industrial equipment is embedded or connected with sensors which typically generate the relevant data. This is further transferred to the cloud/software systems through data communication systems. This huge amount of data has a lot of insights that, if analysed, may help in identifying certain dark areas within the production process. After the analysis of the data, feedback is sent to the production systems for any corrective action.

At EU-level, the biggest initiative that combines “IoT & manufacturing” is the Factories of the Future Public Private Partnership (PPP), launched in 2008/9 as part of the Economic Recovery Package. Projects and results stemming from the FoF PPP can all be accessed through the EFFRA Innovation Portal.

The Factories of the Future PPP identified a set of research priorities along the following research and innovation domains:

- Advanced manufacturing processes
- Adaptive and smart manufacturing systems
- Digital, virtual and resource-efficient factories
- Collaborative and mobile enterprises
- Human-centred manufacturing
- Customer-focused manufacturing

As those priorities are, to a very large extent, related to improvements in embedded system technologies, ARTEMIS took them as input to identify improvements in the following areas of the “Embedded Systems for manufacturing and process automation” subprogram [4]:

- Instant access to virtual dynamic factory

- Increased information transparency between field devices and Enterprise Resource Planning (ERP) systems
- Real-time sensing and networking in challenging automation environments
- Management of critical knowledge to support maintenance decision making
- Automated orchestration of flexible production and distributed manufacturing
- Automation system security and safety
- Tool and methodologies supporting automation system design, engineering, deployment, operation and maintenance.

Many industrial scenarios require deterministic communication. Factory automation (e.g., motion control) demands deterministic ultralow latency transmission in the order of sub-milliseconds. For those scenarios, time-sensitive protocols and technologies with low jitter should be used to ensure applications' integrity and predictable system performance [5].

The architecture the Smart Manufacturing is built upon has to be also capable of supporting closed loop control capabilities for low latency automation functionalities. Closed loop control can only achieve high quality results if the underlying infrastructure can provide appropriate Quality of Service (QoS), especially in relation to high timing constraints scenarios (for both time-critical and time-sensitive automation functionalities). It must be able to distribute from soft real-time (or time-sensitive) to hard real-time (time-critical) data services. Robust hard real time is not possible over open Internet; it needs real time capable physical and transport layers such as Industrial Ethernet, TTTech, time slotted 802.15.4, etc., and the use of local clouds or the concept of fog/edge architectures, where the automation is local. Those local clouds provide a protective security fence that protect sensitive automation operations, such as real time closed control loops and safety critical operations. Hence, edge/cloud network should guarantee the required latency and jitter between layers and lossless networks; the edge/cloud-to-field network should guarantee the quality of service from layer 3 and above, by applying deterministic IP or deterministic networking technologies. Typically edge nodes will be deployed in the plant to realize the most time-critical (<1ms) control functions [5].

From the communications reliability point of view, both mission-sensitive and mission-critical flows distribution must be addressed. The management of different QoS parameters is essential for those automation scenarios:

- End-to-end delay – hard/soft real-time guarantees
- Data bandwidth
- Communication semantics – delivery guarantees, and message ordering
- Message prioritization
- Local device parameters – on device application scheduling
- Service configuration parameters – buffer size, middleware parameters, and prioritization of requests
- Reliability degree
- Fault tolerant communications
- Allowed error rate, etc

The highly automated production and logistics demand industrial-purpose sensors with the adoption of low-cost technologies that meet demanding requirements with respect to safety and reliability. In future production environments, embedded sensor devices with very limited resources must implement complex, distributed, ad-hoc networking protocols. Environmental conditions for field devices are generally harsh, with extreme temperatures, humidity and even corrosive materials. These environmental conditions, as well as the limited access to the areas of interest, makes the integration of sensors a particularly complicated task. Supply power is not always available, and the devices need to run on local power such as batteries. In this scenario of real time sensing and networking in challenging environments, the energy management of sensor and actuator systems is key.

Since smart factories are built around data, cyber security, above all, plays an important and significant role in the entire ecosystem of smart manufacturing. ENISA, the European Union Agency for Network and Information Security, developed a study on Good Practices for Security of the IoT in the context of Industry 4.0 and Smart Manufacturing [6]. It was released in November 2018 and can be considered as reference for security issues within Smart Manufacturing domain. The study identifies risks and attack scenarios, lists security measures related to the use of IoT in Smart Manufacturing and Industry 4.0, and maps them against the aforementioned threats. These security measures and recommendations can be used as a checklist against which to examine the Industry 4.0 security solutions. They will be taken into consideration when defining ELASTIC's security requirements.

The study develops security measures for IoT in Smart Manufacturing, which were organized into three main groups:

- Policies: mostly refers to policies and procedures that should be established within organisations to help ensure a good level of cybersecurity, especially where IIoT solutions are concerned. Privacy issues in the context of manufacturers have also been covered.
- Organizational practices: they explain how Smart Manufacturing should operate, which organisational rules and responsibilities they should establish and follow and which approach they should adopt towards their employees and third-party contractors to effectively handle cybersecurity incidents, manage vulnerabilities, and ensure security of IIoT solutions throughout their lifecycle.

Every group gathers several security recommendations (more than 100 overall). An overview is shown in Figure 2. To know the details of each one of them, go to the document [6].

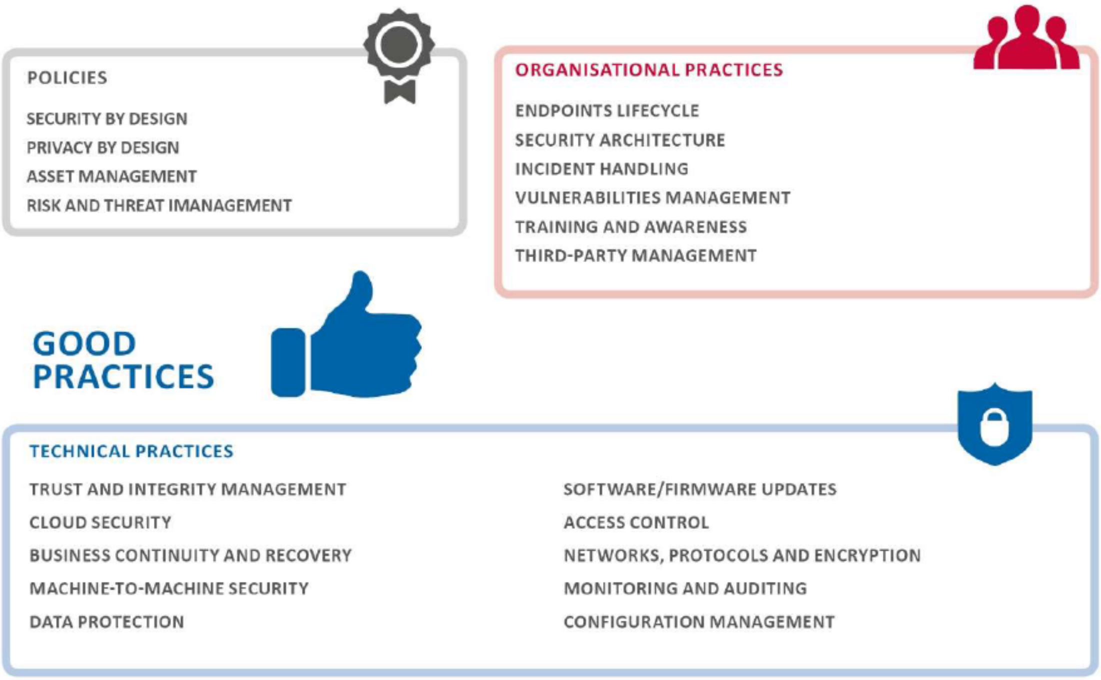


Figure 2: Good practices overview [6]

If we focus on the concept of elasticity in Smart Manufacturing, there are not many references in the bibliographic searches that have been carried out. One of the few projects that has dealt with it is CREMA [7], in which IKL has actively participated. CREMA took into account that the future manufacturing processes are changing and that they need to be highly flexible and dynamic in order to satisfy customer demands for it, e.g. large series production, mass customization, or changing orders. The Cloud Manufacturing has been analysed. The manufacturers virtualize their single services of manufacturing processes from distributed locations in the cloud and it is as if the complete manufacturing were carried out on the same shop floor. An adaptive elastic implementation of manufacturing process management should consider aspects such as QoS and Service Level Agreement (SLA) metrics, and perform an optimization and a runtime adjustment of infrastructural components.

Summarizing, it can be concluded that, as said at the beginning of this section, Smart manufacturing is a broad concept; that all the above mentioned technologies are at the core of Smart Manufacturing (networked sensors, data interoperability, multi-scale dynamic modelling and simulation, intelligent automation, flexibility, scalability, synchronization, integrated performance metrics, multi-level cyber security…), which collectively enable effective integration of the increasingly complex components that make up modern manufacturing systems.

3.2.2 Avionics

As a reminder, avionics deal about all the electronical systems embedded on an aircraft. For years, Avionics systems have been designed using a federated architecture that means one platform embedding a single application. With the introduction of the A380 from Airbus or B787 for Boeing, Avionics Systems have changed to a distributed computing architecture, based on Integrated Modular Avionics (IMA) [8].

3.2.2.1 IMA architecture

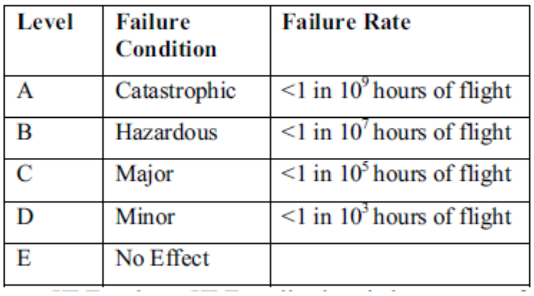
The IMA concept was introduced in Europe through the European funded research projects (FP4/FP5) PAMELA, NEVADA and VICTORIA [9]. This computing architecture was developed for the latest aircraft generations. It defines a set of requirements that are mandatory to ensure Reliability, Availability, Maintainability and Safety (RAMS) during the lifetime of an avionic system, allowing one processing module to host more than one application. This architecture enables to reduce the number of avionics computing modules and then reduces the Size, Weight and Power(SWaP) of the complete avionic platform, thus augmenting fuel consumption efficiency. The IMA architecture introduces two breakthroughs, the first one taking benefit of the increasing performance of Commercial Off-The-Shelf (COTS) mono-core processors, to increase the capacity of hosting many applications on the same computing module, and the second one is the introduction of a network centric architecture where the avionics applications, hosted on the computing modules, are interconnected and communicate through a deterministic Avionics Data Network.

Avionics architectures must ensure to the application a total computing environment isolation and prevent fault propagation. This property has been formalized as robust partitioning in literature and it is a mandatory requirement of IMA. It has been implemented, through spatial and temporal isolation of hardware resources: the applications are enclosed into one or more partitions which have their own memory space, their own hardware resources and their own execution slot. Nowadays, the concept is detailed in a set of standards containing high-level requirements (e.g. DO 297 [10], DO 178-B/C [11][12], DO 254 [13], ARP4754 [14]). Some of them are refined in implementation guidelines (e.g. ARINC 653 [15], ARINC 651 [16]).

3.2.2.2 IMA Computing Module

The adoption of the IMA involves the achievement of a computing module architecture where the spatial and temporal partitioning of shared resources (memory, communication data bus, CPU) must be fully guaranteed for each application. This required the creation of a configuration process to allow the allocation and sharing of all shared resources between all applications and this required the creation of the mechanism to guarantee the access time to their hardware resources and to guarantee the processing time to compute their data.

Firstly, the applications hosted on the avionics module must be classified in conformity with the different Design Assurance Levels (Table 2).



Level	Failure Condition	Failure Rate
A	Catastrophic	<1 in 10^9 hours of flight
B	Hazardous	<1 in 10^7 hours of flight
C	Major	<1 in 10^5 hours of flight
D	Minor	<1 in 10^3 hours of flight
E	No Effect	

Table 2. Software Levels' Failure Rates

So, critical and non-critical applications must be classified according to the five criticality levels. Level A is the most critical level for an avionic critical application; an anomalous behaviour would cause or contribute to a failure of system function which would prevent continued safe flight and landing and maybe the source of crash. Furthermore, according to the DO-178B/C, a complete Worst-Case Execution

Time (WCET) analysis is required for the certification when developing any avionics application and it is required for all hardware and software services offered by the execution platform must be time bounded.

So, as a result, the IMA computing module approach must instantiate several applications with different criticality level on a single hardware platform through an ARINC 653 operating system and a design-time static scheduling which respect the ARINC 653 recommendation. Figure 3 is a model of a IMA computing module: CPIOM (Core Processing and Input Output Modules).

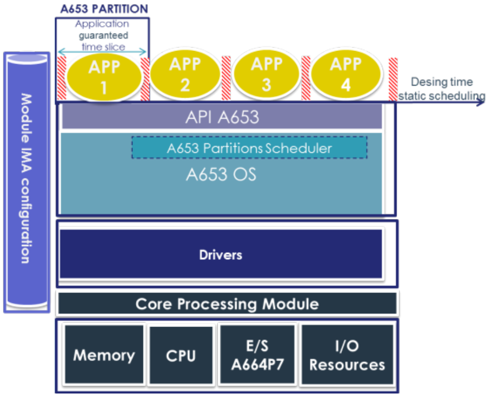


Figure 3. IMA Computing Module (CPIOM)

3.2.2.3 IMA deterministic communication network

Following the IMA objectives, a deterministic network was developed to connect the computing module, not with a one to one link (ARINC 429 [17], ARINC 629 [18], Mil-std 1553 [19]), but through a common redundant deterministic network described by the ARINC 664 [20] specifications.

This deterministic network must support critical inter system communications and also a part of intra system communications. The ARINC 664 Part7 Avionics Full-Duplex Switched Ethernet [21] is the standard in reference for the Avionics Data Network (ADN) and actually largely deployed as Avionics Core Network on several Aircraft platforms. This network architecture is well adapted to support inter avionic partitions communication and must provide to the applications standardized communication services to access the main avionics network or the secondary avionics networks which is able to be used as a communication backup.

The key properties and the embedded associated mechanisms which guarantee the behaviour of the deterministic network and maintain a high level of integrity are based on the elementary dataflow or Virtual Link management and segregation.

The most important properties are per Virtual Link:

- Source and receiver(s) identifications
- Guaranteed throughput
- Dataflow partitioning and monitoring
- Bounded frame delay, with a firm, mathematically provable, bound
- Bounded frame delay jitter mathematically computable
- No frame loss by contention
- Ordinal integrity of frames
- Fault containment and link availability
- Quality of Service (QoS)

The ARINC 664 Part7 communication architecture is designed using dedicated network equipment or shared network equipment acting as Ethernet Frame Switch with capacities and functionalities compliant with the IMA key properties.

It clearly introduces two basic notions:

- The A664 End System located into each subscriber equipment is acting as a message exchanger with the network for the local host user. It provides a Transmission UDP-IP Profile Services and include a shaping mechanism to guarantee the sharing of the available bandwidth between the local host applications
- The A664 Intermediate System (Switch) is acting as Ethernet Frame Switch with capacities and functionalities compliant with the IMA key properties that convey frame only if the VL identification is successful

In typical communication architecture the switch has a central position regarding the location of the subscriber equipment. The system constraints as availability and segregation require a minimum number of independents switches equipment: 2 or 4. In a Centralized Network System as ARINC 664 Part7, each elementary dataflow has two predefined physical way often balanced to reach the group of receiver equipment. The dataflow must be defined “at design time” in static forwarding tables (including frame filtering, traffic policing, Quality of Service (QoS) policies, diffusion) and are stored in each switch. This type of communication architecture is well adapted for a large aircraft. The following figure is an IMA Computing and communication architecture example with one redundant switch (see Figure 4).

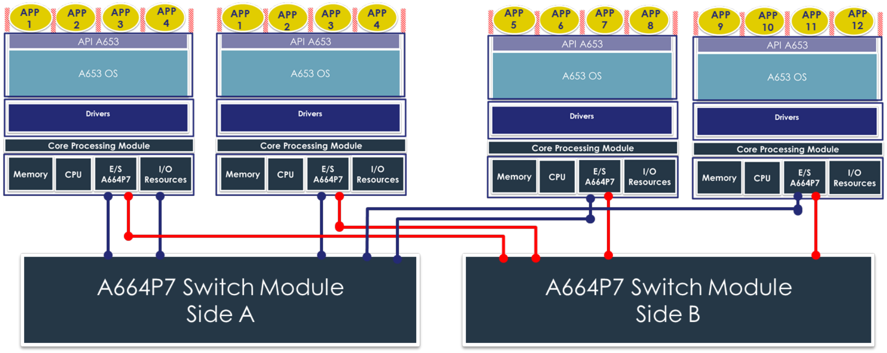


Figure 4. IMA Computing and communication architecture example

3.2.2.4 IMA Conclusion

The evolution towards the IMA architecture has maintained the main mandatory avionics properties: the isolation of the avionics applications, by implementing a robust partitioning on the applications being executed on dedicated computing module and on the dedicated deterministic networks. The successful implementation of the robust partitioning has involved the enforcement of the following non-functional properties:

- The mastery of execution time
- The mastery of memory access
- The mastery of the access to inputs and outputs
- The mastery of all exchanges

3.2.2.5 Avionics Cybersecurity

Furthermore, as the next generation of aircraft will be designed to be more and more connected, they will be more vulnerable to the standard security threats. So the cybersecurity requirements on aircraft systems and networks will be enforced by the EASA/FAA authorities through the adoption of the new following guidance provided in:

- DO-326A: Airworthiness Security Process Specification (2014) [22]
- DO-356A: Airworthiness Security Methods and Considerations (2018) [23]
- DO-355: Information Security Guidance for Continuing Airworthiness (2014) [24]
- And also by applying the recommendations provided from the report, tasked by the FAA, of the Aviation Rulemaking Advisory Committee (ARAC) Aircraft System Information Security/Protection (ASISP) Working Group [25].

From a cybersecurity point of view, this guidance concerns only the cybersecurity threads that may have an impact on the safety and airworthiness of the aeroplane and its operation.

3.2.2.6 Avionics Conclusion

In conclusion, the next generation of aircraft architecture will have to take into account the current constraints provided by aviation regulatory agencies with a high focus on cybersecurity. Now, it would be necessary to add a new property to summarize the main mandatory avionics properties:

- Determinism
- Safety
- Availability
- Cybersecurity for safety

The future generation of aircrafts will still have to comply with the properties of avionics, but they will have to improve cybersecurity for all aircraft functions [26]. As we have seen, the avionics architecture is a critical embedded distributed computing platform which is becoming more and more complex. It will be difficult to guarantee that safety and cyber security properties are well maintained during all phases of the flight. So, it will require implementing advanced monitoring mechanisms to confirm in real-time the validity of end-to-end non-functional requirements. A path of improvement is to distribute the advanced monitoring computing functionality between the aircraft systems and the airline maintenance centre [27].

There is an opportunity to embed in the avionics system some of the monitoring functions performed on the ground, which are able to extract high level safety-related and security-related events that occur during the flight.

One of the relevant technologies that can provide an answer to this challenge was analysed by the ITU-T (Telecommunication Standardization Sector of the International Telecommunication Union) which provided recommendations to the avionics certification authorities. ITU-T and the actors of the aviation industry have established a working Group (2014/2016) on Aviation Applications of Cloud Computing for Flight Data Monitoring [28].

The Flight Data Monitoring is a "on ground" process which brings to an airline company the capacity to analyse a wide range of operational parameters coming from the recorder systems of their aircrafts. The aims of the FDM process is to provide the capacity to analyse and identify the hazardous conditions events in order to determine the root causes of in-service incidents. Currently, the recorded operational parameters are downloaded periodically when the aircraft reaches an airline maintenance center [27].

The ITU-T working group objective was to evaluate the interest to introduce technologies from the cloud computing and big data analytics domain into the avionic domain and all the others aviation related on-ground domains (maintenance, flight tracking) [29] to increase the safety and security of the future aircraft in order to have as soon as possible (on the fly, and not analysed after the event occurs) the aircraft safety and security status.

It has suggested embedding some functionalities of the Flight Data Monitoring. It has recommended to use the key advantage provided by 'Data in Motion' analytics to create the future embedded FDM. The main asset is to implement the fundamental ability to analyse, in flight, the data collected from various aircraft sources and identify the safety-related events that are occurring to take the most appropriate safety action or to send an alert to the ground station for more investigations and recommendations.

The ITU-T Working Group has determined the existing and emerging technologies from the areas of cloud computing and data analytics that could be utilized for FDM in the aviation industry. It as has also established that the following avionics domain may be interest to use the cloud computing and data analytics:

- Real-time flight data monitoring
- Air streaming
- Aircraft tracking

Based on the above Working Group, the TSAG (Telecommunication Standardization Advisory Group) recommend to ITU-T to study in [30] the following existing and emerging technologies clouds domain to develop a specific real-time aviation cloud:

- Inter-cloud computing
- Audio and video analytics
- Digital asset profile system
- Machine learning
- Fog computing
- Data analytics, in-motion data analytics
- Quantum computing

3.2.3 Automotive

Cars are quickly morphing from an isolated, largely mechanical piece of equipment to one of the most technically sophisticated and connected platforms on the planet. From entertainment and navigation to driver assistance and crash avoidance, today's cars are vastly different from those of a few years ago. These connected-cars initiatives are categorized into two primary categories:

- Advanced Driver Assistance Systems (ADAS), which includes services such as navigation, remote diagnostics and collision warning;

- Autonomous Driving (AD), which includes services such as automatic parking and autopilot.

The difference between automated driving and Advanced Driver Assistance Systems (ADAS) can be boiled down to the level of human intervention. Automated driving can be described as driving enhanced by dedicated control through autonomous (sub)systems that support the driver, while he/she is in control or able to timely get back in control, making the driver legally responsible throughout for carrying out the driving task. Autonomous Driving (AD) is the extreme end result of automated driving. There are generally five levels of autonomous or self-driving vehicles ranging from Level 0 to Level 5 [31], in which Level 0 means no automation and Level 5 provides completely self-driven unmanned vehicle. The vehicle controls all its operations by itself, under all conditions. Therefore, with the development of AD, ADAS prevails at the same time.

Based on a recent survey conducted in the USA, UK and Australia, 56.8% of the people had a positive opinion, 29.4% had a neutral opinion and only 13.8% had a negative opinion about the autonomous or self-driving vehicles [32]. These statistics give us a good picture of the general public’s interest in autonomous vehicles. However, the same study also shows us that people do have high levels of concerns regarding safety, privacy, energy and performance issues.

Over the last few decades, safety has become an increasingly important concern for the automotive industry. Safety testing organizations such as the European New Car Assessment Program (Euro NCAP) provide customers with information about the safety ratings for different makes and models. Today, these rankings are based on a vehicle’s *passive safety*, taking into account things such as airbags and crumple-zones. Over the last few years, however, automotive manufacturers have been putting much effort into *active safety*. Active safety systems, also known as Advanced Driver Assistance Systems (ADAS), are a variety of independent electronic systems designed to help the driver manoeuvre through demanding traffic situations. Their overall aim is to reduce traffic accidents and to make the driving experience easier and more efficient [44].

ADAS can offer support to the driver at four different levels. At the most basic level, they present drivers with information which enables them to make more informed driving decisions, for example information about pedestrians not visible to the driver during night driving. At the next level, ADAS can give the driver warnings of an imminent and possibly perilous situation to give them more time for decision making and reaction. The third level of intervention involves the system not only warning the driver but also advising or guiding them through the situation. At the highest level of intervention, ADAS either take action independently or override the action of the driver. Regardless of level of intervention, manufacturers who implement these systems hope to increase driving safety by assisting the driver before a critical situation arises or, at least, to reduce the consequences of driver errors.

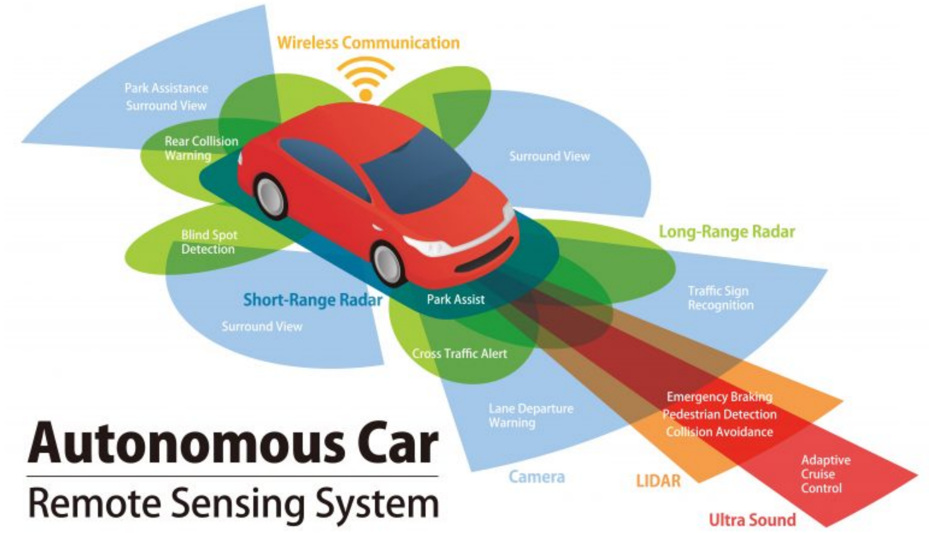


Figure 5. Types of sensors used in ADAS [50]

As shown in Figure 5, some of the sensors used in ADAS include cameras, radar, and LIDAR [50]. Using multiple sensor technologies improves the safety level of the car and at the same time can relax the safety requirement for each individual sensor.

Cameras are the only sensors that actually “see”. They can recognize texture, detect traffic signs, can be used for object detection, and can inexpensively build a 3D map of the area surrounding the car. A significant downside to using camera sensors is that poor visibility conditions – weather, low light, and glare – affect the camera’s efficacy. Also, to avoid image degradation from the original source, little or no video compression can be applied typically. Therefore, cameras in safety-critical ADAS applications require a high-speed interface to transmit raw data to the central sensor fusion unit.

Radar is very robust in every weather condition, in difficult light conditions, and has a good range – but it does have some disadvantages in terms of the angular and range resolution. With an angular resolution of 1.2° and range accuracy of about 10cm, there is little room for error. 10cm can be the difference between a near miss and a catastrophe. A lot of innovation has been recently going on to improve the resolution by developing 77-81GHz imaging radar systems. However, the higher the resolution of the radar sensor, the higher the data rate that needs to be transmitted.

LIDAR offers the best of both worlds: it can capture an effective 3D map of the areas surrounding a car with superior resolution. The angle of resolution for a LIDAR system is 0.1°, and the range accuracy is better than radar, about 5cm or less. There are two popular concepts for solid-state lidar technology, flash lidar sensors, and pulsed timed-flight lidar sensors using micro-electro-mechanical systems (MEMS). Current LIDAR systems require a relatively low transfer data rate (hundreds of Mb/sec), compared to high-resolution imaging radar systems. However, future solid-state LIDAR systems will produce a much higher raw data rate with more than 1Gb/sec, which might ask for on-device processing.

Sufficient power needs to be available to operate all these car components at all times needed, which of course involves when the car is in motion, but also when it is idle, and even to some degree when the car is no longer considered in active use for a driving journey. It's taken for granted that the autonomous systems being tested right now require a lot of computing power, but it's easy to overlook that all of that computing power comes at a cost of actual *electric* power. With the coming autonomous future, it's also taken for granted that cars will all be electric or hybrid by then – Tesla's semi-autonomous Autopilot system is already in a car that's electric - but much more complex Level 3 through Level 5 systems will also require a lot more computing power to run, putting their requirements at odds with the car's own powertrain system. Current prototypes for fully autonomous driving systems consume the equivalent energy of 50 to 100 laptops [47]. This translates to 2 to 4 kilowatts of electricity, which in a modern car makes it 5 to 10 percent more difficult to meet fuel economy and carbon emission targets.

Cybersecurity is also an important concern as it relates to both trust and acceptance of autonomous driving technology as well as safety. Like all other connected communications networks, concerns remain that hackers could steal personal information and spy on people or that malicious control of a vehicle could cause personal harm or disrupt traffic flow [48][49].

Modern cars contain more and more safety-relevant features which require addressing safety aspects during all development phases: on the functional level, on the architectural level, during integration as well as throughout the verification [41].

Non-functional aspects of safe software include a sound and safe timing of the software. The right methods, tools and standards enable OEMs and suppliers developing and providing applications that meet their timing requirements and a high level of quality. Applications are being targeted to allocate them on multi/many-core based systems, in order to exploit their parallel processing capability. The research progresses in the direction of dynamic resource allocation [42].

The most demanding applications are those in which the output of the analysis has a hard time deadline – an object detected too late can lead to self-driving cars' inability to avoid objects and thus cause accidents [51][52]. For instance, the capability of an autonomous vehicle to perform safe and efficient navigation in dynamic environments is dependent on being able to perform the tasks of sensing, planning and actuation. This is challenging because of the tight coupling between the planner and other tasks with respect to time and information. In order to verify that the system as a whole can satisfy the safety and efficiency requirements, each task must produce results that are correct, and those results must be produced within an appropriate time window.

Current state of the art autonomous vehicles can sense their local environment, identify different objects, have knowledge about the evolution of their environment and can plan complex motion by obeying different rules. Many advancements have been made in the recent years, evidenced through different successful demonstrations and competitions. The most prominent historical series of such competitions/challenges were organized by the US Department of Defense under the Defense Advanced Research Projects Agency (DARPA) [33].

The earlier implementations of autonomous vehicles and other autonomous systems were standalone implementations. In fact, most of the existing implementations are still operating independently [33]. In such standalone implementations, the system is limited to the onboard capabilities such as memory, computations, data and programs. Therefore, the vehicles cannot interact with each other or have access to each other's information or information about their surroundings.

On a higher level, they decompose the system architecture in four basic subsystems, namely, sensing, perception, planning and control (see Figure 6 for a graphical representation) [33]. The sensing unit takes raw measurements from different on-

/off-board sensors (e.g. GPS, radar, LIDAR, odometer, vision, inertial measurement unit, etc.) for perceiving the static and dynamic environments. Sensor units pass the raw data to the perception unit, which then generates the usable information about the vehicle (e.g. position, map relative estimations, etc.) and its environment (e.g. lanes of other vehicles, obstacles, etc.), based on provided data. The planner unit takes the usable information/estimations from the perception unit, reasons about the provided information and plans about the vehicle's actuations in the environment, such as path, behavioural, escalation and map planning, etc., to maximize their well-defined utility functions. Finally, the planner unit passes the ultimate information/commands to the control unit, which is responsible for actuating the vehicle.

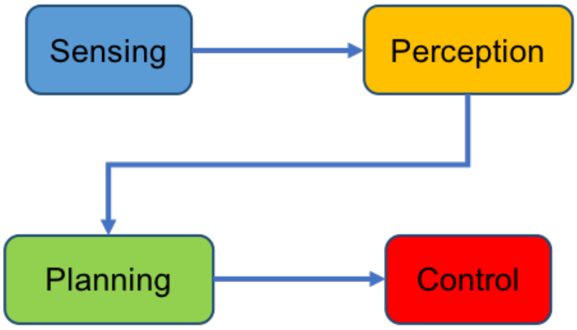


Figure 6. High-level architecture of isolated autonomous vehicles.

One of the main lessons learned from the DARPA challenges was the need for the autonomous vehicles to be connected, that is, the ability to interact with each other and to have access to each other's information or information about their surroundings [33]. Future automotive safety applications based on vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication (jointly referred to as V2X communication) are regarded as a means for decreasing the number of fatal traffic accidents. While these functionalities herald a new era of traffic safety, new security requirements need to be considered in order to prevent attacks on these systems [43].

This also provides us some idea about the importance of the cloud infrastructure [40] in accomplishing the goal of autonomous vehicles, turning cloud-based autonomous driving into an active field of research. In the next paragraph, some of the most representative examples are presented.

In 2012, M. Gerla discussed different design principles, issues and potential applications of the Vehicular Cloud Computing (VCC) [34]. In the same year, S. Kumar et al. proposed the Octree-based cloud-assisted design for autonomous driving of vehicles to assist them in planning their trajectories [35]. In 2014, Gerla et al. investigated the vehicular cloud and deduced that it will be a core system for autonomous vehicles that will make the advancements possible [36]. In early 2016, Maglaras et al. investigated the concept of Social Internet of Vehicles (SIoV), discussed its different design principles, potential applications and research issues [37]. In 2017, Liu et al. proposed a unified cloud infrastructure for distributed computing and storage and exploit heterogeneous computing resources for enhanced performance and energy efficiency [38]. In 2018, Borraz et. al proposed a platform for autonomous driving technology development capable of integrating a wide variety of sensors and actuators which can be used for testing algorithms and control strategies [39]. As a proof of concept, the paper presents a complete navigation application for a commercial vehicle, comprising a complete perception system, an automation of the driving elements of the vehicle, a control system, and a decision-making system.

The high-level system architecture of all these works is presented in Figure 7. The purpose of sensing, planner and controller unit is same as explained for Figure 6, with the modification that the perception unit has been merged into the planner unit, and the planner unit has been divided into two sub-units namely an onboard planner and a planner in the cloud. Both planner units can communicate with each other to exchange desired information and for vehicle-to-vehicle (V2V) communication. The cloud planner can generate requests to various autonomous vehicles for providing sensors' data, which is then aggregated to generate the information about the obstacles, path planning, localization and emergency control, etc. The onboard planner unit communicates with the cloud planner for planning the optimal trajectory and passes the ultimate information to the controller unit, which then actuates the vehicle as required.

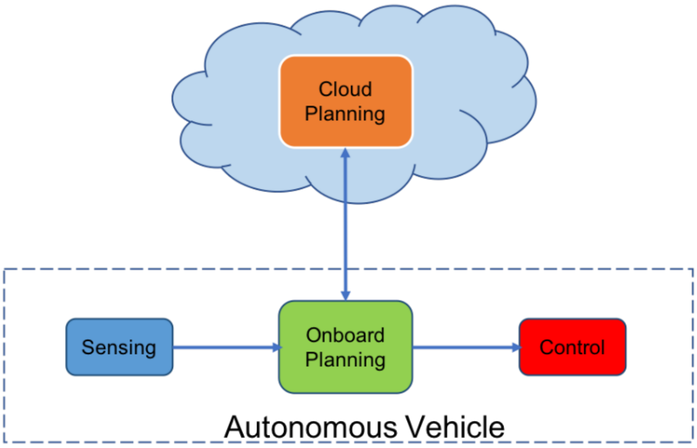


Figure 7. High-level architecture of the cloud-assisted autonomous vehicles

Therefore, the cloud-based infrastructure has the potential to enable a wide range of applications and new paradigms in autonomous vehicles, overcoming the limitations posed by standalone implementations. Developing and deploying AD systems requires the ability to collect, store and manage massive amounts of data, high performance computing capacity and advanced deep learning frameworks, along with the capability to do real-time processing of local rules and events in the vehicle.

Autonomous vehicles require intensive parallel computation cycles to process sensors' data and efficient path planning in the real-world environment [33]. It is certainly not practical to deploy massive onboard computing power with each agent of autonomous vehicle. Such deployments will be cost-intensive and may have certain limitations in parallel processing. On the other hand, the cloud provides massively parallel on demand computation [45]. Nowadays, a wide range of commercial sources (including Amazon's EC2, Microsoft's Azure and Google's Compute Engine) are available for cloud computing services, with the aim to provide access to tens of thousands of processors for on-demand computing tasks [46].

Cloud computing can be used for computationally extensive tasks, such as to find out uncertainties in models, sensing and controls, analysis of videos and images, generate rapidly growing graphs (e.g. RRT*) and mapping, etc. [46]. Many applications require real-time processing of computational tasks, in such applications cloud can be prone to varying network latency and quality of service (QoS), and this has been an active research area nowadays [45][46].

In conclusion, the operation of autonomous vehicles will be heavily reliant on a robust cloud system that enables vehicles to communicate with infrastructure in order to navigate its surroundings safely. While autonomous driving technology from an automotive standpoint exists today, infrastructure still has some way to go in supporting this technology beyond controlled trials. The futuristic advancement of the driverless car requires equally futuristic levels of connectedness – and vast amounts of information both critical and banal – to be accessed simultaneously. This realistically can only be supplied by a cloud system.

4 Non-functional requirements for the ELASTIC software architecture

This section lists the requirements that have some impact in the definition of the software platform. Those requirements are the result of having gathered and analysed the requirements set by the use case providers in WP1 and related application domains, in order to get the specific requirements for the software components and mechanisms that deal with non-functional properties. They are listed one-by-one together with the following attributes:

- ID: the requirement identifier. It is composed by the requirement number headed by three characters NFR.
- Topic: the main system the requirement is applied to. Requirements have been analysed and consolidated in the following topics: Timing, Energy, Communication, Security
- Subtopic: the category of the requirement.
- Name: the friendly name of the requirement
- Description: the body of the requirement, with the associated metrics.
- Means for verification: the way this requirement will be evaluated within the project.
- Type: the type of requirement defined according to the MoSCoW Model:
  - MUST HAVE (M): Defines a requirement that has to be satisfied for final solution to be acceptable. It is mandatory.
  - SHOULD HAVE (S):This is a high-priority requirement that should be included if possible.
  - COULD HAVE (C): This is a desirable or nice-to-have requirement but the solution will still be accepted if the functionality is not included.
  - WOULD LIKE (W): This represents a requirement that stakeholders would like to have but have agreed will not be implemented within the scope of this project
- Implementer: the responsible of the requirement capture within the project (partner acronym).
- Source: Indicates where this requirement comes from.
- Additional Information / Comments: remarks to clarify the requirement, if needed.

4.1 Timing Requirements

ID	REQ-NFR-TIM-0001
Topic	Timing
Subtopic	Control loops
Name	Types of control loops
Description	The ELASTIC architecture must support local and distributed control loops.
Means for verification	Use case demonstrators
Type	M
Implementer(s)	IKL, SIXQ, BSC.
Source	NGAP, ADAS, FLO use case (WP1, D1.1)
Additional Information / Comments	

ID	REQ-NFR-TIM-0002
Topic	Timing
Subtopic	Real-time
Name	Real-time requirements
Description	The ELASTIC architecture must support applications with hard, firm and soft real-time requirements.
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ, BSC, ISEP.
Source	NGAP, ADAS, FLO use case (WP1, D1.1), other domains.
Additional Information / Comments	

ID	REQ-NFR-TIM-0003
Topic	Timing
Subtopic	Real-time
Name	Scope of real-time requirements
Description	The ELASTIC architecture must support local and end-to-end real-time requirements.
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ, BSC, ISEP.
Source	Use cases (WP1, D1.1), other domains.
Additional Information / Comments	

ID	REQ-NFR-TIM-0004
Topic	Timing
Subtopic	Real-time
Name	Measurement of soft timing requirements
Description	The ELASTIC architecture must support soft real-time requirements: • ratio of accepted failure, • maximum admissible number of consecutive failures, and/or • average response time.
Means for verification	Test.
Type	M

Implementer(s)	IKL, SIXQ, BSC, ISEP.
Source	Use cases (WP1, D1.1), other domains.
Additional Information / Comments	

ID	REQ-NFR-TIM-0005
Topic	Timing
Subtopic	Real-time
Name	Periods
Description	The ELASTIC architecture must support sensor processing rates in the order of 1000 measures/s.
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ
Source	Use cases (WP1, D1.1)
Additional Information / Comments	

ID	REQ-NFR-TIM-0006
Topic	Timing
Subtopic	Real-time
Name	Response time
Description	The ELASTIC architecture must support end-to-end response time in the order of milliseconds (msec).
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ, BSC, ISEP.
Source	Use cases (WP1, D1.1)
Additional Information / Comments	

ID	REQ-NFR-TIM-0007
Topic	Timing
Subtopic	Scheduling
Name	Real-time scheduling
Description	The ELASTIC architecture must support schedulers with real-time constraints.
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ, BSC, ISEP.
Source	Other domains.
Additional Information / Comments	

ID	REQ-NFR-TIM-0008
Topic	Timing
Subtopic	Scheduling
Name	Mixed-criticality scheduling
Description	The ELASTIC architecture should support schedulers with mixed-criticality constraints.
Means for verification	Source code

Type	S
Implementer(s)	IKL, SIXQ, BSC, ISEP.
Source	Other domains
Additional Information / Comments	

ID	REQ-NFR-TIM-0009
Topic	Timing
Subtopic	Scheduling
Name	Multi-core scheduling
Description	The ELASTIC architecture must support schedulers with multi-core constraints.
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ, BSC, ISEP.
Source	Other domains.
Additional Information / Comments	

ID	REQ-NFR-TIM-0010
Topic	Timing
Subtopic	Worst-case execution time
Name	Worst-case execution time measurements
Description	The ELASTIC architecture must support worst-case execution time (WCET) measurements.
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ, ISEP.
Source	Use cases (WP1, D1.1), other domains.
Additional Information / Comments	

ID	REQ-NFR-TIM-0011
Topic	Timing
Subtopic	Worst-case execution time
Name	Worst-case execution time analysis
Description	The ELASTIC architecture should support worst-case execution time (WCET) analysis.
Means for verification	Inspection
Type	S
Implementer(s)	IKL, SIXQ, ISEP.
Source	Use cases (WP1, D1.1), other domains.
Additional Information / Comments	

ID	REQ-NFR-TIM-0012
Topic	Timing
Subtopic	Worst-case execution time
Name	Execution time on multi-core / many-core platforms
Description	The ELASTIC architecture must support applications executing in nodes with multi-core / many-core

	processors that require execution time measurements/analysis.
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ, ISEP.
Source	Use cases (WP1, D1.1), other domains.
Additional Information / Comments	

ID	REQ-NFR-TIM-0013
Topic	Timing
Subtopic	Modes of operation
Name	Multiple modes of operation
Description	The ELASTIC architecture must support applications with multiple modes of operation. Modes of operation depend on: • internal conditions (e.g. system self checks, etc), • external conditions (e.g. position/speed of the train, track conditions, etc).
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ, ISEP, BSC.
Source	Use cases (WP1, D1.1), other domains.
Additional Information / Comments	

ID	REQ-NFR-TIM-0014
Topic	Timing
Subtopic	Configuration
Name	Allocation of components to nodes
Description	The ELASTIC architecture must support components to be allocated to nodes: • dynamically at run-time, • statically at deployment time.
Means for verification	Test.
Type	M
Implementer(s)	IKL, SIXQ, ISEP, BSC.
Source	Use cases (WP1, D1.1), other domains.
Additional Information / Comments	

4.2 Energy Requirements

ID	REQ-NFR-NRG-0001
Topic	Energy
Subtopic	Energy monitoring
Name	Energy monitoring on the edge
Description	The ELASTIC architecture must support energy monitoring on edge nodes.
Means for verification	Test.

Type	M
Implementer(s)	IKL, SIXQ
Source	Use cases (WP1, D1.1), other domains.
Additional Information / Comments	

ID	REQ-NFR-NRG-0002
Topic	Energy
Subtopic	Energy monitoring
Name	Energy monitoring on the cloud
Description	The ELASTIC architecture should support energy monitoring on cloud nodes.
Means for verification	Inspection.
Type	S
Implementer(s)	SIXQ
Source	Use cases (WP1, D1.1), other domains.
Additional Information / Comments	

ID	REQ-NFR-NRG-0003
Topic	Energy
Subtopic	Scheduling
Name	Energy-aware scheduling
Description	The ELASTIC architecture should support scheduling techniques designed to ensure energy requirements.
Means for verification	Test.
Type	S
Implementer(s)	IKL, SIXQ, ISEP, BSC.
Source	Other domains.
Additional Information / Comments	

ID	REQ-NFR-NRG-0004
Topic	Energy
Subtopic	Energy-efficiency
Name	Hardware speed scaling
Description	The ELASTIC architecture should support speed scaling techniques as means to ensure energy requirements.
Means for verification	Test.
Type	S
Implementer(s)	IKL, SIXQ, BSC, ISEP
Source	Other domains.
Additional Information / Comments	

ID	REQ-NFR-NRG-0005
Topic	Energy
Subtopic	Energy-efficiency
Name	Multi-core energy mechanisms

Description	The ELASTIC architecture should support on-line task-to-core allocation techniques that ensure energy requirements.
Means for verification	Test.
Type	S
Implementer(s)	IKL, SIXQ, BSC, ISEP
Source	Other domains.
Additional Information / Comments	

ID	REQ-NFR-NRG-0006
Topic	Energy
Subtopic	Modes of operation
Name	Multiple modes of operation
Description	The ELASTIC architecture should support modes of operation designed for specific energy utilisation profiles. ELASTIC components switch between modes depending on predefined conditions.
Means for verification	Test.
Type	S
Implementer(s)	IKL, SIXQ, BSC, ISEP
Source	Other domains.
Additional Information / Comments	

4.3 Communication Requirements

ID	REQ-NFR-COM-0001
Topic	Communication
Subtopic	Communication interfaces
Name	Gateway interfaces with the cloud
Description	The Gateway must have the following interfaces for communicating with the cloud: • LTE • WiFi (802.11 a/b/g/n/ac enterprise)
Means for verification	Visual / Source code
Type	M
Implementer(s)	IKL, TRT, SIXQ, BSC, ISEP
Source	Use cases (WP1, D1.1)
Additional Information / Comments	

ID	REQ-NFR-COM-0002
Topic	Communication
Subtopic	Communication protocols
Name	Protocols supported by the fog architecture
Description	The fog computing architecture must allow users to deploy the following protocols for the Gateway to communicate with the nodes:

	- TCP/IPv4, TCP/IPv6, HTTP, HTTPS, FTP, SSH, TCP, SOAP, REST, NTP, UDP, SNMP, DNS, RADIUS
Means for verification	Use-case demonstrators
Type	M
Implementer(s)	IKL, TRT, SIXQ, BSC, ISEP
Source	Use cases (WP1, D1.1)
Additional Information / Comments	Not all the protocols are going to be implemented in all the use-cases.

ID	REQ-NFR-COM-0003
Topic	Communication
Subtopic	Information
Name	Data exchange among the fog computing architecture
Description	- The Gateway must be able to receive monitoring data, images and video from the edge nodes and resend them to the cloud. - The Gateway must be able to receive control data from the cloud and resend them to the edge nodes.
Means for verification	Source code / Use-case demonstrators
Type	M
Implementer(s)	IKL, TRT, SIXQ, BSC, ISEP
Source	Use cases (WP1, D1.1)
Additional Information / Comments	Data messages frequency is one every [100-200] ms

ID	REQ-NFR-COM-0004
Topic	Communication
Subtopic	Communication features
Name	End to end integrity
Description	The communication among the Gateway and the cloud must provide end to end integrity by means of the following mechanisms: - CRC or similar - Message timestamping - Geo-localization of the computing node/sensor which produces the message - ID of the computing node/sensor which produces the message
Means for verification	Source code / Use-case demonstrators
Type	M
Implementer(s)	IKL, TRT, SIXQ, BSC, ISEP
Source	Use cases (WP1, D1.1)
Additional Information / Comments	Not all the mechanisms have to be used in every data communication.

ID	REQ-NFR-COM-0005
Topic	Communication

Subtopic	Communication features
Name	Broadcast/multicast
Description	The communication among the Gateway and the cloud could provide broadcast/multicast messages.
Means for verification	Source code / Use-case demonstrators
Type	C
Implementer(s)	IKL
Source	Use cases (WP1, D1.1)
Additional Information / Comments	

ID	REQ-NFR-COM-0006
Topic	Communication
Subtopic	Communication features
Name	Acknowledge
Description	The communication among the Gateway and the cloud could provide acknowledge messages.
Means for verification	Source code / Use-case demonstrators
Type	C
Implementer(s)	IKL
Source	Use cases (WP1, D1.1)
Additional Information / Comments	

ID	REQ-NFR-COM-0007
Topic	Communication
Subtopic	Communication features
Name	Priority
Description	The communication among the Gateway and the cloud could provide priority in messages.
Means for verification	Source code / Use-case demonstrators
Type	C
Implementer(s)	IKL
Source	Use cases (WP1, D1.1)
Additional Information / Comments	

ID	REQ-NFR-COM-0008
Topic	Communication
Subtopic	Quality of Service
Name	Channel availability
Description	ELASTIC must provide a means to monitor and measure the availability of the communication service.
Means for verification	Source code / Use-case demonstrators
Type	M
Implementer(s)	IKL, TRT, SIXQ, BSC, ISEP

Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	Relative parameters will be provided: bandwidth, bit error rate, latency…

ID	REQ-NFR-COM-0009
Topic	Communication
Subtopic	Quality of Service
Name	Latency
Description	ELASTIC must provide low latency communication protocols in order to collect real time data from devices and must support real-time traffic in order to process sensor data.
Means for verification	Source code / Use-case demonstrators
Type	M
Implementer(s)	IKL, TRT, SIXQ, BSC, ISEP
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	Response time should be in the range of ms.

ID	REQ-NFR-COM-0010
Topic	Communication
Subtopic	Quality of Service
Name	Reliable communication channel
Description	ELASTIC must ensure reliable communication channels to avoid information loss and must implement verification mechanisms for data communication, in order to realise reliable communication on top of communication infrastructures with unknown quality (mechanisms for out-of-order-delivery and acknowledgement, for example).
Means for verification	Source code / Use-case demonstrators
Type	M
Implementer(s)	IKL
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0011
Topic	Communication
Subtopic	Quality of Service
Name	QoS configuration
Description	ELASTIC must provide the service to configurate the communication parameters: buffer size, bandwidth, throughput, middleware parameters, prioritization of requests, etc.
Means for verification	Source code / Use-case demonstrators
Type	M
Implementer(s)	IKL

Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0012
Topic	Communication
Subtopic	Communication Dataflow
Name	Dataflow classification
Description	All communication dataflow must be the specified and must be classified into critical and non-critical dataflow and their main dataflow characteristic must be defined.
Means for verification	Inspection.
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0013
Topic	Communication
Subtopic	Communication Dataflow QoS
Name	Critical dataflow QoS
Description	A critical dataflow must have always a greater QoS priority than the non-critical dataflow.
Means for verification	Inspection.
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0014
Topic	Communication
Subtopic	Communication Dataflow QoS
Name	Critical message timeliness
Description	At the receiver device, for a given critical dataflow the messages must be processed respecting the timeliness.
Means for verification	Test
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0015
Topic	Communication
Subtopic	Communication Dataflow QoS
Name	Non-critical dataflow QoS

Description	A non-critical dataflow should be managed with best effort QoS mechanisms.
Means for verification	Test
Type	S
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0016
Topic	Communication
Subtopic	Safety Communication systems
Name	Wireless spectrum analysis
Description	It could be also be necessary to do a wireless spectrum analysis all along the tramway line.
Means for verification	Analysis Data
Type	C
Implementer(s)	TRT, THALIT, GEST, FLO
Source	Other domains
Additional Information / Comments	It may be useful to detect the areas of the tramway line that have a wireless spectrum able to degrade the tramway wireless communication.

ID	REQ-NFR-COM-0017
Topic	Communication
Subtopic	Safety Communication systems
Name	Denial of service robustness
Description	The communication systems should be robust against denial of service.
Means for verification	Test
Type	S
Implementer(s)	IKL, TRT
Source	Other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0018
Topic	Communication
Subtopic	Safety Communication systems
Name	End to end bandwidth allocation
Description	The communication systems should be able to guarantee an end to end bandwidth allocation for the critical dataflow.
Means for verification	TEST
Type	S
Implementer(s)	IKL, TRT
Source	Other domains

Additional Information / Comments	

ID	REQ-NFR-COM-0019
Topic	Communication
Subtopic	Message Structure
Name	Message structure
Description	The critical and non-critical messages must have different structures achieved by applying at least safety code into the critical messages.
Means for verification	Inspection
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0020
Topic	Communication
Subtopic	Message Structure
Name	Identifiers uniqueness
Description	The entities(nodes) identifiers must be unique in the entire communication system.
Means for verification	Inspection
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0021
Topic	Communication
Subtopic	Safety Critical Message Structure
Name	Safety critical message minimal metadata
Description	The critical messages must include at least the following meta data fields: • Source identifiers • Destination identifiers • Source Geo-localization • Sequence number • Time stamps • Safety code • QoS
Means for verification	Inspection
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains

Additional Information / Comments	

ID	REQ-NFR-COM-0022
Topic	Communication
Subtopic	Safety Critical Message Structure
Name	Sequence number
Description	The length of the sequence number must be at least of 32 bits.
Means for verification	Inspection
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0023
Topic	Communication
Subtopic	Safety Critical Message Structure
Name	Message time stamp length
Description	The length of the time stamp must be at least of 64 bits.
Means for verification	Inspection
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0024
Topic	Communication
Subtopic	Safety Critical Message Structure
Name	Message Geo localization
Description	The length of the Geo localization stamp must be at least of 32 bits.
Means for verification	Inspection
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0025
Topic	Communication
Subtopic	Safety Critical Message Structure
Name	Message Geo localization initialisation

Description	Geo localization stamp must be considered as the space travelled from a starting point 0 saved in the database splines.
Means for verification	Test
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0026
Topic	Communication
Subtopic	Message Safety Code
Name	Safety code structure
Description	The safety code must be different from the transmission code.
Means for verification	Inspection
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0027
Topic	Communication
Subtopic	Message Safety Code
Name	EMI safety code
Description	The safety code should at least detect the transmission errors caused by an EMI environment.
Means for verification	Test/ Polynomial state of the art
Type	S
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0028
Topic	Communication
Subtopic	Communication Safety Code
Name	CRC safety code
Description	The safety code must be a 64-bit cyclic redundancy check.
Means for verification	Inspection
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains

Additional Information / Comments	

ID	REQ-NFR-COM-0029
Topic	Communication
Subtopic	Communication Safety Code
Name	Safety Code monitoring
Description	At the receiver device, the safety code must be checked. If the safety code check detects an error the message must be dropped. The number safety code errors must be notified to the user.
Means for verification	Test
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0030
Topic	Communication
Subtopic	Message Sequence Number
Name	Sequence number rules
Description	For a given critical dataflow, the message sequence number should be incremented by one, starting at 0 and wrapping at (2^{32} -1) to 1.
Means for verification	Test
Type	S
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	A sequence number of 0 could be used to indicate a reset the transmitting device

ID	REQ-NFR-COM-0031
Topic	Communication
Subtopic	Message Sequence Number
Name	Critical dataflow ordinality
Description	At the receiver device, for a given critical dataflow the messages must be processed in order.
Means for verification	Test
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0032
Topic	Communication
Subtopic	Message Sequence Number

Name	Lost message
Description	At the receiver device, for a given critical dataflow the missing messages must be counted and notified to the applications, and to the operational centre.
Means for verification	Test
Type	M
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-COM-0033
Topic	Communication
Subtopic	Runtime message transition time mechanisms
Name	Transition time monitoring
Description	Runtime message transition time mechanisms could be considered to monitor the transition time between each communication node. It shall measure and create the following statistic based on the observed duration of the Message Transition Time indicators: • the Best Case (shortest), • the Average Case (median) • and the Worst Case (longest)
Means for verification	Test
Type	C
Implementer(s)	IKL, TRT
Source	Other domains
Additional Information / Comments	

4.4 Security Requirements

ID	REQ-NFR-SEC-0001
Topic	Security Requirements
Subtopic	Security
Name	Data Flow Security
Description	The fog architecture should support the security of all data flows.
Means for verification	Source code / Use-case demonstrators
Type	S
Implementer(s)	IKL
Source	Other domains
Additional Information / Comments	

ID	REQ-NFR-SEC-0002

Topic	Security Requirements
Subtopic	Security
Name	Authentication
Description	The fog architecture should support authentication.
Means for verification	Source code / Use-case demonstrators
Type	S
Implementer(s)	IKL, TRT
Source	Other domains
Additional Information / Comments	It should have the capability to use digital signatures in order to verify the contents and the sender's identity. Signatures should be protected from tampering by procedure or mechanisms.

ID	REQ-NFR-SEC-0003
Topic	Security Requirements
Subtopic	Security
Name	Encryption
Description	The fog architecture should support message encryption.
Means for verification	Source code / Use-case demonstrators
Type	S
Implementer(s)	IKL, TRT
Source	Other domains
Additional Information / Comments	It should provide secure communications, secure exchange of messages.

ID	REQ-NFR-SEC-0004
Topic	Security Requirements
Subtopic	Security
Name	Integrity
Description	The fog architecture should support integrity.
Means for verification	Source code / Use-case demonstrators
Type	S
Implementer(s)	IKL, TRT
Source	Other domains
Additional Information / Comments	Integrity means (e.g. keys) should be protected (e.g., access, modification, …).

ID	REQ-NFR-SEC-0005
Topic	Security Requirements
Subtopic	Security
Name	Privacy
Description	The fog architecture should support privacy.
Means for verification	Source code / Use-case demonstrators
Type	S
Implementer(s)	IKL

Source	Other domains
Additional Information / Comments	

ID	REQ-NFR-SEC-0006
Topic	Security Requirements
Subtopic	Security
Name	Secure Access of Data Storage
Description	The fog architecture should protect the access to Data Storage (local and cloud).
Means for verification	Source code / Use-case demonstrators
Type	S
Implementer(s)	IKL, TRT
Source	Other domains
Additional Information / Comments	Sensitive data should be stored encrypted. Encryption means should be protected (e.g., modification, access, …).

ID	REQ-NFR-SEC-0007
Topic	Security Requirements
Subtopic	Security
Name	User Roles
Description	The fog architecture should differentiate between user roles (privilege management, Role-based Access Control (RBAC)).
Means for verification	Source code / Use-case demonstrators
Type	S
Implementer(s)	IKL
Source	Other domains
Additional Information / Comments	

ID	REQ-NFR-SEC-0008
Topic	Security Requirements
Subtopic	Security
Name	Credentials' robustness
Description	The fog architecture should support the robustness of credentials.
Means for verification	Source code / Use-case demonstrators
Type	S
Implementer(s)	IKL, TRT
Source	Other domains
Additional Information / Comments	

ID	REQ-NFR-SEC-0009
Topic	Security Requirements

Subtopic	Security
Name	Firmware and credential updates
Description	The fog architecture should provide the mechanisms to securely updates the firmware and the security credentials.
Means for verification	Source code / Use-case demonstrators
Type	S
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1), other domains
Additional Information / Comments	

ID	REQ-NFR-SEC-0010
Topic	Security Requirements
Subtopic	Security
Name	Security Standards
Description	It would be desirable that the fog architecture adhered to the following cybersecurity standards: • ISA/IEC 62443 SL2
Means for verification	-
Type	W
Implementer(s)	IKL, TRT
Source	Use cases (WP1, D1.1)
Additional Information / Comments	The risk analysis is beyond the scope of the project. However, ELASTIC will be designed with security in mind.

ID	REQ-NFR-SEC-0011
Topic	Security Requirements
Subtopic	GDPR (General Data Protection Regulation)
Name	Data Protection Officer
Description	ELASTIC must provide the person responsible of data protection management.
Means for verification	Nomination document
Type	M
Implementer(s)	BSC
Source	GDPR
Additional Information / Comments	This issue is tackled in WP8 and deliverable D8.1.

ID	REQ-NFR-SEC-0012
Topic	Security Requirements
Subtopic	GDPR
Name	Data Protection Mechanisms
Description	ELASTIC must provide mechanisms for data protection management: • Procedure for notifying any kind of data breach

	- Mechanisms to authorise personal data processing, communication and storage.
Means for verification	Source code / Use-case demonstrators
Type	M
Implementer(s)	IKL
Source	GDPR
Additional Information / Comments	

ID	REQ-NFR-SEC-0013
Topic	Security Requirements
Subtopic	GDPR
Name	Storage of Personal Data
Description	The fog architecture must not storage personal data by its own.
Means for verification	Source code / Use-case demonstrators
Type	M
Implementer(s)	IKL
Source	GDPR
Additional Information / Comments	The use case that use the ELASTIC software architecture ecosystem can explicitly storage personal data if needed. It will be responsibility of the use case provider to use pseudonymisation where it applies.

ID	REQ-NFR-SEC-0014
Topic	Security Requirements
Subtopic	GDPR
Name	Records of Processing Activities
Description	ELASTIC must use a syslog server for recording all operations.
Means for verification	Source code / Use-case demonstrators
Type	M
Implementer(s)	IKL
Source	Use cases (WP1, D1.1)
Additional Information / Comments	

5 Technical constraints imposed to the software architecture

Based on what has been discussed in the previous sections, the benefits of distributing tasks along the compute continuum are clear: reliability, availability and privacy provided by the edge combined with cost efficiency provided by the cloud. Figure 8 shows the overall ELASTIC software development ecosystem, including the main software components.

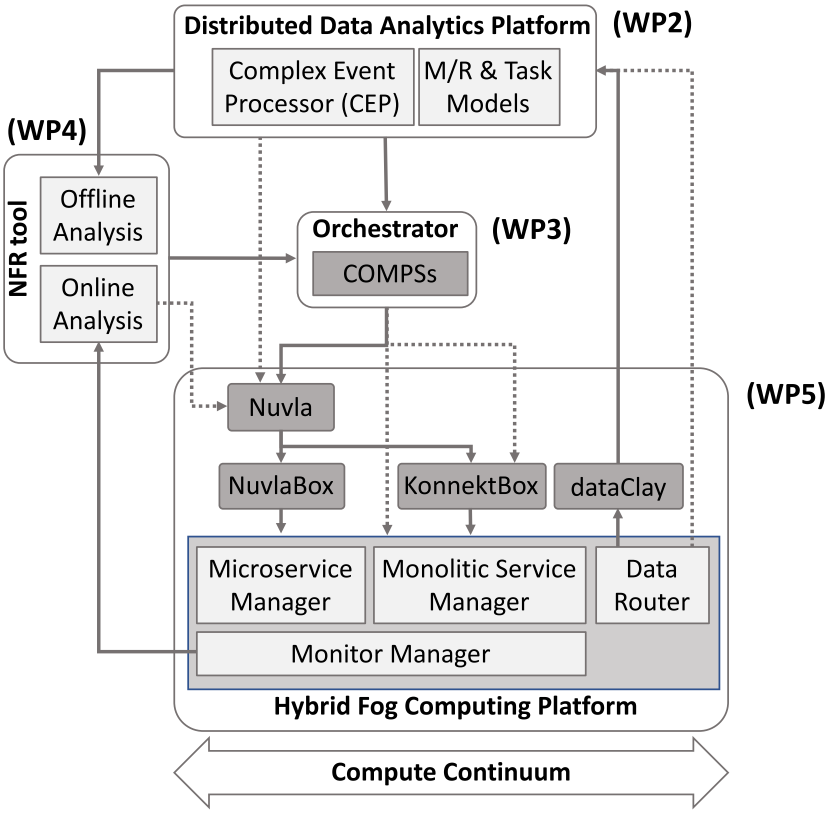


Figure 8. ELASTIC Software Development Ecosystem.

The orchestration layer will consider the input from the NFR tool (WP4), that will (statically) analyse the internal structure of the data analytics and (dynamically) monitor the execution of the system using the hybrid fog computing platform capabilities.

The NFR tool will continuously monitor and evaluate the extent to which non-functional properties required levels are guaranteed by the ELASTIC software components (i.e., real-time, energy, security and communication quality). Moreover, this tool will identify and implement the appropriate mechanisms to deal with these constraints, monitoring system behaviour and helping taking decisions (such as offloading or reducing performance).

5.1 Real-time

Coping with real-time computing across the compute continuum requires the ability to specify and manage different timing perspectives. Two main challenges arise: tasks deployed at the edge (for example, on board the connected car) need to guarantee “hard real-time” responses (e.g. very low latency) and those deployed at the cloud need to guarantee certain QoS levels regarding time: right-time or “soft real-time” guarantees.

Closer to the environment, at the edge, tight timing mapping and scheduling approaches can be used, while at the cloud, time is measured in terms of average statistical performance with Quality of Service (QoS) constraints. These perspectives complement each other, and ELASTIC will provide solutions that will allow to dynamically deploy application components distributed over the system, to provide the required response time to applications, whilst optimizing energy and communication costs. ELASTIC will provide analysis and tools to determine offline a predictive response-time for a particular application, and then dynamically adjusting the system resources to which the application is mapped, depending on the actual load of the system. A particular concern will be given to model the different time scales of the system, providing execution models that consider temporal behaviour of applications from the fast interactions at the edge side to the wider timing perspective of the cloud side.

5.2 Energy

ELASTIC will augment the system “introspection” capabilities in terms of power consumption, by means of energy-aware execution models, from the hardware platform to the holistic system. This will allow to propagate workload-specific information from the run-time to the decision-making module, which can be exploited to better adapt to the requirements, as well as to the time predictability and security optimisation levels. Furthermore, a richer knowledge of applications’ requirements and concurrency structure, coupled with precise energy models for the underlying hardware, combined with the possibility of dynamically switching between edge and cloud deployments, constitutes an enabling factor towards larger energy savings, and the development of novel online and offline optimization strategies.

Moreover, ELASTIC will devise methods to extract information on: (1) workload specification (number and type of containers, tasks, data dependencies, etc.), (2) non-functional specifications included in the distributed programming interface and (3) platform characteristics, impacting on energy efficiency. Based on this information, ELASTIC will develop energy-aware execution models that push the system introspection capabilities beyond what is currently feasible. This will allow ELASTIC to devise allocation strategies and run-time mechanisms that will consider energy information and energy-aware execution models to efficiency tune power consumption over the complete continuum.

5.3 Communication

The compute continuum needs to achieve a safe communication [53] between applications hosted on at least two electronic equipment connected to the transmission system. The objective of the communication in D1.1 Use Cases [1] is to transfer information between applications from the edge side to applications hosted in a private cloud, through a wireless environment.

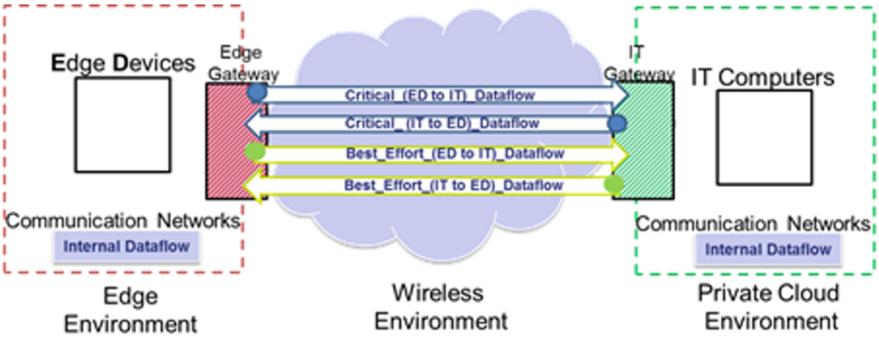


Figure 9. ELASTIC Communication Environment.

The compute continuum needed to implement safety related communication channel to allow the communication between the IT environment (*clouds*) and the Critical Edge Environment (*Tram, Aircraft, Car,...)*. In safety related systems, the communication can be defined as the safe end-to.end transfer of information between applications that are running in different locations. It shall be shown that the end-to-end information being transmitted is complete, not altered, not missing, on time and shall be monitored at runtime. All this monitoring function shall be distributed between the node of the compute continuum and the NFR tool.

Runtime monitoring mechanisms shall be considered to monitor the communication time between each node of the compute continuum, and it shall measure and create the following statistic based on the observed duration and also the observed variation duration of the following Message Transition Time indicators: the Best Case (shortest), the Average Case (median), and the Worst Case (longest). All this monitoring function shall be distributed between the node of the compute continuum and the NFR tool.

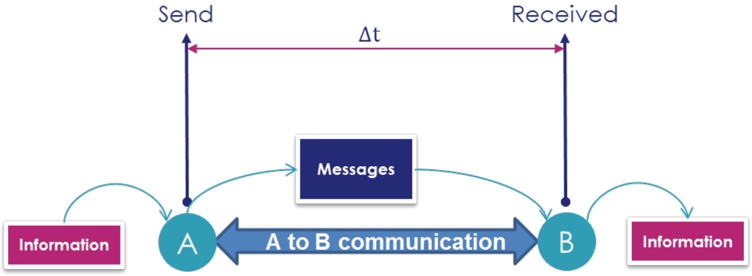


Figure 10. Continuum Communication Node

So, the compute continuum architecture needed to implement a safe full duplex communication system to forward the information. It can use different kinds of network protocols and standards depending mainly on the type of the network physical medium. The protocols are a combination of rules used to manage the communication and defining the structure of the transmitted message that encapsulates the useful information for the application with the useful “metadata” for the communication networks to route the message to the good receiver(s). The main metadata fields (@source, @Dest, Protocol, crc, ….) shall be identified at design time and shall be monitored at runtime in each communication node and the erroneous messages shall be deleted; each metadata field errors shall be reported to the NFR Tool communication supervisors function.

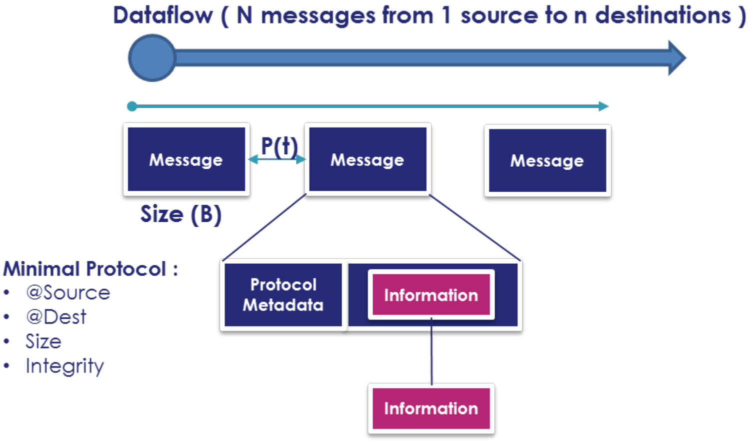


Figure 11. ELASTIC Communication Dataflow

Depending on the trustworthy level of the application domain (Railway, avionics, automotive,…), some key properties and the embedded associated mechanisms are needed to be implemented in order to master the behaviour the communication system. One of the most important tasks shall be the identification, classification and management in real time of all dataflow involved in the system.

A dataflow can be described as a set virtual communication links carrying a set of messages from a unique data source device to one or several data receiver's devices. It is recommended that all dataflow involved in the ELASTIC communication Use Cases shall be identified and their main characteristics defined. The dataflow shall be classified into at least two levels of criticality: the critical dataflow and the best effort dataflow. The critical dataflow must have always a greater QoS priority than the best effort dataflow.

The ELASTIC wired and wireless communication system shall ensure for the critical dataflow a robust partitioning of its allocated bandwidth. It shall ensure the end to end enforcement of the safety and security properties defined during the design phase. The critical dataflow bandwidth availability shall be monitored at runtime in each communication nodes by the NFR tool; each error shall be reported to the NFR tool communication supervisors' function.

5.4 Security

ELASTIC shall satisfy GDPR [54] for managing personal data. Following, the main requirements of this regulation are detailed:

- Scope: The regulation applies if data controller or processor (e.g. cloud service providers) or data subject (person) is based in the EU.
- Responsibility and accountability: There is a requirement for including a retention system for personal data and contact information; a data protection office has to be provided as well.
- Consent: Explicit and valid consent must be given to authorise personal data processing and storage, and the system must be able to prove it.
- Data Protection Officer: A person with expert knowledge of data protection laws and practices must assist the controller or processor to monitor internal compliance with the GDPR regulation.
- Pseudonymisation: The GDPR requires a process that transforms personal data in such a way that the resulting data cannot be attributed to a specific data subject without the use of additional information.

- Data breaches: It is a legal obligation to notify the Supervisory Authority, without undue delay, of any data breaches. Individuals have to be notified if adverse impact is determined. However, the data processor or controller does not have to notify the data subjects if anonymised data is breached.
- Right to erasure: The system must provide the right to be forgotten, that is, to erase personal data related to people.
- Data protection by Design and by Default: The system must be designed with the data protection in mind.
- Records of processing activities: The system must maintain records of activities, and these must be accessible to the supervisory authority upon request.

ELASTIC project allows developers to benefit from a software architecture that provides computing continuum services. The user of the architecture is the final responsible of the data protection. However, this also represents a great responsibility, particularly in the safeguard of personal data. There are different critical stages where personal information is managed. Some examples include to facilitate the erasure of data with no restrictions, the creation of a hierarchy of roles and users for controlling the access to restricted components of the system, etc.

The security requirements resulted of the need of the GDPR regulation compliance by ELASTIC architecture have been divided into two main aspects: technical and those about the limitation of responsibility. The technical aspects focus on management of existing personal information. They have already been gathered in the Security Requirements in section 4.4. The aspects related to the limitation of the responsibility of the ELASTIC platform are aimed to lower the responsibility, limiting, if possible to zero, the unnecessary storing of personal information and providing a solution with data protection in mind. In order to provide the adequate level of privacy, all the data collected from the devices must not be linked with any kind of personal information.

The security requirements of the different use cases must be considered in the ELASTIC architecture. The architecture should implement a secure channel which allows secure bidirectional communications between the different actors.

Secure communication protocols based on TLS (HTTPS, TFTP, MQTTS, etc.) could be used in this channel in order to provide privacy, data integrity and authentication. A RBAC (rule-based access control) can be implemented in order to control the access of each device to certain endpoints of services. The use of device certificates and mutual authentication (server and client) would offer a higher level of security, and improved control of the connected devices and permissions.

5.5 NFR Analysis (Offline)

Contemporary cloud computing solutions, both research projects and commercial products, have mainly focused on providing functionalities at levels close to the infrastructure. Furthermore, they tend to focus on functional aspects only. In order to provide an improved ecosystem, which considers the full compute continuum, there is a great need for tools that support higher-level concerns and non-functional aspects in a comprehensive manner.

In a first phase, represented in Figure 12, the goal of the analysis is to guarantee the fulfilment of each single non-functional property in isolation. In order to be able to use the chosen evaluation metrics in a self-correcting system, it is first necessary to define these metrics in a non-holistic way, for purposes such as deploying alternative combinations of components, or adding/removing a specific component to/from an existing deployed system, etc. Today, a plethora of formal verification techniques exist for different layers in the cloud stack (IaaS, PaaS, SaaS) that typically address a single non-functional property. The result of this analysis is a set of possible initial deployment configurations.

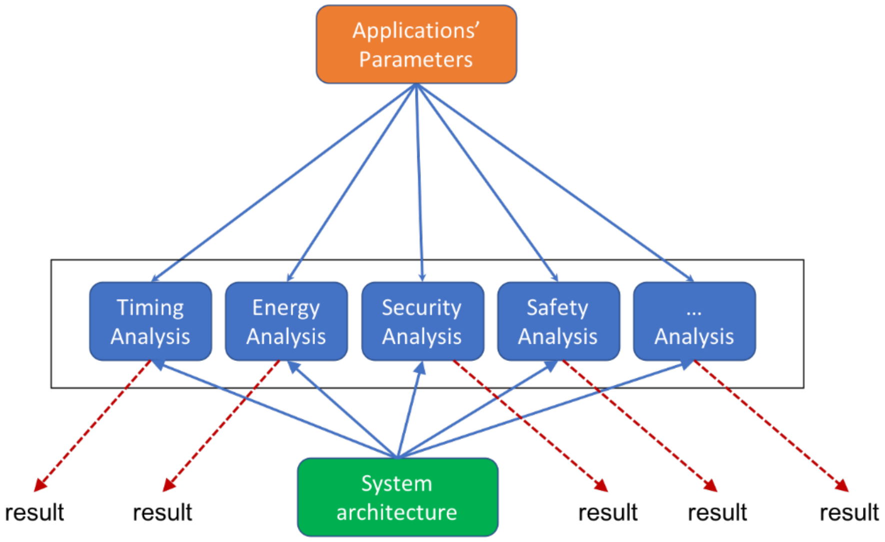


Figure 12. Offline NFR analysis of each property in isolation.

In a second phase, described in Figure 13, this tool will consider them in holistic perspective, considering the potential trade-offs between performance, predictability, energy-efficiency, communication quality and security. For example, in one configuration, to fulfil security requirements, a stronger encryption algorithm might be used than another alternative service configuration. However, using a stronger encryption algorithm may lead to consuming more memory or processing capacity and CPU time, and, in this way, it impacts memory and performance requirements.

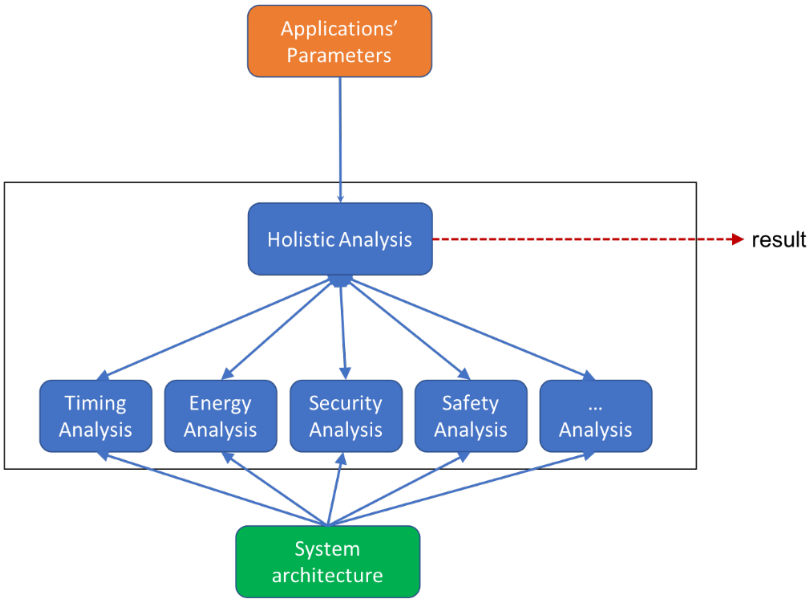


Figure 13. Holistic offline NFR analysis.

Therefore, the NFR tool should be able to carefully identify how satisfying and fulfilling one requirement can impair the satisfaction of other NFRs in the system. Establishing and maintaining such interdependencies during the development process and the lifecycle of the system is also an important point, taking into account the evolution of the software architecture and the introduction of new requirements or the modification of existing ones. Moreover, not only NFRs can have impact on each other, but also one NFR usually crosscuts different parts of a system. For example, achieving security in a system requires design decisions for different parts of a system spanning from user interfaces (e.g., what a user can enter as input), database backends, communication protocols, network topology, and so on.

Table 3 presents the interface of the NFR Offline tool. The tool can be used for the isolated analyses (in phase 1) using the optional --NFR parameter. The other inputs of the tool are the files with the platform description, the application services and the QoS parameters (the required attributes for the non-functional properties). The analysis provides a feasibility check of the system (verifies if the system is able to guarantee the required QoS) as well as identify an appropriate configuration (mapping of services to platform resources).

Description	API
Analyze a single property. The result is the system configuration.	`analyze [--NFR XXX] --platform-description platform.xml --application-services services.xml --QoS-parameters qos.xml --output system_configuration.xml`

Table 3. NFR Offline tool interface

5.6 NFR Monitor (Online)

Deployment decisions should be made in light of a target system, aiming for high quality of the system deployed under given constraints. However, in order to support deployment decisions, it is essential to identify concrete measures as a basis for decision making and evaluation of the proposed solutions. Such measures can be static, as described in the previous section, and dynamic, in the case of systems that evolve continuously as the workloads, allocated resources and requirements of these systems change over time.

Therefore, runtime monitoring of NFRs should be used to guide this evolution towards configurations that are guaranteed to satisfy the system's NFRs. Monitoring should be used to identify the scenario the system operates in, and to select a model whose quantitative verification enables the detection or, sometimes, prediction of NFRs violations. The subsequent synthesis and execution of a provably correct reconfiguration plan help the system to re-instate or maintain compliance with NFRs.

Similarly to the offline phase, in the online phase ELASTIC will first follow an isolated online analysis, as described in Figure 14. For each non-functional property, a specific Service Level Agreement (SLA) manager is provided. The application parameters and monitoring information are sent in parallel to all SLA Managers which output independent decisions on how changes are required to system configuration.

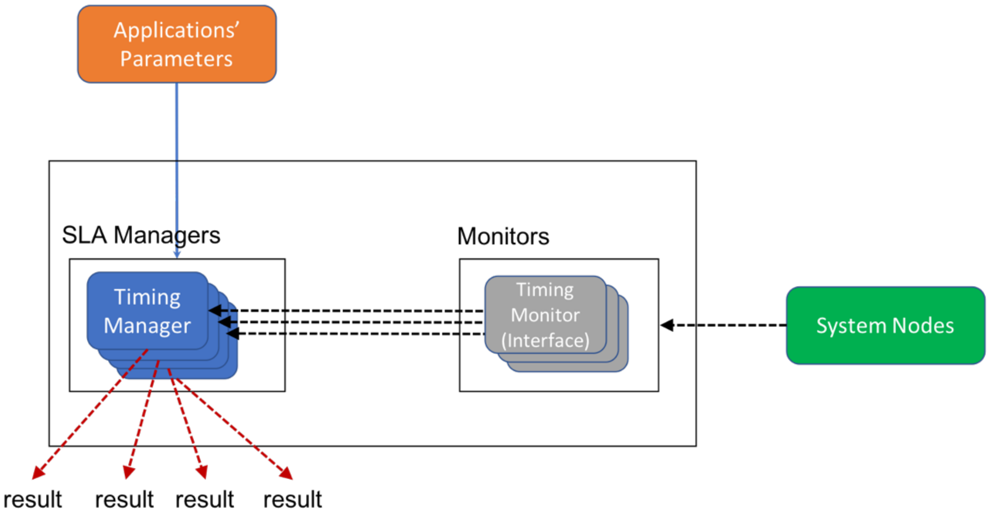


Figure 14. Online NFR analysis of each property in isolation.

In a second phase, described in Figure 15, a holistic online NFR analysis will be performed. Each individual SLA Manager is iteratively used by the Global Manager, which will provide a single change result.

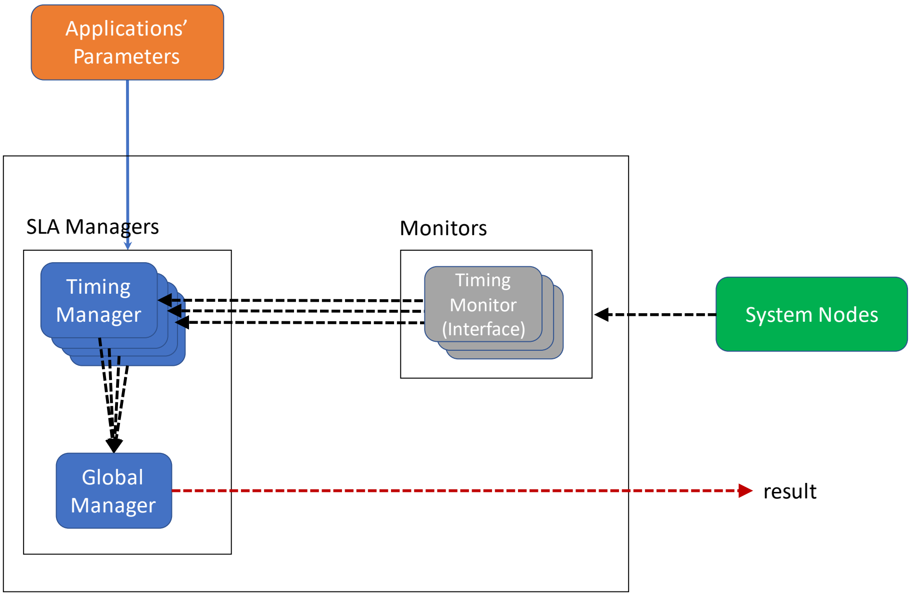


Figure 15. Online holistic NFR analysis.

In the first phase of the ELASTIC project, where an isolated NFR analysis will be performed, each SLA manager will provide the generic interface described in Table 4. `Platform_Information` denotes the configuration of the ELASTIC nodes, Services lists the currently executing services, `QoS_Parameters` provides the services' desired level of service, and `Runtime_Information` indicates the monitored information acquired during runtime. This analysis will output a possible new system configuration, describing the set of required changes to the current system configuration or an empty set if the system is currently satisfying with the required QoS levels. A simple feasibility check is also available.

Description	API
Create a System Configuration	`SystemConfiguration system = createConfiguration(Platform_Information, Services,QoS_Parameters)`
Isolated NFR property feasibility check, when a change is detected in a service execution	`Boolean feasible = checkFeasibility(Runtime_Information)`
Isolated NFR property analysis, returning the set of required changes to the current system configuration (it can be empty if QoS levels are already satisfied)	`SystemConfiguration changes = isolatedAnalysis(Runtime_Information)`
Configuration Change Information	`SystemConfiguration system = changeConfiguration(Service, QoS_Parameters)`
Platform Change Information	`SystemConfiguration system = changeConfiguration(Platform_Information)`

Table 4. Isolated NFR analysis API

In the second phase of the ELASTIC project, where a holistic NFR analysis will be performed, each individual SLA Manager is iteratively used by the Global Manager, considering the potential trade-offs between performance, predictability, energy-efficiency, communication quality and security. The API of the Global Manager is described in Table 5, being similar to the one in Table 4.

Description	API
Create a System Configuration	`SystemConfiguration system = createConfiguration(Platform_Information, Services,QoS_Parameters)`
Holistic NFR property feasibility check, when a change is detected in a service execution	`Boolean feasible = checkFeasibility(Runtime_Information)`
Holistic NFR property analysis, returning the set of required changes to the current system configuration (it can be empty if QoS levels are already satisfied)	`SystemConfiguration changes = holisticAnalysis(Runtime_Information)`
Configuration Change Information	`SystemConfiguration system = changeConfiguration(Service, QoS_Parameters)`
Platform Change Information	`SystemConfiguration system = changeConfiguration(Platform_Information)`

Table 5. Holistic NFR analysis API

A note that the SLA Managers in the second phase will implement analyses which already take into consideration the multiple non-functional properties.

5.7 Impact in the ELASTIC Software Architecture

The use of the NFR tools (particularly the online components) requires that the ELASTIC software architecture is able to:

- Provide information on the resource usage and application execution in the nodes;
- Receive from the NFR Monitor the change results described in the previous section.

Figure 16 and Figure 17 present the relation of the NFR tool with the other components of the ELASTIC software architecture. The tool receives monitored data information from the nodes: Figure 16 provides edge examples with the NuvlaBox (D3.1 [55]) and the ELASTIC hybrid fog computing nodes (D5.1 [56]) while Figure 17 provides an example with Nuvla (D3.1 [55]) in the cloud side. The analysis results are provided to the COMPs orchestrator and the data analytics services. Figure 18 provides a further example where a Docker Swarm manager is integrated in the architecture.

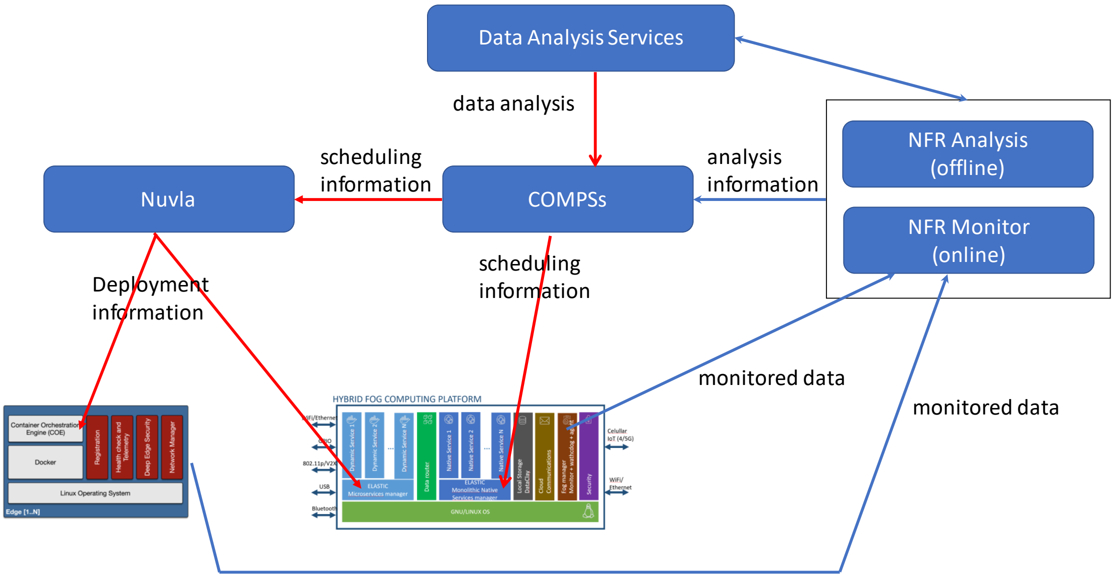


Figure 16. NFR Monitor relation with the ELASTIC Software Architecture (edge side)

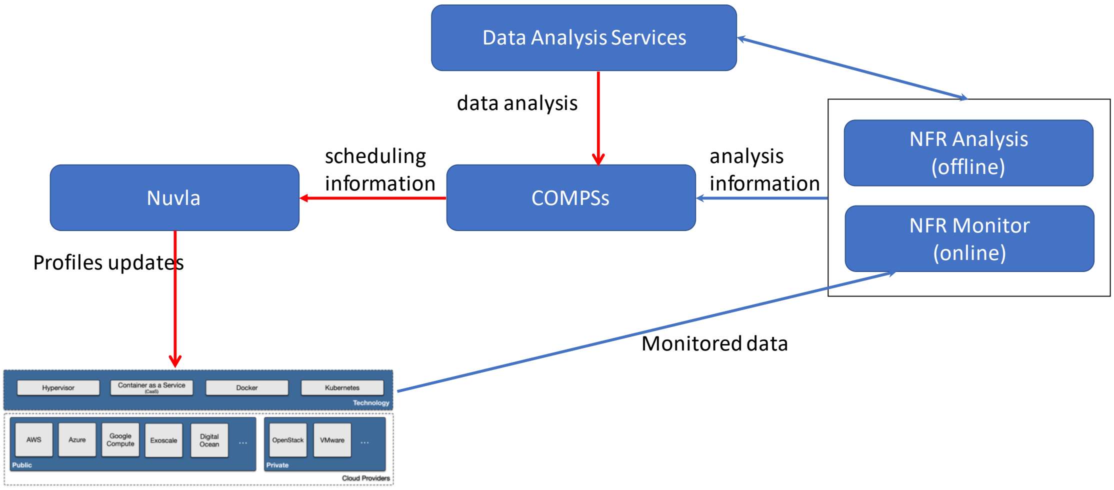


Figure 17. NFR Monitor relation with the ELASTIC Software Architecture (cloud side)

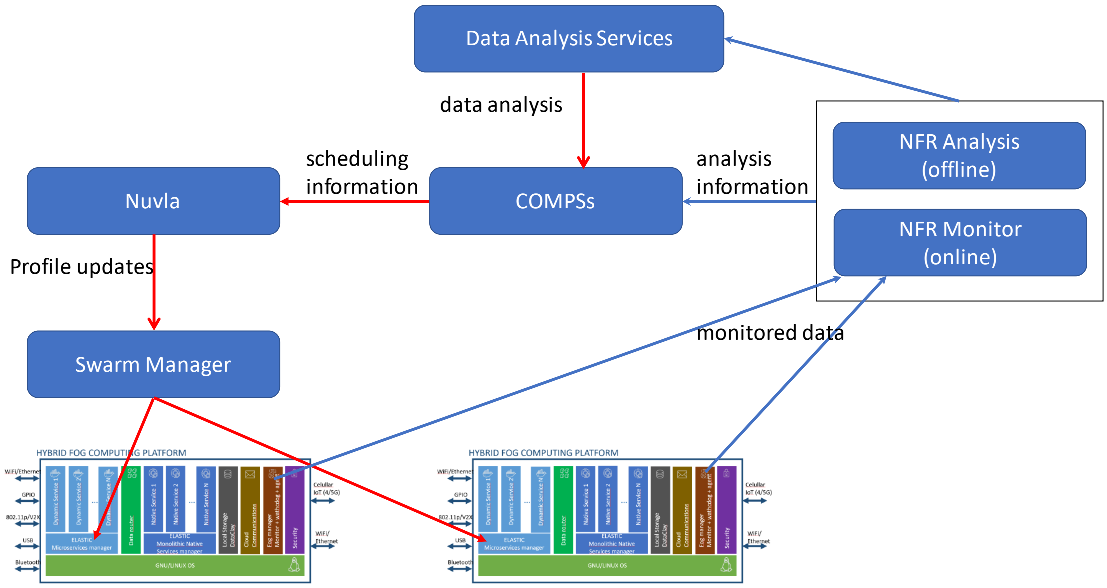


Figure 18. Example with Docker Swarm Manager

6 Results and impacts

The document provides the results of task 4.1 of the ELASTIC project: the target at MS1, at month 6, was a first release of the technical constraints imposed by the non-functional use-case properties and associated evaluation metrics.

The work presented in this document will be used to drive the development of the mechanisms required to monitor and deal with the non-functional properties, to be performed in tasks 4.2 and 4.3, as well as their integration in the ELASTIC software architecture (tasks 3.2 and 3.3).

7 Conclusion

This document presented the non-functional requirements identified from the analysis of the smart applications targeted by the project and discussed the technical constraints which are imposed into the ELASTIC software architecture.

The requirements have been extracted both from the analysis of the use cases provided to the project activities, as well as from other related application domains (smart manufacturing, automotive and avionics). The description of the requirements includes the concrete criteria and metrics, as well as the means of verification, for the evaluation of the project results.

From the analysis of these requirements, the document also specified the software components to be developed in the consequent tasks of the WP (4.2 and 4.3), as well as the mechanisms which need to be provided by the overall software architecture.

Acronyms and Abbreviations

AD - Autonomous Driving

ADAS - Advanced Driver Assistance Systems

COTS - Commercial Off-The-Shelf

D - Deliverable

DARPA - Defense Advanced Research Projects Agency

DoA - Description of Action (Annex 1 of the Grant Agreement)

EB - Executive Board

EC - European Commission

GDPR - General Data Protection Regulation

GPS - Global Positioning System

HPC - High Performance Computing

IIoT - Industrial internet of things

IMA - Integrated Modular Avionics

IoT - Internet of things

KPI - Key Performance Indicator

LIDAR - Light Detection And Ranging

M - Month

MS - Milestones

NFR - Non-Functional Requirement

OEM - Original Equipment Manufacturer

QoS - Quality of Service

SLA - Service Level Agreement

SWaP - Size, Weight and Power

V2I - Vehicle-to-Infrastructure

V2V - Vehicle-to-Vehicle

V2X - Vehicle-to-Vehicle and Vehicle-to-Infrastructure

WCET - Worst-Case Execution Time

WP - Work Package

References

[1] ELASTIC Consortium, Deliverable D1.1: “Use case requirement specification and definition”, May 2019

[2] Definition of a Research and Innovation Policy Leveraging Cloud Computing and IoT Combination, European Commission DG Communications Networks, Content & Technology, 2014 ISBN 978-92-79-47760-7 DOI:10.2759/38400.

[3] Report AIOTI WG11 - Smart manufacturing, Alliance for Internet of Things Innovation 2015.

[4] Embedded/ Cyber-Physical Systems ARTEMIS Major Challenges: 2014-2020 2013, DRAFT Addendum to the ARTEMIS-SRA 2011.

[5] Lou, D., Holler, J., Whitehead, C., Germanos, S., Hilgner, M. & Qiu, W. (2018). Industrial Networking Enabling IIoT Communication. An Industrial Internet Consortium White Paper.

[6] Good Practices for Security of Internet of Things in the context of Smart Manufacturing, November 2018. European Union Agency for Network and Information Security (ENISA), ISBN: 978-92-9204-261-5, DOI: 10.2824/851384.

[7] H2020-FoF-2014-1-637066: CREMA Cloud-based Rapid Elastic Manufacturing (2014-2017).

[8] Henning Butz, Open integrated modular avionic (IMA): State of the art and future development road, Technical report, Department of Avionic Systems, Airbus Deutschland GmbH, 2010.

[9] Joseph Huysseune, Philippe Palmer, NEVADA-PAMELA-VICTORIA: Towards the definition of new aircraft electronic systems, Air & Space Europe, Volume 3, Issues 3–4, 2001

[10] RTCA, DO-297: Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations, 2005

[11] RTCA, DO-178B: Software Considerations in Airborne Systems and Equipment Certification, 1992

[12] RTCA, DO-178C: Software Considerations in Airborne Systems and Equipment Certification, 2011

[13] RTCA, DO-254: Design Assurance Guidance for Airborne Electronic Hardware, 2000

[14] SAE, ARP4754: Certification Considerations for Highly-Integrated or Complex Aircraft Systems, 1996

[15] ARINC, Avionics Application Software Standard Interface: ARINC Specification 653 Part 1, 2006.

[16] ARINC, Design Guidance for Integrated Modular Avionics, 1997

[17] ARINC, Digital Data Transmission System, 1977

[18] ARINC, Multi-Transmitter Data Bus, 1999

[19] NPFC, MIL-STD-1553: Digital Time Division Command/Response Multiplex Data Bus, 1986

[20] ARINC, Aircraft Data Network Part 1, 2002

[21] ARINC, Aircraft Data Network Part 7 Avionics Full Duplex Switched Ethernet (AFDX) Network, 2005

[22] RTCA, Airworthiness Security Process Specification, 2014

[23] RTCA, Airworthiness Security Methods and Considerations, 2014

[24] RTCA, Guidance for Installation of Automatic Flight Guidance and Control Systems (AFGCS), 2012

[25] FAA, Recommendations regarding ASISP rulemaking, policy, and guidance on best practices for airplanes and rotorcraft including both certification and continued airworthiness, 2015

[26] EASA, Notice of Proposed Amendment Aircraft cybersecurity, 2019

[27] EASA, Flight Data Monitoring on ATR Aircraft, 2016

[28] ITU-T, https://www.itu.int/en/ITU-T/focusgroups/ac/Pages/default.aspx, 2014

[29] ITU-T, Existing and emerging technologies of cloud computing and data analytics, 2016

[30] ITU-T, Key findings, recommendations for next steps and future work, 2016

[31] Society of Automotive Engineers (SAE). “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles – Standard J3016_201806”, 2018.

[32] B. Schoettle, M. Sivak. “A Survey of Public Opinion about Autonomous and Self-driving Vehicles in the US, the UK, and Australia”. Report No. UMTRI-2014-21. University of Michigan, Ann Arbor, Transportation Research Institute, 2014.

[33] M. Campbell, M. Egerstedt, J.P. How, R.M. Murray. “Autonomous driving in urban environments: approaches, lessons and challenges”. Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 368:4649–4672, 2010. DOI: 10.1098/rsta.2010.0110

[34] M. Gerla. “Vehicular Cloud Computing”. In Proceedings of the 11th Annual Mediterranean on AdHoc Networking Workshop (Med-Hoc-Net), pp. 152–155, 2012 DOI: 10.1109/MedHocNet.2012.6257116

[35] S. Kumar, S. Gollakota, D. Katabi. “A Cloud-assisted Design for Autonomous Driving”. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, pp. 41–46, 2012. DOI: 10.1145/2342509.2342519

[36] M. Gerla, E.K. Lee, G. Pau, U. Lee. “Internet of Vehicles: From Intelligent Grid to Autonomous Cars and Vehicular Clouds”. In Proceedings of the IEEE World Forum on Internet of Things (WF-IoT), pp. 241–246, 2014. DOI: 10.1109/WF-IoT.2014.6803166

[37] L. A. Maglaras, A. H. Al-Bayatti, Y. He, I. Wagner, H. Janicke. “Social Internet of Vehicles for Smart Cities”. Journal of Sensor and Actuator Networks, 5(1):1–22, 2016. DOI: 10.3390/jsan5010003

[38] Liu, S.; Tang, J.; Wang, C.; Wang, Q.; Gaudiot, J.L. “A Unified Cloud Platform for Autonomous Driving”. *Computer,* 50: 42–49, 2017

[39] Raúl Borraz, Pedro J. Navarro, C. Fernández, Pedro María Alcover. “Cloud Incubator Car: A Reliable Platform for Autonomous Driving”. Applied Sciences, 8(2):303, 2018. DOI: 10.3390/app8020303

[40] P. Mell, T. Grance. “The NIST Definition of Cloud Computing”. Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology Gaithersburg, 2011

[41] Santos, M., Ambiel, V., Acras, M., and Gliwa, P., “On the Timing Analysis at Automotive Real-Time Embedded Systems”, SAE Technical Paper 2017-01-1618, 2017. DOI: 10.4271/2017-01-1618

[42] Amit Kumar Singh, Piotr Dziurzanski, Hashan Roshantha Mendis, and Leandro Soares Indrusiak. “A Survey and Comparative Study of Hard and Soft Real-Time Dynamic Resource Allocation Strategies for Multi-/Many-Core Systems”. ACM Computing Surveys, 50:2, 2017. DOI: 10.1145/3057267

[43] O. Henniger, L. Apvrille, A. Fuchs, Y. Roudier, A. Ruddle and B. Weyl. "Security requirements for automotive on-board networks". In Proceedings of the 29th International Conference on Intelligent Transport Systems Telecommunications, pp. 641-646, 2009. DOI: 10.1109/ITST.2009.5399279

[44] Lu, M., Wevers, K., Van der Heijden, R. “Technical feasibility of advanced driver assistance systems (ADAS) for road traffic safety”. Transportation

Planning and Technology, vol. 28, no. 3, pp. 167-187, 2005. DOI: 10.1080/03081060500120282

[45] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, M. Zaharia. “A View of Cloud Computing”. Communications of the ACM, 53(4):50–58, 2010. DOI: 10.1145/1721654.1721672

[46] Z. Li, H. Zhang, L. O’Brien, R. Cai, S. Flint. “On Evaluating Commercial Cloud Services: A Systematic Review”. Journal of Systems and Software, 86(9):2371–2393, 2013. DOI: 10.1016/j.jss.2013.04.021

[47] Gabrielle Coppola, Esha Dey. “Driverless Cars Are Giving Engineers a Fuel Economy Headache”. Bloomberg, October 2017. Available at: https://www.bloomberg.com/ news/articles/2017-10-11/driverless-cars-are-giving-engineers-a-fuel-economy-headache

[48] Andy Greenberg. “The Jeep Hackers are Back to Prove Car Hacking Can Get Much Worse”, Wired, January 2018. Available at: https://www.wired.com/2016/08/jeep-hackers-return-high-speed-steering-acceleration-hacks/

[49] Jared Gall. “Can a Connected Car Ever Be Safe from hacking?”, Car and Driver, October 2017. Available at: https://www.caranddriver.com/features/a15079914/ can-a-connected-car-ever-be-safe-from-hacking-feature/

[50] Rajeev Thakur. “Infrared Sensors for Autonomous Vehicles”. Recent Development in Optoelectronic Devices, IntechOpen, 2017. DOI: 10.5772/intechopen.70577

[51] Christos Katrakazas, Mohammed Quddus, Wen-Hua Chen, Lipika Deka. “Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions”. Transportation Research Part C: Emerging Technologies, 60, pp. 416-442, Elsevier, 2015. DOI: 10.1016/j.trc.2015.09.011

[52] Andrew Walker. “Hard real-time motion planning for autonomous vehicles”. PhD thesis, Swinburne University of Technology, Melbourne, Australia, 2011

[53] CENELEC, EN 50159 : Railway applications - Communication, signalling and processing systems - Safety-related communication in transmission systems, 2011

[54] EU General Data Protection Regulation, https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en

[55] ELASTIC Consortium, Deliverable D3.1: “Software architecture requirements and integration plan”, May 2019

[56] ELASTIC Consortium, Deliverable D5.1: “General requirements of the fog architecture”, May 2019

Annex A - NFR Questionnaire

Timing requirements

Timing requirements:	
Control loops are	□ Local □ Distributed
Real-time requirements are	□ End-to-end □ Break-down into requirements at node and communication level
Are there hard real-time requirements? *Hard real-time: the requirement must be met in all executions*	□ YES □ NO
Are there firm real-time requirements? *Firm real-time: if the requirement is not met the execution has no value*	□ YES □ NO
Are there soft (Quality of Service) requirements?	□ Ratio of accepted failure □ Maximum admissible number of consecutive failures □ Average response time Other: ....
Response-time requirements are in the order of	Select all that fit □ μsec □ msec □ sec □ min Other: ....
Modes of operation:	
Do systems have modes of operation?	□ YES □ NO
Modes of operation are related to	□ Location of train in relation to intersections/areas □ Speed of train □ Detection of obstructions Other: ....

Local Scheduling:	
Are there computing nodes with real-time schedulers?	□ YES □ NO
If yes, which type of real-time schedulers are used?	Select all that fit: □ Priority-based preemptive scheduler □ Deadline-based preemptive scheduler □ Round-robin non-preemptive □ Server-based scheduler □ Cooperative non-preemptive Other: ………………………………………
Do nodes require mixed-criticality scheduling?	□ YES □ NO
Do nodes require multi-core scheduling?	□ Global scheduling □ Partitioned □ Semi-Partitioned
Global Scheduling:	
Applications/components/partitions are allocated to nodes	□ Statically during deployment □ Dynamically during execution If dynamically, which approach(es) are used to perform the allocation: ……………………………………… ……………………………………… ………………………………………
Execution time:	
Are there applications that require worst-case execution time analysis	□ YES □ NO
Which approach is used to derive execution time values	Select all that fit: □ Static analysis □ Measurements □ Knowledge of previous systems Other: ………………………………………

For worst-case analysis, do you use hardware knowledge?	Select all that fit: □ Caches, memory hierarchy □ Buses access, NoC □ Pipelines, intra-core mechanisms Other: ………………………………………
Are there applications requiring execution time analysis executing in computing nodes with multicore/manycore processors?	□ YES □ NO If yes, which approach is used for multicore analysis: ……………………………………… ……………………………………… ………………………………………
Data Sharing:	
Which protocols are used to shared data in applications	Select all that fit: □ Priority Inheritance □ Priority Ceiling Other: ………………………………………
Synchronization:	
Should the nodes/sensors use the same synchronization common source?	□ YES □ NO If yes, specify the maximum time difference relative to a synchronization source: ………………………………………

Energy requirements

Processing nodes with energy limitations:	
Are there embedded nodes with energy limitations?	Select all that fit □ Power dissipation limitations □ On batteries □ Communication energy consumption Other: ………………………………………

Are there cloud servers with energy requirements?	Select all that fit □ Energy budget □ Cost model Other: …………………………………………
Mechanisms for energy handling	
Do nodes use energy-aware scheduling approaches	Select all that fit □ Dynamic Voltage and Frequency Scaling (DVFS) □ Dynamic Power Management (DPM) Other: …………………………………………
Do nodes use energy monitoring approaches?	Select all that fit □ Measurements □ Performance counters Other: …………………………………………
Do nodes use static slack reclaiming?	□ YES □ NO
Do nodes use dynamic slack reclaiming?	□ YES □ NO
Do nodes use offline DPM?	□ YES □ NO
Do nodes use online DPM?	□ YES □ NO
Do nodes use offline speed scaling?	□ YES □ NO
Do nodes use online speed scaling?	□ YES □ NO
Do nodes use multicore energy mechanisms?	Select all that fit □ Per-core DVFS □ Per-task DVFS Other: …………………………………………

Are there modes of operation that modify the profile of energy use (e.g. energy saving when train is in some parts of the track)?	□ YES □ NO If yes, specify: ………………………………………… ………………………………………… …………………………………………

Communication

Control	□ Centralized □ Distributed
The tram shall be able to communicate through wireless network communications with:	Select all that fit: □ other trams □ traffic sensors □ cloud □ actuators Other: …………………………………………
Where is the device(s) that communicates with the cloud located?	Select all that fit: □ In the tram □ In the trams garage/shed □ In the Control Centre Other: …………………………………………
Why is the data sent to the cloud?	Select all that fit: □ To be stored and later monitored □ To be processed □ To notify about events/alarms Other: …………………………………………
When/Where is the data sent to the cloud?	Select all that fit: □ Immediately □ In the tram shed □ Depends on the urgency Other: …………………………………………
Which device is the receptor of the output of the data processing in the cloud?	Select all that fit: □ Tram central node □ Tram' actuators □ Traffic actuators □ Control Centre Other: …………………………………………
Does the network provide robust partitioning guarantees?	□ YES □ NO
Does the network perform store-and-forward action?	□ YES □ NO

Are network tables updated during live network operation?	□ YES □ NO
Is the network used to download software, constants, or databases?	□ YES □ NO If so, is the network required to continue communicating essential? □ YES □ NO
Information:	
Type of information	Select all that fit: □ Voice □ Data □ Image Other: ....
Type of data	Select all that fit: □ Monitoring □ Control □ Files, documents Other: ....
Which data is critical	Select all that fit: □ Monitoring □ Control □ Files, documents Other: ....
Most common information to be transmitted (e.g. status, temperature, track change command, error table etc.)	....
Data rate (bps)	Maximum .... Typical ....
Broadcast/multicast messages	□ Mandatory □ Recommendable □ Unnecessary
Acknowledge messages	□ Mandatory □ Recommendable □ Unnecessary
Priority in messages	□ Mandatory □ Recommendable □ Unnecessary

Message timestamping	□ Mandatory □ Recommendable □ Unnecessary
Geo-localization of the computing node/sensor which produces the message	□ Mandatory □ Recommendable □ Unnecessary
ID of the computing node/sensor which produces the message	□ Mandatory □ Recommendable □ Unnecessary
Communication protocol:	
List the communication protocols to be used within the use-case (wired and wireless)	…………………………………………… …………………………………………… …………………………………………… ……………………………………………
List the communication protocols to be used to communicate between nodes and the Gateway (the element that gives access to the cloud)	…………………………………………… …………………………………………… …………………………………………… ……………………………………………
Do you need a specific protocol between the Gateway and the cloud?	□ YES □ NO If yes, specify: ……………………………………………
Communication Quality:	
What is the level of integrity needed for the data communication?	□ Failure rate for the wired/wireless communication □ Global (end to end)
What tools and techniques were used to assess system throughput, latency, jitter and other performance?	…………………………………………… …………………………………………… …………………………………………… ……………………………………………
Is there any redundancy mechanism for the dataflow?	□ YES □ NO If yes, specify: ……………………………………………
Does the dataflow need to respect Ordinal and/or cardinal timing?	□ Ordinal □ Cardinal

What mechanisms prevent or detect network malfunction (e.g., loss or corruption of data, corrupted addresses)?	……………………………………………… ……………………………………………… ……………………………………………… ………………………………………………
What integrity mechanisms are used to protect the network from failure on the communication path?	……………………………………………… ……………………………………………… ……………………………………………… ………………………………………………
Does the intermediate stage perform recalculation of integrity check sequences?	□ YES □ NO
Does the network meet the required integrity values (undetected error probabilities, Hamming Distance) for the worst-case error pattern requirements?	□ YES □ NO
Does the network need to support system architectures to achieve fault-tolerance?	□ YES □ NO
Does the network provide mechanisms to contain host software failures?	□ YES □ NO If yes, specify network mechanisms that ensure this containment: ……………………………………………… ………………………………………………
Does the network technology implement adequate checking mechanisms to ensure the run-time integrity of the routing tables?	□ YES □ NO
Are network configuration data tables stored with sufficient integrity?	□ YES □ NO
Do you have network channel availability requirements?	□ YES □ NO
What mechanisms exist to detect erroneous message forwarding (*e.g. the forwarding of old messages or messages to incorrect addresses)*?	……………………………………………… ……………………………………………… ……………………………………………… ………………………………………………
Does the network deliver valid messages within bounded latency with sufficient probability despite expected error rates?	□ YES □ NO
Does the network manage QoS policy and priority for your Dataflow?	□ YES □ NO
Other:	

Is it necessary to distinguish the different nodes within a system?	□ YES □ NO

Security

General Security Requirements:	
Order of priority for security requirements:	□ confidentiality-integrity-availability □ availability-integrity-confidentiality Other: …………………………………………
The system shall be able to switch between different security levels according to the power policy.	□ YES □ NO
Communication Security Requirements:	
Are there identified security risks (pre and post application of ELASTIC solutions)?	□ YES □ NO
Any risk analysis?	□ YES □ NO
Is there application of the IEC 50159 standard? (PS: Cybersecurity)	□ YES □ NO
Do your systems have current or future security requirements that have to be supported by the network?	□ YES □ NO If yes, specify: …………………………………………
Can security weaknesses adversely affect network dependability (safety)?	□ YES □ NO
Does the network technology support sufficient secure services for user and application authentication?	□ YES □ NO
Does the network technology support secured-data transmission mechanism?	□ YES □ NO
Does the network support multilevel security?	□ YES □ NO If yes, specify the number: …………………………………………

Is network configuration data protected and secured during deployment and during load?	□ YES □ NO
What is the size of the mass storage for the security log	…………………………………………
What is the life time of the saved data	…………………………………………
Application Services Requirements:	
ELASTIC must support the security of all data flows	□ YES □ NO
ELASTIC must provide secure communications: secure exchange of messages as well as message encryption	□ YES □ NO
ELASTIC must provide means for ensuring data security and persistency	□ YES □ NO
ELASTIC must provide an infrastructure to securely store data within the Edge	□ YES □ NO
ELASTIC must provide an infrastructure to securely store data within the Cloud	□ YES □ NO
ELASTIC must provide mechanisms to secure access/retrieve/save/modify data in data storage	□ YES □ NO
Sensitive data should be stored encrypted to prevent security flaws	□ YES □ NO
Security must ensure that only apps with permission can subscribe to certain data	□ YES □ NO
ELASTIC should support different user categories (roles) and corresponding privileges in order to manage access control.	□ YES □ NO
Security must support the definition of different access rules (security privileges) to data to provide mechanisms to access/save/modify data in data storages	□ YES □ NO
The security component needs for each access credentials to make clear which user accesses which resource	□ YES □ NO
Every software component as well as the system itself shall have a session time-out for inactivity. Once this time-out is triggered, the password shall be requested to operate again.	□ YES □ NO

ELASTIC platform must provide customised UI according to the privileges of each role	□ YES □ NO
Only a "super-user" shall have rights to manage the configuration of the system (user management, hardware installation, etc).	□ YES □ NO
Notifications should only be sent to authorised personnel to prevent security risks	□ YES □ NO
Accessing rules management must be from authorised personnel only to ensure security	□ YES □ NO
The software installed in the system shall be correctly managed in a repository using at least a Control Version System. It shall be possible to come back to a previous software configuration.	□ YES □ NO

Security CIDT requirements at device level (especially on tram and edge)	
Is any mechanism put in place?	Select all that fit: □ Distinct users/privileges □ Secure Boot □ TPM Other: ....
Data Criticality	
Do you have requirements on Data?	□ Confidentiality □ Integrity □ Availability □ Trace

Standards and Regulations

Please, identify the standards that are significant for the Use Case.

Electrical and/or electronic Safety standard: IEC 61508 (2011)	
61508-1	□ YES □ NO □ PROFILE
61508-2	□ YES □ NO □ PROFILE

61508-3	□ YES □ NO □ PROFILE
61508-4	□ YES □ NO □ PROFILE
61508-5	□ YES □ NO □ PROFILE
61508-6	□ YES □ NO □ PROFILE
61508-7	□ YES □ NO □ PROFILE

Railway Safety standard	
IEC 50126(2017)	□ YES □ NO □ PROFILE
IEC 50126-1, 50126-2	□ YES □ NO □ PROFILE
IEC 50128(2011)	□ YES □ NO □ PROFILE
IEC 50129(2016)	□ YES □ NO □ PROFILE

Industrial Control Systems Cyber security standard (IEC 62443)	
62443-1.1(2009)	□ YES □ NO □ PROFILE
62443-2.4(2016)	□ YES □ NO □ PROFILE
62443-3.2(2018), 62443-3.3(2013)	□ YES □ NO □ PROFILE
62443-4.1(2018), 62443-4.2(2017)	□ YES □ NO □ PROFILE

Safety-related communication in transmission systems *Railway applications. Communication, signaling and processing systems. Safety-related communication in transmission systems.*	
IEC 50159 (PS : Cybersecurity)	□ YES □ NO □ PROFILE

IT Security Standard imposed by Italy/European laws	
ISO 15408	□ YES □ NO □ PROFILE
ISO 27000	□ YES □ NO □ PROFILE
Other:	□ YES □ NO □ PROFILE

Industrial Cybersecurity	
Shall ELASTIC adhere to any industrial cyber security standard?	If yes, specify: ....
Must any Functional Security Level be fulfilled?	□ YES □ NO If yes, specify: ....

GDPR	
ELASTIC shall satisfy GDPR for managing personal data	□ YES □ NO