Project Acronym: SecureIoT
Grant Agreement number: 779899 (H2020-IoT03-2017 - RIA)
Project Full Title: Predictive Security for IoT Platforms and Networks of Smart Objects

DELIVERABLE

Deliverable Number	**D5.1**
Deliverable Name	**IoT Risk Assessment and Mitigation as a Service_First version**
Dissemination level	Public
Type of Document	Demonstrator
Contractual date of delivery	31/12/2018
Deliverable Leader	ATOS
Status & version	Final - V1.0
WP / Task responsible	WP5 / T5.1-INTRASOFT, T5.5-ATOS
Keywords:	Risk assessment, mitigation, predictive security, Security as a Service (SECaaS)
Abstract (few lines):	The present deliverable is devoted to the description and the documentation of the risk assessment (SECaaS) service of the SecureIoT platform. The deliverable is of a "demonstrator' nature and includes information about the initial proof-of-concept implementation of the service, which is destined to be enhanced and fine-tuned in subsequent releases of the deliverable, namely deliverables D5.2 and D5.3 that are linked to this one.

Deliverable Leader:	Daniel Calvo (ATOS)
Contributors:	Nikos Kefalakis (INTRASOFT), Jose Ruiz (ATOS), Daniel Calvo (ATOS), Giannis Ledakis (SiLO), Septimiu Nechifor (SIEMENS)
Reviewers:	Sofianna Menesidou (UBI), John Soldatos (AIT)
Approved by:	Stylianos Georgoulas (INTRASOFT)

Executive Summary

SecureIoT aims to design and implement a data-driven platform for security monitoring and security automation of Internet-of-Things (IoT) systems. The platform will support various security scenarios by providing a range of security services that will be accessible through appropriate APIs (Application Programming Interfaces) or as-a-service. In this context, it will provide "build-in" security services, which will be deployed and delivered on the basis of a SECaaS (Security as a Service) modality. These "build-in" services are designed and implemented as part of WP5 and include a risk assessment service, a compliance auditing services, a security programming support service for IoT developers, as well as a security knowledge base for IoT systems.

The present deliverable is devoted to the description and the documentation of the risk assessment (SECaaS) service of the SecureIoT platform. The deliverable serves as a "demonstrator' nature and includes information about the initial proof-of-concept implementation of the service, which is destined to be enhanced and fine-tuned in subsequent releases of the deliverable, namely deliverables D5.2 and D5.3, which will describe the refinement, evolution and conclusions of the work presented here.

Risk assessment is in general one of the primary security functions performed by security stakeholders, as it provides insights on the vulnerabilities of the various assets and their relevant importance to the IoT system at hand. In SecureIoT, risk assessment functionalities are performed by a risk assessment engine, which is integrated with other modules of the SecureIoT architecture such as data collection, data analytics and security automation modules. In particular, SecureIoT integrates and customizes a readily available risk assessment component, which facilitates the bootstrapping of the developments. The risk assessment service offers functionalities in two main directions: (i) Risk assessment configuration functionalities, which provide the means for setting up the parameters that drive the operation of the risk assessment engine; and (ii) Run-time operations of the risk assessment engine, which provide the means for implementing risk assessment workflows. The implementation of risk assessment workflows enables the identification and presentation of various risks to the end-users of the service along with appropriate risk mitigation activities. The latter are streamlined with proper security policies that drive the operation of these services in the scope of the SecureIoT platform. A typical operation of the risk assessment services involves the configuration of the risk assessment engine, an instantiation of the engine based on given parameters, the presentation of the risk assessment results to an end-user (e.g., a security expert or security systems operator), as well as the implementation of one or more of the suggested risk mitigation actions through the SecureIoT platform. The above-listed functionalities are accessible through proper APIs, which are detailed in this deliverable.

The implementation of the risk assessment workflows is based on security information collected and analysed by other modules of the SecureIoT platform, notably modules implemented in WP3

and WP4 of the project. As such the risk assessment engine is an integrated service of the SecureIoT platform, which is used as a basis for validating the integration of multiple cybersecurity-oriented modules developed in the project. In this context, subsequent implementations of the SecureIoT risk assessment SECaaS will be based on more advanced versions of the data collection and data analytics prototypes.

Overall, the present deliverable provides detailed specifications for the risk assessment SECaaS of the SecureIoT platform, including functional specifications for its configuration and run-time operations as well as relevant APIs. These specifications provide a sound basis for advancing the initial MVP (Minimum Viable Product) implementation of this deliverable, as part of its subsequent versions.

Document History			
Version	**Date**	**Contributor(s)**	**Description**
0.00	03/10/2018	Daniel Calvo (ATOS)	Consolidated ToC for WP5 deliverables.
0.01	05/10/2018	Nikos Kefalakis (INTRASOFT)	RA&MS Logical view and Task dependencies
0.02	11/10/2018	Nikos Kefalakis (INTRASOFT)	RA&MS Process/use case diagrams, Draft API specification
0.03	14/12/2018	Nikos Kefalakis (INTRASOFT)	Review of initial ToC, new specific sections for T5.1, assignment of responsibilities.
0.04	17/12/2018	Nikos Kefalakis (INTRASOFT)	Updated RA&MS Architecture Views
0.05	18/12/2018	Nikos Kefalakis (INTRASOFT)	Updated RAE API specifications, Added adapted PoC RAE component
0.06	19/12/2018	Daniel Calvo (ATOS)	Continuous integration, testing and deployment section.
0.07	19/12/2018	Jose Ruiz (ATOS)	Added ATOS RAE as background technologies, various edits
0.08	19/12/2018	Nikos Kefalakis (INTRASOFT)	Updated introduction and analysis of requirements
0.09	20/12/2018	Jose Ruiz (ATOS)	Update with contribution to section 2, section 4 and section 7
0.10	20/12/2018	Nikos Kefalakis (INTRASOFT)	Various Edits and Fine-Tuning, Updated Conclusions/Executive Summary, prepared the document for the internal review
0.11	21/12/2018	Sofianna Menesidou (UBI)	Internal Quality & Technical Review 1
0.12	26/12/2018	John Soldatos (AIT)	Internal Quality & Technical Review 2
0.13	27/12/2018	Nikos Kefalakis (INTRASOFT)	Addressed reviewers' comments, various Edits and Fine-Tuning, Prepared for final submission
1.00	28/12/2018	Stylianos Georgoulas (INTRASOFT)	Deliverable approval and submission to EC

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Table of Contents

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Table of Figures

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

List of Tables

Definitions, Acronyms and Abbreviations

Acronym	Title
API	Application programming interface
CD	Continues Development
CI	Continues Integration
CO	Confidential, only for members of the consortium (including Commission Services)
CR	Change Request
CT	Continues Testing
CVSS	Common Vulnerability Scoring System
D	Demonstrator
DDoS	Distributed Denial-of-Service
DL	Deliverable Leader
DoA	Description of Action
DoS	Denial-of-Service
Dx	Deliverable (where x defines the deliverable identification number e.g. D1.1.1)
EU	European Union
IDS	Intrusion Detection Systems
IoT	Internet of Things
MSx	project Milestone (where x defines a project milestone e.g. MS3)
MVP	Minimum Viable Product
Mx	Month (where x defines a project month e.g. M10)
NIST	National Institute of Standards and Technology
O	Other
OBU	On Board Unit
P	Prototype
P&CIR	Probes & Common Interoperability Registry
PC	Project Coordinator
PEP	Policy Enforcement Point
PoC	Proof of Concept
PP	Restricted to other programme participants (including the Commission Services)
PU	Public
R	Report
RA	Risk Assessment
RA&MS	Risk Assessment & Mitigation Services
RAD	Risk Assessment Dashboard
RAE	Risk Assessment Engine
RE	Restricted to a group specified by the consortium (including Commission Services)
REST	Representational State Transfer
SCM	Source Code Management
SIEM	Security Information and Event Management
SKB	Security Knowledge Base

V2I	Vehicle to Infrastructure
V2Internet	Vehicle to Internet
V2V	Vehicle to Vehicle
WP	Work Package

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

1 Introduction

The main goal of the SecureIoT project is to introduce, validate and promote a novel approach to the security of IoT applications, which emphasizes a timely, predictive and intelligent approach to the identification and mitigation of security threats and incidents. One of main characteristics of this approach will be its ability to deal with smart objects, while at same time supporting security interoperability in supply chain scenarios that involve multiple IoT systems and platforms with diverse security capabilities. The project will reflect its approach to an architectural concept, which will serve as a basis for implementing predictive and intelligent security systems. Based on this overarching architectural concept, the project will develop concrete security services, like the Risk Assessment & Mitigation services, that will be validated in the scope of the project's use cases.

In this context, the purpose of the present deliverable is to introduce the Risk Assessment & Mitigation services framework. It lists the requirements and dependences that drive the design and implementation of the framework. Additionally, it provides a detailed description of the RA&MS (Risk Assessment & Mitigation Services) architecture and the interactions with the different components of the SecureIoT solution. It provides a detailed specification of an Open API which can be exposed in order to utilize the RA&MS solution. Moreover, in this deliverable we reuse existing background technologies for the Risk Assessment Engine created from another European project and we redesign its exposed services in order to fit the SecureIoT solution and needs. In this context we provide an initial experimental PoC (Proof-of-Concept) implementation of this adaptation. Finally, the deliverable provides the Continue Integration, Continues Deployment and Continues Testing methodology that will be followed in WP5 and will be extended in the rest of the work packages that offer software components.

1.1 Available Risk Assessment Engines

NIST defines risk assessment as "the process of identifying the risk to system security and determining the likelihood occurrence, the resulting impact and the additional safeguards that mitigate this impact"[1]. This process creates information that is used in the decision making to prevent economical or reputational damage to individuals, organizations, or states.

Traditionally, static risk assessment was used to track risks and potential losses. It consists of a group of steps that identifies threat sources, events, vulnerabilities, the conditions needed to exploit them, the likelihood of successful attacks and its potential impact[2]. Although this risk assessment can be helpful for organizations, it does not take into account the possible relation between the risks, that could change significantly the result of the risk assessment.

[1] NIST. https://csrc.nist.gov/glossary/term/risk-assessment. Last visited on December 19th
[2] https://www.thesslstore.com/blog/cyber-risk-assessment/

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Dynamic risk assessment aims to solve this problem, by correlating multiple sources of information that discover possible interrelations between different risks[3]. This kind of risk assessment provides more accurate data about risks impact, giving the opportunity to implement better mitigation measures.

In order to give a better background of the different research and development activities in this field for risk assessment we describe some of the approaches that can be found in the market.

Symantec Risk Insight[4]

Risk Insight is a tool provided by Symantec, one of the biggest security companies in the world. This tool is thought for enterprises, providing executive dashboards that summarizes the risk faced by the whole enterprise, and analyzing all the systems present in the organization. Risk Insight uses the threat intelligence that Symantec collects from its global network, enhancing the risk analysis with latest cybersecurity trends all over the world.

Stream Cyber Risk Platform[5]

The Stream Cyber Risk Platform is an awarded tool built by Acuity, a London based company specialized in risk management. This platform aims to integrate several control frameworks and regulations, including ISO 27000, GDPR, PCI-DSS and NIST Cyber Security Framework, among others.

The Stream Cyber Risk Platform allows to import reports of vulnerability scanners, correlating the results to provide a more informed risk report. It also comes with preconfigured dashboards and reports, that can be configured to show results by areas of the enterprise.

Trick Service[6]

Trick Service is a tool developed by itrust consulting with the goal of assist companies in the complete risk management lifecycle. The tool calculates both qualitative and quantitative risk analysis. It also provides an API that can be used to receive information about Security Incident and Event Management (SIEMs), Intrusion Detection Systems (IDSs), firewalls, etc. and apply that information to the analysis, showing the users in real time potential losses of each risk.

Along with the risk management, Trick Service can be used to have an asset management, create tickets in some of the most common used ticketing systems like Jira or Redmine from the risks, and collect the information about the risks in reports.

[3] https://www.compact.nl/en/articles/dynamic-risk-assessment
[4] https://www.symantec.com/products/risk-insight
[5] https://acuityrm.com/
[6] https://www.trickservice.com/

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

1.2 Document structure

The structure of the deliverable is as follows:

- **Section 2** lists the RA&MS requirements documented in the different SecureIoT tasks. Moreover, it provides the dependencies that are used as inputs and outputs from the SecureIoT tasks.
- **Section 3** presents the RA&MS framework architecture by using the 4+1 view model methodology. Additionally, it provides a detailed RAE API specification
- **Section 4** provides a detailed description of the RA&MS background technologies that are going to be reused for the PoC implementation and more specifically the ATOS's RAE solution.
- **Section 5** provides the source code availability and requirements of an experimental PoC implementation for the RA&MS.
- **Section 6** describes the CI, CT, CD strategy that will be followed more specifically in WP5 and in other Work packages.
- **Section 7** is the final and concluding section of the deliverable. It draws the main conclusions from the SecureIoT architecture specification process and provides an outlook for the subsequent version/release of this deliverable (D5.2).

2 Analysis of requirements

This section describes the basis knowledge we have used for the design of the risk assessment component. It is composed of the different elements, each of them with a specific goal for the design. The requirements have been elicited from different sources: scenarios, stakeholders, and the SecureIoT architecture. We compiled them and specified the ones that we plan to fulfil with the risk assessment component. This way we could have a better understanding of how the risk assessment component fits in the general architecture of SecureIoT and the functionalities it offers for fulfilling the needs of the use cases of the project (and beyond).

Therefore, the requirements used as basis are the ones coming from the reference scenarios and use cases. Additionally, we also investigated the ones defined by the stakeholders, the technical aspects of the architecture and the scenarios of application. Of course, this initial description is a preliminary and we will update it as we foresee to enhance the component as the work progresses and it is used in the different scenarios of the project. Finally, we describe the different dependencies and interactions of the RA&MS with the SecureIoT project tasks.

2.1 SecureIoT reference scenarios and use cases

The use cases of SecureIoT cover very different scenarios of applications for IoT, which implies different needs and requirements. These pilots are Factory 4.0, connected vehicles and socially assistive robots.

Factory 4.0 covers smart manufacturing, which aims to allow for better and improved functionalities in industry, being able to use complex IoT devices for performing partial tasks and obtain very valuable information and feedback of the supply chain activities. This use case has different scenarios such as head-mounted device, control cabinet, manufacturing facility, production in Eco2Eco systems, order-controlled production and smart process automation.

The cyberthreats more important for each scenario are:

- Head-mounted device: the attack surface is considered small as no data exchange is done over public networks. Therefore, the more critical threat is the communication (wireless) between the IoT devices and the manufacturing execution system. This way, in this scenario the interest would be to check the communication component used for the wireless communication.
- Control cabinet: this component stores data of the supply chain so it is interesting to be able to identify known vulnerabilities, impact they could have in the supply chain and possible recommendations.
- Manufacturing facility: in this scenario there are several elements that could be target of known vulnerabilities or attacks such as the software of the sensors, the communication devices and the infrastructure for storing and sharing data such as fog and cloud infrastructure.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

- Eco2Eco system: this is a subset of the previous use case and all needs and vulnerabilities apply here too.
- Order-controlled production: this is a superset of the "manufacturing facility" use case so, apart from its identified needs, there exist other such as the existence of public web application and digital services that could be the target of DoS or DDoS attacks.
- Smart process automation: in this scenario there exist two main critical risks: the information theft and service disruption. This scenario assumes access to the key process components, but it is still a major issue due to the criticality of the data stored. Therefore, it is important to have assurance of the system against known vulnerabilities and threats

The connected vehicles scenario focuses on vehicles sharing information between them (V2V), vehicles accessing the internet (V2Internet), or to an infrastructure (V2I). In this sense, vehicles are treated as mobile networks of smart objects that have a very dynamic connectivity and share different types of data to trusted or untrusted nodes. For this use case it is regarded as very useful the identification of vulnerabilities in different elements of the car such as the motor, the communication components for transmission of vehicle data, battery, applications running in the system, etc. The communications are done via wireless and on-site (e.g. USB) so protection against data theft, misuse or hacking is critical.

Socially assistive robots aim to provide specific functionalities. They can do it in an isolated state or interconnected with other IoT devices, which implies a high number of connections and, therefore, threats. Also, there exists a high concern with security issues, being one of the main barres for their adoption in the market. Socially assistive robots, which are a specific sector of this area have a huge attack surface. They have a large distribution of assets, applications, operating systems, communication protocols, etc. For this reason, it is important to be able to identify and protect them against vulnerabilities and threats. The issues go from the data they store to the communications of data to other devices or over the internet.

2.2 SecureIoT architecture and technical specifications

As mentioned in D2.4 [1] and depicted in Figure 1 below, the Security Services (SECaaS) layer comprises the three SECaaS services that will be primarily provided by the SecureIoT platform, namely **Risk Assessment (RA) services**, Compliance Auditing services and Developer Support services. These services are specified in detail and implemented in the scope of WP5 of the project. In principle, this layer will also implement other services that will be based on the data processing outcomes of the Security Intelligence layer, such as alerting and visualization services (e.g., presentation of security-related information in dashboards). Note also that all these services can be implemented and offered on the basis of the SECaaS paradigm, which has been introduced earlier in this section.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

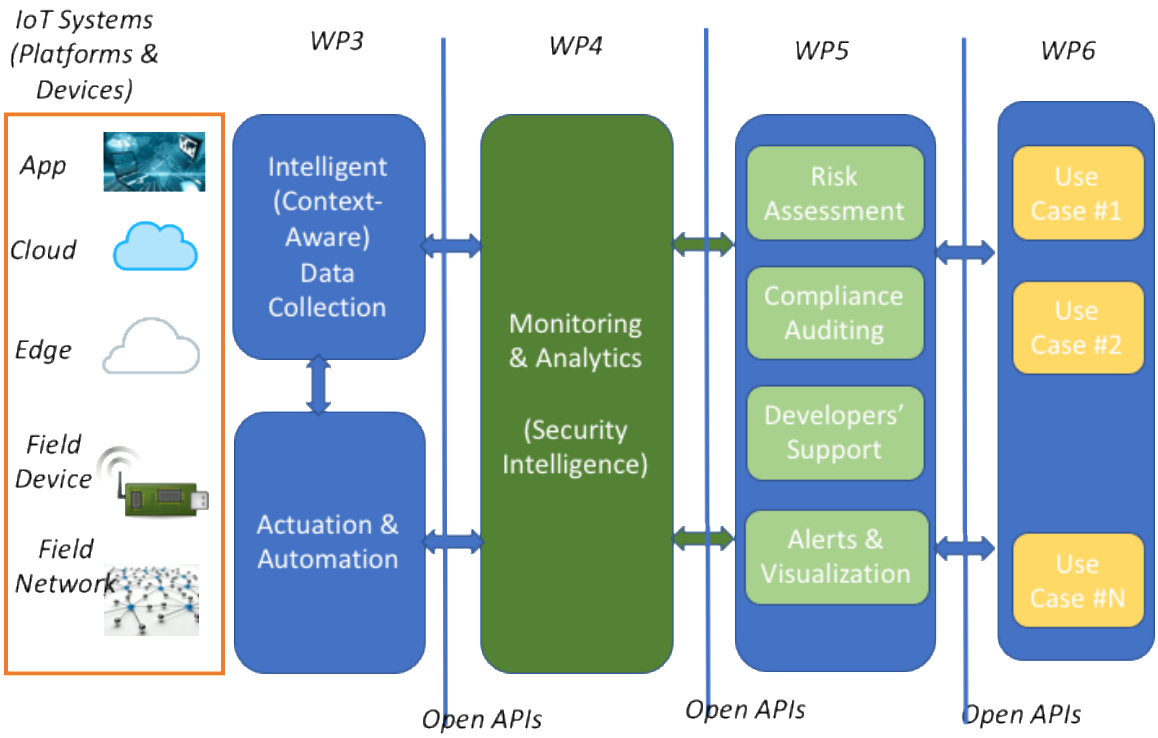


Figure 1: High Level Logical View of the SecureIoT Architecture and Mapping to Technical Workpackages [1]

The Risk Assessment services should provide an appropriate interface to the use case layer either through a well-defined Open API or a Risk assessment Dashboard. The Risk Assessment services should consume data streams and semantics produced from Data Collection and Analytics layers in order to evaluate identified threats and report them to the end user. Risk Assessment services should also provide means for mitigation actions by providing either advices/best practices (passive actions) or allow the user to trigger the SecureIoT PEP (active actions).

2.3 RA&MS Interactions with different tasks

The RA&MS, as also shown in Figure 2 in the next section, interacts with various tasks of the SecureIoT project. Below we list the different dependencies with these tasks and the type of the dependency.

Tasks producing expected inputs:

- T2.2 Stakeholders' Requirements for Security, Privacy and Trust: Requirements
- T2.4 Security Services Architecture and Technical Specifications: Architecture
- T4.1 Continuous Security Monitoring and Knowledge Inference: Data Input, Data Modelling
- T4.2 Predictive Analytics for IoT Security: Data Input
- T4.3 Security and Privacy Policies Interoperability: Data Modelling

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

- T5.3 Programming Support Services: Use output for enforcement of security policies (SW asset)
- T5.4 IoT Security Knowledge Base: Use the provided Knowledge Base (SW asset)
- T5.5 Continuous Integration and Configuration of SECaaS Services: Hosting of the SECaaS services (deployment platform), Integration of WP4/WP5 results outputs to Risk Assessment and Mitigation Services
- WP6 Usage Scenarios, Validation and Evaluation: Implementation scenarios

Tasks needing expected outputs:

- T6.2/T6.3/T6.4: use the RA&MS as SW asset
- T7.1/T7.4: use the RA&MS as SW asset

2.4 SecureIoT specification of usage scenarios

As we described in Section 2.1, we have identified in the three uses cases of the project necessities and requirements that are in line with the functionalities and characteristics of the risk assessment service. However, we are currently working in analysing in more detail with the use case partners how they plan to integrate and use the service for providing security and trust to their scenario.

So far, after discussing with the partners, we have decided to apply the service in the connected vehicles use case. In this scenario the benefits the risk assessment service brings is very valuable as there exist several assets and functionalities that are critical. In this way we have identified the elements that need to be protected and analysed for vulnerabilities:

- Onboard unit: captures information of different sensors and elements of the car, processes the information (aggregation), and sends it to the IoT platform for its storage. This component is very critical and therefore the protection of the risk assessment is very valuable. Besides, the other devices that communicate and exchange data with it are also highlighted as possible targets.
- IoT cloud platform: this platform receives information and stores it in the cloud. The information comes from different data sources so, in order to protect the functionality for other data providers (e.g. other cars), it is necessary to monitor this component for vulnerabilities and threats.

Regarding the assets planned to be used in the scenario we are researching how we can adapt the risk assessment service for being applied to them, bearing in mind their architecture, communication characteristics, technical requirements, etc. These assets are described in Table 1 below (more information about them can be found in D6.1).

Table 1: Assets planned to be used in the connected vehicles scenario

IoT asset	Component of the scenario
IDAPT platform	Vehicle OBU (On Board Unit)

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

FIWARE Orion context broker	IoT cloud platform
FIWARE IDAS	IoT cloud platform
FIWARE PEP proxy	IoT cloud platform
FIWARE KeyRock	IoT cloud platform
FIWARE Cygnus	IoT cloud platform

3 Specification of IoT Risk Assessment and Mitigation Service

In this section we describe the Risk Assessment & Mitigation Services (RA&MS) architecture by using the 4+1 view model designed by Philippe Kruchten [2]. First, we provide the architecture **Logical View** by using a high-level block diagram which identifies the interactions between the different SecureIoT components and the dependences between the SecureIoT tasks. We continue by presenting two different high-level **Scenarios** where the RA&MS could be used. Finally, we present the **Process View** with the help of UML sequence diagrams. At this stage the RA&MS design is not mature enough to present a detailed Development and Physical View which we will provide in the next version of the deliverable. In the second part of this section we provide a detailed API of the RA&MS component.

3.1 Architectural View Model

3.1.1 Logical view

Risk Assessment & Mitigation Services layer as shown in Figure 2 below sits on top of the SecureIoT architecture and interacts with all the other SecureIoT layers. The core component of the RA&MS is the Risk Assessment engine which is responsible to collect processed data from various sources of the SecureIoT underlying infrastructure to successfully assess potential risks and propose mitigation actions. As shown in Figure 2 below the main volume of data for risk assessment will be driven by information produced from the Data Monitoring & Analytics components from WP4. The data will be retrieved either by subscribing to the data streams or by accessing to the SecureIoT data storage where the Analytics results are persisted. Moreover, information will be retrieved from the Security Knowledge base where the NIST's Common Vulnerability Scoring System will drive the computation of a vulnerabilities' "likelihood" factor.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

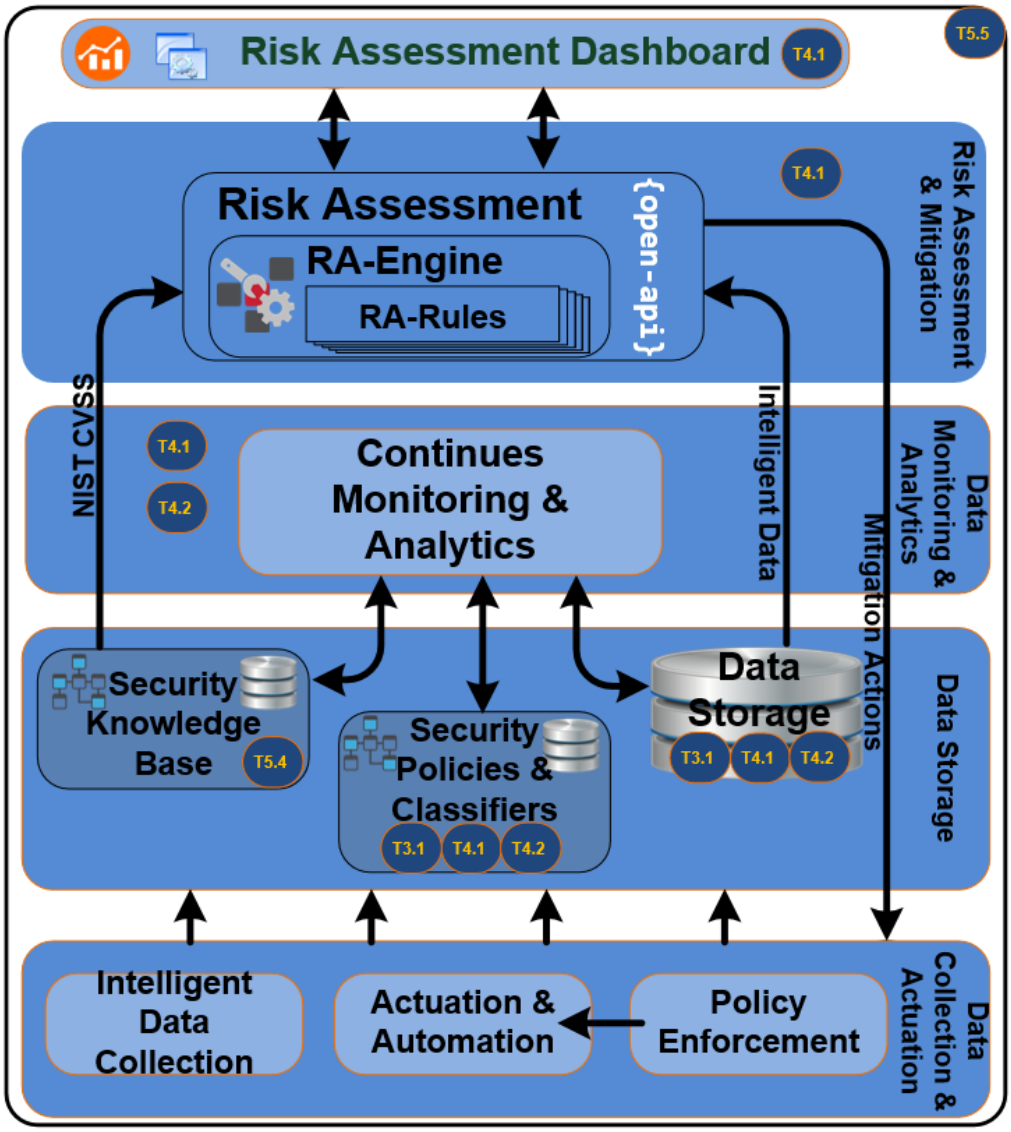


Figure 2: RA&MS Logical Architecture and Task dependencies

The results produced from the RA engine will be presented to the end-user accompanied with possible risk alleviation activities. These activities will include enforcement of proper security policies based on the capabilities specified by the monitored platform and the SecureIoT Policy Enforcement component. As shown in Figure 2 above as soon as a mitigation action is specified the RA engine should be capable of sending the command to the exposed interface from the Policy Enforcement component at the Data Collection & Actuation layer. Then the SecureIoT Policy Enforcement component will be responsible to execute the chosen action based on the exposed capabilities of the monitored platform.

3.1.2 Scenarios view

In this section we present two high level scenarios of using the RA&MS component. The first one is the configuration of the RA&MS component and the second one is the runtime of the RA&MS

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

component. Since we focus on the end users' perspective as primary actors we consider the Risk Assessment Dashboard as the point of interaction with the RA&MS. Therefore, we identify the Risk Assessment Dashboard as the system of the use case UML diagrams presented below.

3.1.2.1 Configuration

Figure 3 below illustrates a high-level RA&MS configuration use case. In this use case we can identify three actors. We have a primary actor which is the IT manager that is going to configure the RA&MS. Moreover, we identify two secondary actors which are the Knowledge base which is hosting the CVSS information and the Risk Assessment Engine which is our target configuration point.

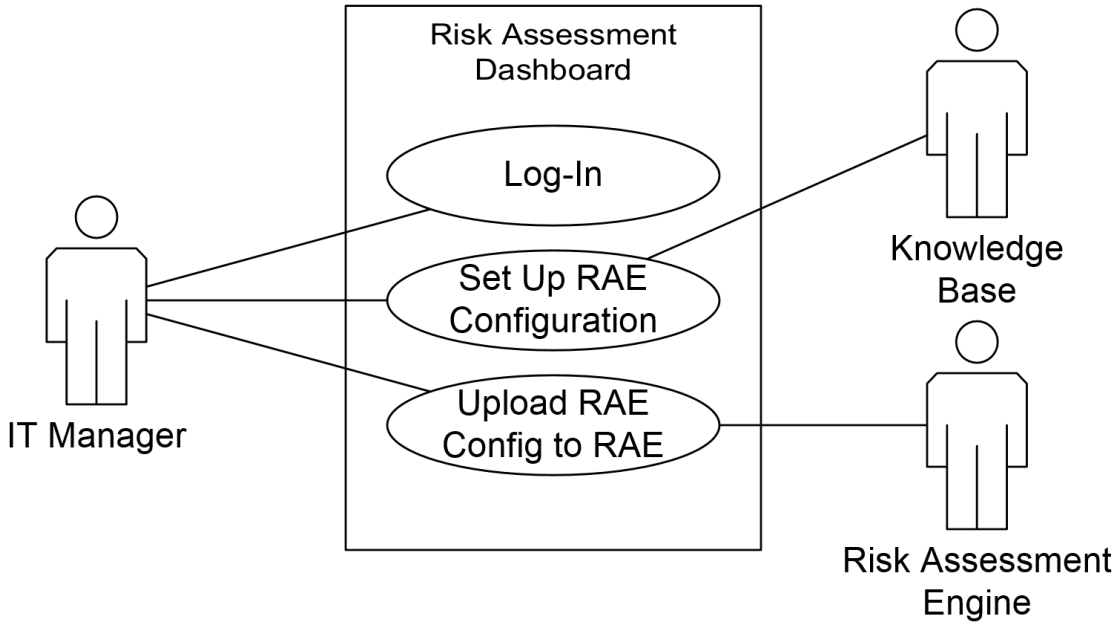


Figure 3: RA&MS configuration scenario View

In Figure 3 above we can see three different base use cases:

- **Log-in**: This base use case is required for the user to get identified and authorized in order to use the Risk Assessment Dashboard system. The interaction with the security secondary actor is out of the scope of this deliverable and thus not depicted.
- **Setup RAE Configuration**: the second step and base use case is the setup of the RAE configuration specification. In order to achieve that it requires a combination of interactions from the User to provide inputs for the RAE configuration and the Knowledge base in order to get feedback for the CVSS information.
- **Upload RAE Configuration to RAE**: The final step and base use case is the upload of the generated configuration file to the RAE in order to be persisted and to be used later at the runtime instantiation.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

3.1.2.2 Runtime

Figure 4 below illustrates a high-level RA&MS runtime use case. In this use case we can identify three actors. We have a primary actor which is again the IT manager that is going to instantiate the RA&MS. Additionally, we identify two secondary actors which are the Risk Assessment Engine which is going to be instantiated and the Policy Enforcement Point which we are going to trigger for an active mitigation action.

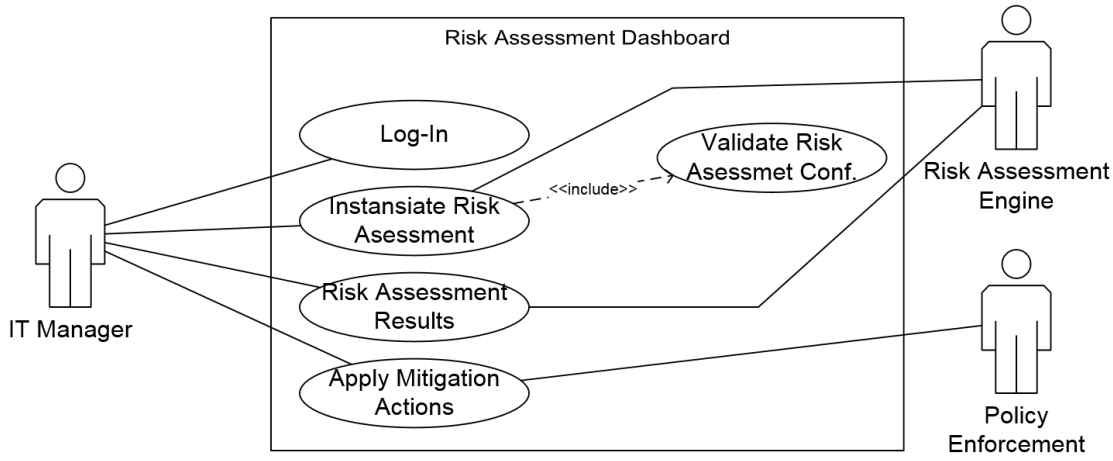


Figure 4: RA&MS runtime scenario View

In Figure 4 above we can see four different base use cases and an included use case:

- **Log-in**: This base use case is required for the user to get identified and authorized in order to use the Risk Assessment Dashboard system. As mentioned above the interaction with the security secondary actor is out of the scope of this deliverable and thus not depicted.
- **Instantiate Risk Assessment**: This base use case reflects the instantiation of the RAE for an already uploaded RAE specification from the previous use case. The RAE instantiation includes a validation use case where the RAE configuration document is validated for correctness before it is used for the RAE instantiation.
- **Risk Assessment Results**: After the RAE instantiation we have the result production base use case where the potential risks have been assessed and reported to the Risk Assessment Dashboard. Moreover, mitigation actions are proposed to the end user in passive (i.e., best practice/ advices) or active form (i.e. illustrate a button where a mitigation action can be triggered through the PEP)
- **Apply Mitigation Actions**: in this final base use case we have the application of an active mitigation action from the end user. By choosing the active mitigation action from the dashboard it is transferred to the PEP in order to be executed.

3.1.3 Process view

In this section we provide a more detailed description of the interactions between the different SecureIoT components in order to achieve the configuration and runtime use case described above.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

3.1.3.1 Risk Assessment & Mitigation Configuration

Figure 5 below illustrates the process for producing the RA&MS document in order to instantiate the RAE.

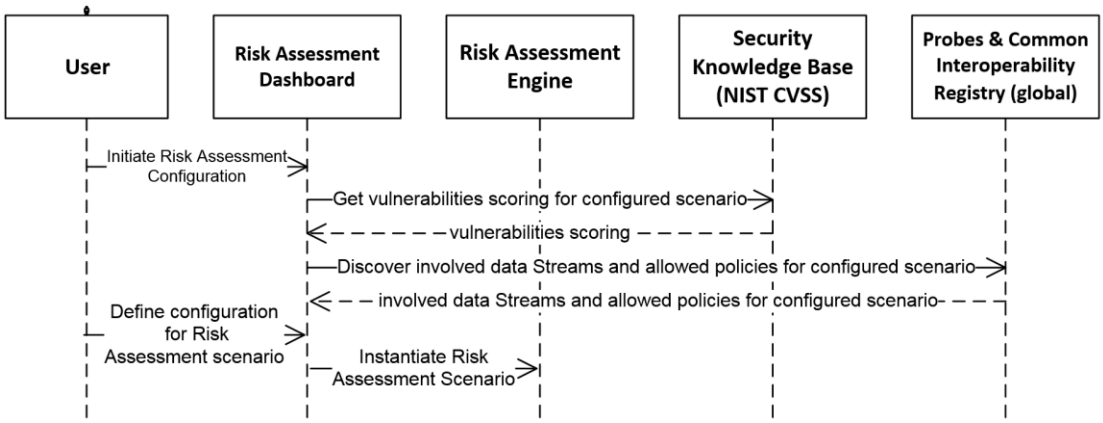


Figure 5: RA&MS configuration sequence diagram

The involved components are the Risk Assessment dashboard, the RAE, the Security Knowledge Base and the Probes & Common Interoperability Registry (P&CIR). The process is following the steps listed below:

1. First the user initiates a new Risk Assessment (RA) configuration procedure.
2. For the new Risk Assessment configuration to be initiated, the Risk Assessment Dashboard (RAD) contact the Security Knowledge Base (SKB) to retrieve the vulnerabilities scoring for the configured scenario.
3. Then the RAD communicates with the P&CIR in order to discover the involved data streams and allowed policies for the configured scenario.
4. Then the user having the retrieved information along with the scenario specific data which is required from him enter the RAD is ready to produce the RAE configuration file.
5. This file is then persisted to the RAE in order to be instantiated for the runtime.

3.1.3.2 Risk Assessment & Mitigation Runtime

Figure 6 below illustrates the process of the RA&MS instantiation and runtime.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

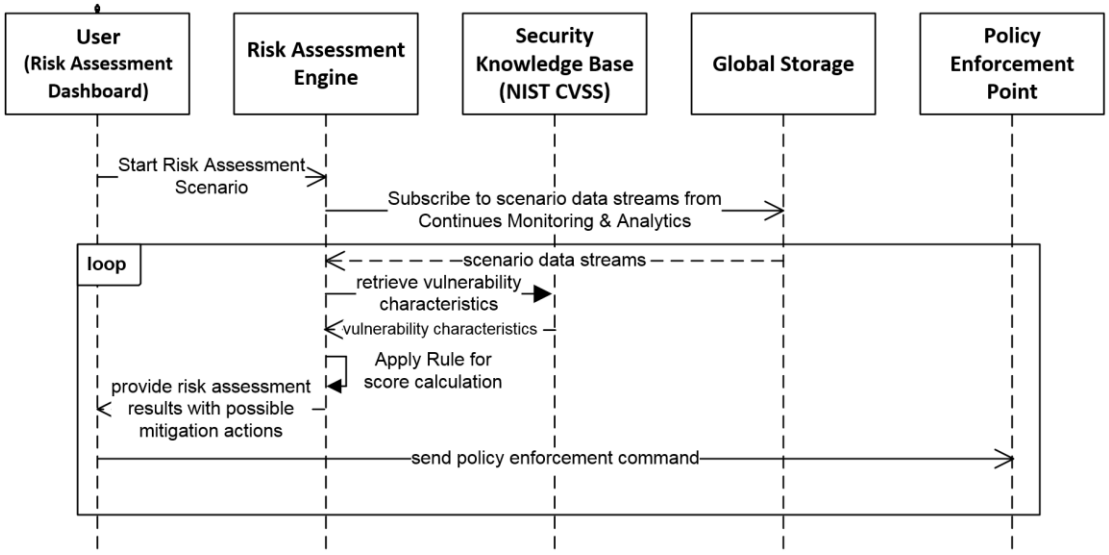


Figure 6: RA&MS runtime sequence diagram

The involved components are the Risk Assessment dashboard, the RAE, the Security Knowledge Base, the Global Data Storage and the Policy Enforcement Point (PEP). The process is following the steps listed below:

1. The user is instantiating the Risk Assessment scenario from the RA configuration that persisted from the previous process.
2. The RAE subscribes to the data streams of the scenario identified to the RA configuration that are generated from the Continues Monitoring & Analytics component.
3. The RAE goes to a continues loop by retrieving and evaluating the scenario specific data streams coming from the Continues Monitoring & Analytics component.
4. The data streams are evaluated against the vulnerability characteristics retrieved from the SKB and rules are applied to them based on the initial user inputs.
5. A report is provided as feedback to the user through the RAD presenting risk assessment results with possible mitigation actions.
6. The user may finally execute a proposed mitigation action which is forwarded to the PEP.

3.2 Risk Assessment and Mitigation Service API specification

This section provides the specifications of the Risk Assessment Engine API i.e. the API used to access functionalities of the RAE. The specification is provided in the form of RESTful interfaces. This API is a core part of the RA&MS infrastructure, since it enables external systems to take advantage of the RAE functionalities.

3.2.1 API and Interfaces Overview

Table 2 below illustrates the main API primitives that support the Risk Assessment Engine functionalities, while Table 3 to Table 14 below provide more details about each one of the functions that comprise the API.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Table 2: List of primitives comprising the API

```
<<interface>>
Risk AssessmentEngineInterface
---
POST: raeConf(raeConfig:RaeSpec):raeID:String
PUT: raeConf(raeID:String, raeConfig:RaeSpec):raeID:String
DELETE: raeConf(raeID:String) :raeID:String
GET: raeConf(raeID:String): RaeSpec
GET: allRaeConfigs():List<RaeSpec>
DELETE: allRaeConfigs()
POST: startRaeProcess(raeID:String):raeID:String
POST: stopRaeProcess(raeID:String)
POST: pauseRaeProcess(raeID:String)
POST: resumeRaeProcess(raeID:String)
POST: removeRaeProcess(raeID:String)
POST: removeAllRaeProcesses()
```

SecureIoT implements the methods of the Risk Assessment Engine API as specified in Table 3 to Table 14 below:

Table 3: Risk Assessment Engine API definition

Title	**Risk Assessment Engine** **Note**: Used to submit a new Risk Assessment Engine Configuration to the component. Takes as input the Risk Assessment Engine Spec document which includes the required configurations for the Risk Assessment Engine instantiation. Returns the Risk Assessment Engine Instance id.
URL	rae/rest/raeConf
Method	POST
Data Params (Body)	**Type**: XML
Success Response	**Code**: 201 **Content**: { id:RA_Scenario_INSTANCE_ID }
Error Response	**Code**:401 | **Content**: {error: "SecurityException"} **Code**:400 | **Content**: {error: "RaeSpecValidationException"} **Code**:500 | **Content**: {error: "ImplementationException"}

Table 4: Risk Assessment Engine Update API definition

Title	**Risk Assessment Engine Update** **Note**: Used to update an existent Risk Assessment Engine Configuration to the Risk Assessment Engine component. Takes as input the Risk

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

	Assessment Engine ID and the RaeSpec document which includes the required configurations for the Risk Assessment Engine instantiation.
URL	rae/rest/raeConf?raeID=:raeID
Method	PUT
URL Params	Required: raeID=[String]
Data Params (Body)	**Type:** XML
Success Response	**Code:** 200 **Content:** {success : "SuccessfullyUpdated"}
Error Response	**Code:**401 | **Content:** {error: "SecurityException"} **Code:**400 | **Content:** {error: "RaeSpecValidationException"} **Code:**404 | **Content:** {error: "NoSuchRaeID"} **Code:**500 | **Content:** {error: "ImplementationException"}

Table 5: Risk Assessment Engine API definition

Title	**Risk Assessment Engine Delete** **Note**: Used to delete an existent Risk Assessment Engine Configuration. This action stops and removes the related Risk Assessment Engine Processes.
URL	rae/rest/raeConf?raeID=:raeID
Method	DELETE
URL Params	Required: raeID=[String]
Success Response	**Code:** 200 **Content:** {success : "SuccessfullyDeleted"}
Error Response	**Code:**401 | **Content:** {error: "SecurityException"} **Code:**404 | **Content:** {error: "NoSuchRaeID"} **Code:**500 | **Content:** {error: "ImplementationException"}

Table 6: Retrieval of Risk Assessment Engine API definition

Title	**Retrieval of Risk Assessment Engine** **Note**: Used to retrieve a RaeSpec. Takes as input the Risk Assessment Engine ID. Returns an RaeSpec document.
URL	rae/rest/raeConf?raeID=:raeID
Method	GET
URL Params	Required: raeID=[String]

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Success Response	**Code:** 200 **Body Type:** XML
Error Response	**Code:**401 | **Content:** {error: "SecurityException"} **Code:**404 | **Content:** {error: "NoSuchRaeID"} **Code:**413 | **Content:** {error: "ResultTooLargeException"} **Code:**500 | **Content:** {error: "ImplementationException"}

Table 7: Retrieve All Risk Assessment Engines API definition

Title	**Retrieve All Risk Assessment Engines** **Note**: Used to retrieve all the available Risk Assessment Engine configurations. No input required. Returns a list of RaeSpec documents.
URL	rae/rest/allRaeConfigs
Method	GET
Success Response	**Code:** 200 **Body Type:** XML **Body Content Schema:** <xs:sequence> <xs:element ref="rae:RaeSpec" maxOccurs="unbounded"> </xs:sequence>
Error Response	**Code:**401 | **Content:** {error: "SecurityException"} **Code:**413 | **Content:** {error: "ResultTooLargeException"} **Code:**500 | **Content:** {error: "ImplementationException"}

Table 8: Delete All Risk Assessment Engines API definition

Title	**Delete All Risk Assessment Engines** **Note**: Used to delete all Risk Assessment Engine configurations. This action stops and removes the related Risk Assessment Engine Processes. Returns a status message.
URL	rae/rest/allRaeConfigs
Method	DELETE
Success Response	**Code:** 200 **Content:** {success : "AllSuccessfullyDeleted"}
Error Response	**Code:**401 | **Content:** {error: "SecurityException"} **Code:**500 | **Content:** {error: "ImplementationException"}

Table 9: Start a Risk Assessment Engine Process API definition

Title	**Start a Risk Assessment Engine Process**

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

	Note: Used to start a Risk Assessment Engine instance Process. Gets as input the Risk Assessment Engine ID. Returns the process ID.
URL	rae/rest/startRaeProcess?raeID=:raeID
Method	POST
URL Params	Required: raeID=[String]
Success Response	**Code**: 201 **Content**: { id : RA_Scenario_Rule_ID }
Error Response	**Code**:401 | **Content**: {error: "SecurityException"} **Code**:404 | **Content**: {error: "NoSuchRaeID"} **Code**:500 | **Content**: {error: "ImplementationException"}

Table 10: Stop a Risk Assessment Engine Process API definition

Title	**Stop a Risk Assessment Engine Process** **Note**: Used to stop a Risk Assessment Engine instance Process. Gets as input the Risk Assessment Engine ID.
URL	rae/rest/ stopRaeProcess?raeID=:raeID
Method	POST
URL Params	Required: raeID=[String]
Success Response	**Code**: 200 **Content**: {success: "SuccessfullyStoped"}
Error Response	**Code**:401 | **Content**: {error: "SecurityException"} **Code**:404 | **Content**: {error: "NoSuchRaeID"} **Code**:500 | **Content**: {error: "ImplementationException"}

Table 11: Pause a Risk Assessment Engine Process API definition

Title	**Pause a Risk Assessment Engine Process** **Note**: Used to pause a Risk Assessment Engine instance Process. Gets as input the Risk Assessment Engine ID.
URL	rae/rest/ pauseRaeProcess?raeID=:raeID
Method	POST
URL Params	Required: raeID=[String]
Success Response	**Code**: 200 **Content**: {success : "SuccessfullyPaused"}
Error Response	**Code**:401 | **Content**: {error: "SecurityException"} **Code**:404 | **Content**: {error: "NoSuchRaeID"} **Code**:500 | **Content**: {error: "ImplementationException"}

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Table 12: Resume a Risk Assessment Engine Process API definition

Title	**Resume a Risk Assessment Engine Process** **Note**: Used to resume a Risk Assessment Engine instance Process. Gets as input the Risk Assessment Engine ID.
URL	rae/rest/resumeRaeProcess?raeID=:raeID
Method	POST
URL Params	Required: raeID=[String]
Success Response	**Code**: 200 **Content**: {success: "SuccessfullyResumed"}
Error Response	**Code**:401 | **Content**: {error: "SecurityException"} **Code**:404 | **Content**: {error: "NoSuchRaeID"} **Code**:500 | **Content**: {error: "ImplementationException"}

Table 13: Remove a Risk Assessment Engine Process API definition

Title	**Remove a Risk Assessment Engine Process** **Note**: Used to remove a Risk Assessment Engine instance Process. Gets as input the Risk Assessment Engine ID.
URL	rae/rest/ removeRaeProcess?raeID=:raeID
Method	POST
URL Params	Required: raeID=[String]
Success Response	**Code**: 200 **Content**: {success : "SuccessfullyRemoved"}
Error Response	**Code**:401 | **Content**: {error: "SecurityException"} **Code**:404 | **Content**: {error: "NoSuchRaeID"} **Code**:500 | **Content**: {error: "ImplementationException"}

Table 14: Remove all Risk Assessment Engine Processes API definition

Title	**Remove All a Risk Assessment Engine Processes** **Note**: Used to remove all Risk Assessment Engine instance Processes. No input is required.
URL	rae/rest/ removraellRaeProcesses
Method	POST
Success Response	**Code**: 200 **Content**: {success : "AllSuccessfullyDeleted"}
Error Response	**Code**:401 | **Content**: {error: "SecurityException"} **Code**:500 | **Content**: {error: "ImplementationException"}

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

3.2.2 Exceptions

Methods of the RAE API signal error conditions to the client by means of exceptions. The following exceptions are defined in Table 15 below.

Table 15: Exceptions associated with the RAE API

Response String	Description	Error Code
`SecurityException`	The operation was not permitted due to an access control violation or other security concern. This includes the case where the service wishes to deny authorization to execute an operation based on the authenticated client identity. The specific circumstances that may cause this exception are implementation-specific, and outside the scope of this specification.	401
`RaeSpecValidationException`	The Risk Assessment Engine failed to successfully validate the RaeSpec comprising the service request. This can be a result of a malformed RaeSpec specification.	400
`NoSuchRaeID`	The Risk Assessment Engine failed to identify the Risk Assessment Engine ID i.e. the Risk Assessment Engine ID is not available within the RAE repository.	404
`ImplementationException`	A generic exception raised by the implementation for reasons that are implementation-specific. This exception contains one additional element: a severity member whose values are either ERROR or SEVERE. ERROR indicates that the Risk Assessment Engine implementation is left in the same state it had before the operation was attempted. SEVERE indicates that the Risk Assessment Engine implementation is left in an indeterminate state.	500
`ResultTooLargeException`	An attempt to execute a request resulted in more data than the service was willing to provide.	413

All status codes are standard HTTP status codes. The below ones are used in the RAE API:

- 2XX - Success of some kind
- 4XX - Error occurred in client's part
- 5XX - Error occurred in server's part

Table 16 below lists the available Status Code Description can be found.

Table 16: Error Codes Description

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Status Code	Description
200	OK
201	Created
400	Bad request
401	Authentication failure
404	Resource not found
413	Request Entity Too Large
500	Internal Server Error

The exceptions that may be raised by each Risk Assessment Engine method are indicated in Table 17 below. A RAE-Processor implementation SHALL raise the appropriate exception listed below when the corresponding condition described above occurs. If more than one exception conditions apply to a given method call, the Risk Assessment Engine implementation may raise any of the exceptions that applies.

Table 17: Exceptions thrown by the different Risk Assessment Engine services

Primitive	Throws
POST:raeConf(raeConfig:RaeSpec):String	SecurityException RaeSpecValidationException NoSuchRaeID ImplementationException
PUT:raeConf (raeID:String, raeConfig:RaeSpec)	SecurityException RaeSpecValidationException NoSuchRaeID ImplementationException
DELETE:raeConf (raeID:String)	SecurityException NoSuchRaeID ImplementationException
GET:raeConf (raeID:String):RaeSpec	SecurityException NoSuchRaeID ImplementationException ResultTooLargeException
GET:allRaeConfigs():List<RaeSpec>	SecurityException NoSuchRaeID ImplementationException ResultTooLargeException
DELETE:allRaeConfigs()	SecurityException ImplementationException
POST:startRaeProcess(raeID:String): String	SecurityException NoSuchRaeID ImplementationException
POST:stopRaeProcess(raeID:String)	SecurityException NoSuchRaeID ImplementationException
POST:pauseRaeProcess(raeID:String)	SecurityException

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

	NoSuchRaeID ImplementationException
POST:resumeRaeProcess(raeID:String)	SecurityException NoSuchRaeID ImplementationException
POST:removeRaeProcess(raeID:String)	SecurityException NoSuchRaeID ImplementationException
POST:removraellRaeProcesses()	SecurityException ImplementationException

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

4 Background technologies

The cybersecurity risk assessment component aims to provide and quantify information of cybersecurity risks of the target IoT system. The component will evaluate the risks using a common vulnerability scoring system and provide the information both visually and as reporting. This way, and following the architecture shown in the previous section, we studied some options for the design and development of this component, which should satisfy the requirements obtained from the use cases, stakeholders and SecureIoT objectives.

The solution we propose to use as basis for the risk assessment component is a solution called Risk Assessment Engine (RAE), provided by Atos. The use of an existing RAE accelerated the initial implementation and permitted the SECaaS service developers to focus on IoT specific functionalities of the risk assessment service. Following, we describe the architecture, functionalities it provides and an initial plan to extend it for supporting the needs of the IoT domain based in the SecureIoT approach.

4.1 ATOS Risk Assessment Engine

The Risk Assessment Engine is a solution for providing information of the cybersecurity risks of a target system. The information to be evaluated is separated in two types: (i) financial and economic information of the impact in a component or service of the organization, and (ii) the cybersecurity status of the organization. The information is analysed using different models and provides information about the cyber risks of a company. This is provided both for the technical and management areas of the company. The technical experts will use the reporting for knowing the status of the system in terms of risks and vulnerabilities. The management level will be able to take decisions about the cybersecurity strategies or solutions to be used in their organization as the report provided offers information about the economic impact of the threats. This way managers are able to understand the cyber risk exposure of their organization, helping them in taking decisions in long term rather than short term.

The RAE does this evaluation in near real-time, executing a set of machine-readable risk algorithms that can be modified or extended according to the needs and specifics of the area of application, size of the company, type of input, etc. The RAE not only evaluates the cyber-risks but also is able to propose mitigation measures, which are defined by experts in the area.

4.1.1 Architecture description

The RAE is composed of four different components with specific functionalities. These components communicate internally for receiving, processing and showing the information of the system. Figure 7 below shows a high-level diagram of the architecture, showing its components, inputs and output.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

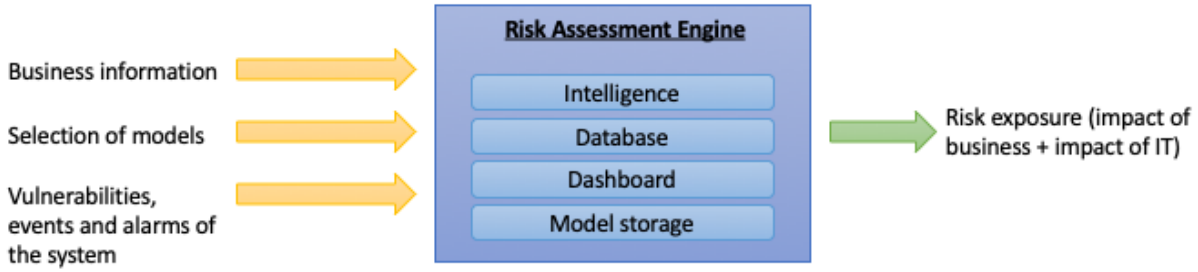


Figure 7: High-level diagram of the RAE

As it can be seen in Figure 7 above, the Risk Assessment Engine has four main components:

- Intelligence: this module is in charge of data analysis, correlation, classification, etc.
- Database: it is in charge of storing the information of the targeted system. Having a database allows for performing historical analysis that are very valuable for demonstrating the cyber-risk evolution of the system and impact of the solutions in the business
- Dashboard: this component is used for showing several interfaces of the RAE. For example, it shows questionnaires for compiling management data, the results of the analysis, recommendations, etc.
- Model storage: it stores the models used for the analysis of the cyber risks of the organization. As we mentioned before, there exist several models that can be applied together for the evaluation of the system. Additionally, more can be created and used according to the specific needs of the organization or field of application.

Regarding the input to the system we have on the one hand the input of the system to be analysed (business and technical information) and, on the other hand, the models to be applied in the assessment. More specifically:

- Business information: describes information such as costs, impact, financial aspects, etc. It is used for the management-level report of the risk analysis
- Vulnerabilities, events: is a report with the findings of the technical issues of the target system. It can contain cybersecurity events, vulnerabilities, etc. For example, it can use the event report generated by a SIEM or vulnerability manager.
- Models: the last input is the selection of the models that are going to be applied in the assessment of the system. As we mentioned more than one model can be used for the analysis according to the type of system analysis to be done, etc.

Finally, the output of the RAE is a report containing both technical and management-level information of the target system. This cyber-risk report also contains recommendations for certain threats in the system in order to provide some guidance about best strategies for solving issues.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Regarding the indicators, which are used for compiling information of the system (the input for the assessment), we have four different types, each aiming to one specific objective. Figure 8 below shows a diagram with each type.

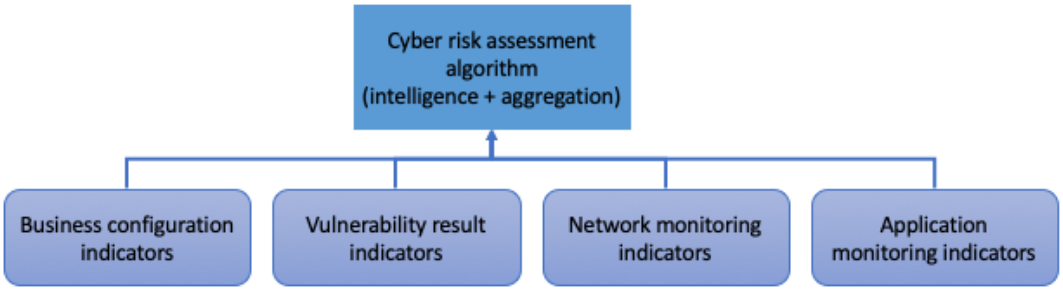


Figure 8: Indicators of the RAE

The cyber-risk assessment algorithm is the one in charge of doing the aggregation of all data of the system and the analysis. In order to do this uses the indicators (information) of the system. Each indicator can be compiled in different ways and has a specific goal. More specifically:

- Business configuration indicators: this is the management-level information compiled from the employees. It compiles information regarding critical assets, business, type of services offered, criticality, costs of losing an asset for an unknown period, etc. Usually the information is compiled using questionnaires or other type of user-level interaction. This is later transformed to machine-oriented for the assessment of the system.
- Vulnerability result indicators: this input is the vulnerability analysis of the target system. It is a report generated by a tool such as a vulnerability analysis or SIEM. This information is usually provided machine-ready as it is the output of risk and vulnerability assessment.
- Network monitoring indicators: this indicator, also machine-oriented, is the identification and report of the networking of the system and status. The goal with this indicator is to be aware of the networking of the system and its cyber-status.
- Application monitoring: it provides information about the cybersecurity monitoring of the applications of the target system. This one, together with the network monitoring and vulnerability results, aim to provide the cyber-status of the system covering different layers of the system such as the vulnerabilities of the whole system, network and application level.

Finally, the lifecycle of the models used for the assessment is shown in Figure 9 below. The models are of three different types: CORAS[7], DEXi[8] and R[9]. In order to focus in this document only in the RAE more information about them can be found in the literature. The models are created by experts for the assessment of the system. They are stored in the RAE so users can select them

[7] http://coras.sourceforge.net/
[8] https://kt.ijs.si/MarkoBohanec/dexi.html
[9] https://www.r-project.org/

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

in the tool when using it. The models are then applied by the intelligence component to the input data and, if necessary, can be analysed by the users for understanding their assessment weight and strategy so they can select the ones that better fits their needs.

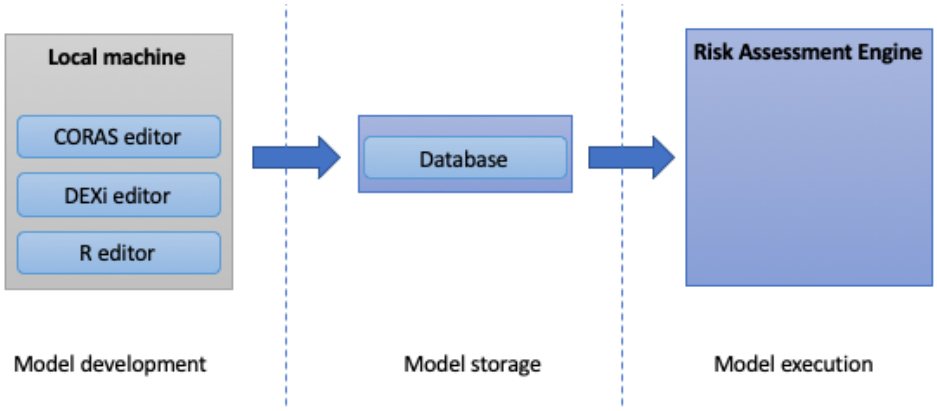


Figure 9: Lifecycle of models

4.1.2 Features offered

The RAE, as explained above, provides a functionality for assessing the cyber-status of a system or organization providing technical and management-level feedback. These features fit with the objectives of the SECaaS Risk Assessment Component, which is described in Section 2. The RAE uses as input models for the analysis of the status, being able to work with several ones at the same time as each of them focus in different characteristics. This model functionality can be integrated with the SecureIoT security knowledge database, so the analysis is oriented to IoT and its special constraints. Additionally, the output is generated in an interface that can be exported to other formats (e.g. PDF, pattern, etc.) in order to facilitate the information from any device.

The input is supported either via manually with questionnaires or automatically with APIs for providing the data. The format in which the data are processed (of vulnerabilities or events) is already defined and can be adapted to any type of tool used for this. This way the RAE can be integrated with any solution of vulnerability assessment or event management.

4.1.3 Data Models

Regarding the format of the inputs we present examples of two types of indicators: the business one and an event used for the vulnerability indicators.

The business one, as we commented, uses a manual and user-oriented functionality with questionnaires. These questionnaires can be modified, enhanced or created new as necessary according to the field of application, type of organization, etc. Figure 10 below shows an excerpt of a questionnaire about estimated economical loss due to cybersecurity threats for availability, confidentiality, etc. attacks. Bear in mind these estimations are done by experts of the organization that can estimate the impact in their business.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

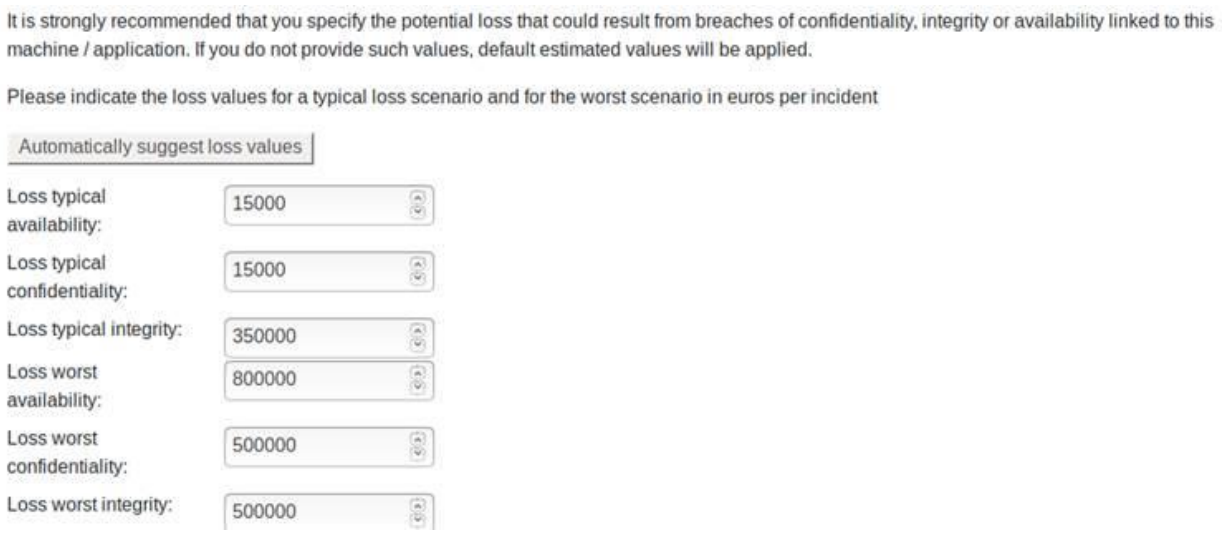


Figure 10: Excerpt of questionnaire about financial impact

Regarding the input of the vulnerabilities and events we show an excerpt in Table 18 below about the format of an event of the system used as input for the assessment. As it is shown, the event is linked with a model (represented in the "question" field) in order to evaluate the impact it has and what is affected.

Table 18: Excerpt of SIEM event for risk assessment

```
{
"id": 7,
"question": "IN-11:Is there a malware-Trojan inside the infrastructure?",
"data_type": "boolean",
"motivation": "It has been detected malware signatures in the client's
                        infrastructure network traffic.",
"indicator_type": 3,
"means": "alarm",
"default_value": "No",
"rule": "PLUGIN_ID=70000 AND PLUGIN_SID IN (40000:44999)",
"mode_of_operation": 1,
"sensor_types": []
},
```

Once the input is provided the RAE can use the specified models for calculating the assessment. The model (or models) used are executed against the R and DEXi scripts, which are generated using as basis the cybersecurity model of reference. Figure 11 below shows an example of a model.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

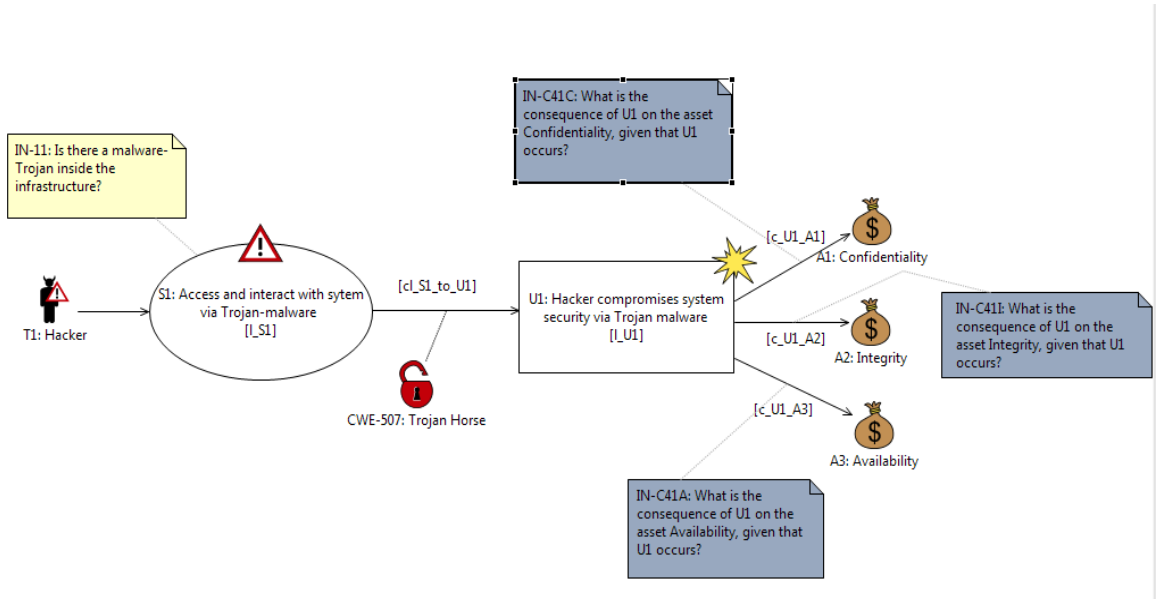


Figure 11: Cybersecurity model used for assessment

4.1.4 Reuse of Components

In the initial assessment of the solution we think it could fit with the needs of the SECaaS service as described in Section 3. The four components of the RAE, shown in Figure 7, will be extended in order to support the requirements and necessities identified in Section 2. Currently we are analysing the extensions and enhancements to be done but some of the initial lines of research and development we have are:

- Adapt the intelligence models and analysis to the SecureIoT security knowledge database and the type of data used for the assessment. This repository of cyber-information follows a specific format and is used by other components of SecureIoT, which makes it a very useful component as it will be in-line with the knowledge of SecureIoT and its more important characteristics.
- Adapt the input of data to the IoT market, business and special constraints. In order to perform a correct and complete evaluation of the target system it is required that the input, both the business and the technical one, is adapted to the IoT domain. It is necessary to enhance and modify all the indicators we described previously to the requirements described in Section 2.
- Adapt the results of the risk assessment to include information and functionality with the assurance component of SecureIoT. As it is described in the description of work the risk analysis could recommend actions done by the assurance component in order to solve vulnerabilities or risks detected.
- Enhance the risk analysis to cover specific characteristics of IoT such as using assets of the IoT system as potential attack platforms, considering IoT devices can couple and decouple in a given system, change the scope for the periodic assessment (IoT systems can change in a very short period of time), etc.

- Integration in the trust and authentication system of SecureIoT. The RAE is integrated with KeyCloak so it can be easily adapted to the general authentication infrastructure of SecureIoT.

We are currently working in extending this list adapting the RAE to the requirements of SecureIoT, so the previous key elements of research and development will be updated periodically as we continue the development of the SecureIoT system and its integration in the use cases.

5 Experimental PoC implementation

In this section we present the components for an initial experimental PoC implementation. This implementation consists of two software blocks. The one is the RAE which is provided as background technology from ATOS and the second one is the "adapter/wrapper" of ATOS's RAE which is exposing the functionalities described in Section 3.2 (RAE API specification) above and will be used from the other SecureIoT components. Please note that in this initial stage the PoC implementation it is not integrated with the components of the SecureIoT architecture, for which outputs are not specified yet (e.g., the Continues Monitoring & Analytics).

5.1 ATOS RAE

Following we describe technical requirements for the Risk Assessment Engine in order to run it in a system. We plan to provide a deployed version for SecureIoT so we can have a version for the testing and assessment of the project, so all partners can access it and provide feedback in its enhancement and integration with the SecureIoT architecture.

5.1.1 Binaries availability

Currently we are working in providing the RAE as binaries and deployed in the development platform, so all partners can test it and provide feedback for its improvement. The binaries can be found in the GitLab repository of the project.

5.1.2 Requirements

Minimum hardware requirements for running the application:

- 4GB RAM
- 1 CPU
- 5GB free disk space

Recommended hardware requirements:

- Software Requirements:
- Python 2 virtual environment
- Python 3 virtual environment
- PostgreSQL database
- Notice that you probably also will need a RabbitMQ queue manager installed in a server that has connectivity with the Risk Assessment Engine server. It can be installed in the same server.

5.2 ATOS RAE adaptation

This section provides information on the code availability and installation of the SecureIoT RAE prototype implementation based on ATOS's RAE. ATOS's RAE is controlled through a command shell and runs as an operating system service. This component offers a wrapper/adapter over ATOS's RAE which exposes the SecureIoT RAE API as REST web services. In this version of the

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

deliverable a basic functionality of the RAE adapted services has been implemented that offers an experimental functionality. The implemented services are:

- Submit a new Risk Assessment Engine Configuration to the Risk Assessment component.
- Retrieve a Risk Assessment Engine Configuration from the Risk Assessment component.
- Start a Risk Assessment Engine instance Process.
- Stop a Risk Assessment Engine instance Process.

5.2.1 Code and Binaries availability

The Risk Assessment Engine component is offered at SecureIoT GitLab repository. The repository is named RiskAssessmentEngine and is available at: https://gitlab.atosresearch.eu/secureiot/riskassessmentengine. The latest version of the components is under the "develop" branch.

The repository is organized in 3 categories/folders:

- core: provides the core modules of the Risk Assessment Engine
- utils: provides utilities and documents related with Risk Assessment Engine
    - eu.secureiot.rae.commons: includes common utils/objects used in more than 2 projects/components

5.2.2 System Requirements

The RAE adaptation component is deployed within a WildFly container and uses Maven for project management. The prototype runs on Windows, Linux, Mac OS X, and Solaris. In order to run the prototype, you need to ensure that Java 8 and WildFly are installed and/or are available on your system. In order to build the prototype, you will also need Maven. Before attempting to deploy and run the prototype applications, make sure that you have started WildFly.

5.2.3 Installing & Running the RAE

The prototype has been implemented as a Maven-based web application. Below **WILDFLY_HOME** indicates the root directory of the WildFly distribution, and **PROJECT_HOME** indicates the root directory of the project.

In order to **configure** the prototype,

1. make sure that all properties listed in **$PROJECT_HOME/src/main/resources/secureiot.properties** have the appropriate values,
2. copy that file into **$WILDFLY_HOME/standalone/configuration**, and
3. issue the following commands:

In order to **build** the prototype, run the following command in **PROJECT_HOME**:

```
mvn clean package
```

Finally, in order to **deploy** the prototype, run the following command in **PROJECT_HOME**:

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

```
mvn wildfly:deploy
```

The last step assumes that WildFly is already running on the machine where you run the command.

Alternatively copy the produced (from the build process above) **rae.war** file from the **target** directory (**$PROJECT_HOME/target/**), into the **standalone/deployments** directory of the WildFly[10] distribution, in order to be automatically deployed.

If the deployment has been successfully completed, you will be able to access all web services described in the section above using the following URL:

http://[HOST]:[PORT]/rae

Where [HOST] is the host and [PORT] the port that WildFly uses.

[10] http://wildfly.org/

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

6 Continuous integration (CI), continuous testing (CT) and continuous deployment (CD) strategy

The SecureIoT security services following the Security as a Service (SECaaS) paradigm must provide high-added value functionalities to help all the stakeholders of the IoT value chain to design, implement and maintain secure and privacy-aware IoT systems and to respond properly to dynamic and changing vulnerabilities, threads and attacks. Following SecureIoT approach, the SECaaS will exploit the relevant batch and streaming security information collected by WP3 intelligent data probes and the abnormal behaviour detectors and predictions of WP4 data analytics. Nevertheless, to provide really valuable functionalities that address and satisfy the technical and business requirements from the involved actors, WP5 will follow an agile and iterative approach focusing on the needs of the three industrial domains that are part of WP6. Thus, at the end of the project, SecureIoT SECaaS will include a portfolio of services with relevant features which will be validated and demonstrated in real IoT scenarios, warrantying their exploitation potential and a high Technology Readiness Level (TRL).

SecureIoT WP5 includes a dedicated task focused on the adoption and enforcement of agile and DevOps methodologies: *T5.5 Continuous Integration and Configuration of SECaaS Services.* The main goals of this philosophy are:

- To warranty that all SECaaS are released using high-quality and industrial readiness prototypes at the end of the project.
- To minimize the impact and difficulties with respect to integration with the already existing ICT infrastructures and services of WP6 use-cases.
- To promote and enforce a common and consolidated methodology and approach, aiming to early detect and mitigate risks, e.g., delays, lack of considering of requirements, etc.

Thus, T5.5 is a horizontal activity that will support and guide the design, implementation, validation and release of SECaaS, being thus a key element contributing to all WP5 tasks, deliverables and especially to the development of the prototypes. Nevertheless, as there is not a specific deliverable or placeholder to document and describe the overall approach, design patterns and tools that result from the effort of T5.5, this content is included in the present deliverable D5.1. Moreover, subsequent updates and iterations will be also integrated with the next versions D5.2 and D5.3, to be released in M21 and M30 respectively. They will also contain detailed and specific information with respect to the application of T5.5 results in the risk assessment and mitigation service. Similar content will be part also of D5.4 – D5.12 for the rest of SECaaS' deliverables.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

6.1 SECaaS common development methodology (T5.5)

6.1.1 Overall approach and methodology selection

Traditionally, software systems (and even hardware) are developed using a waterfall approach as shown in Figure 12:

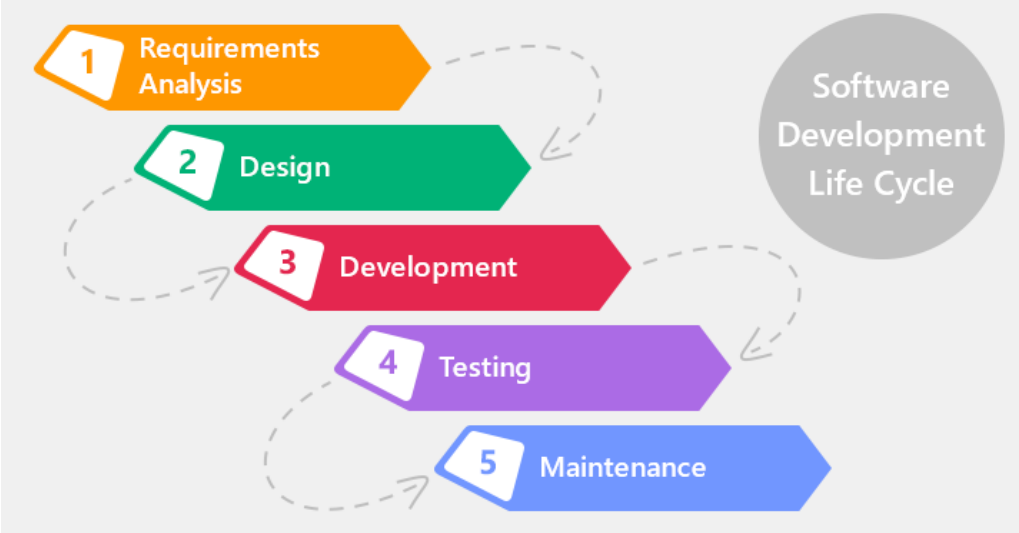


Figure 12: Software development lifecycle following a waterfall approach [3]

From a high-level perspective, following this model, the process starts to capture the requirements for systems or applications gathering information from customers, users, regulators, stakeholders, etc. Then they are analysed to produce the architecture, models and technical specifications that will guide the development process. In the third step, the system is implemented, and third-party libraries or technologies are integrated. Testing and verification address the discovery / debugging of bugs and finally the validation of the resulting functionalities against the initial requirements. Finally, the system is installed or deployed in production, giving appropriate support and maintenance to our clients and users. The waterfall model is still widely used in many companies, but the fact of executing the different phases sequentially may lead in some cases to ineffective approaches that do not consider properly the initial requirements and feedback from customers since they are only targeted in the first and last stages of the design process.

Following a waterfall model, these phases are executed sequentially, which may lead in some cases to ineffective approaches because for instance the feedback from customers and partners is only assessed at the end.

As an alternative during the last years, agile methodologies have appeared to promote iterative, incremental, continuous and holistic development procedures. Some of the most well-known and widely adopted alternatives are Scrum, Lean Kanban, Extreme programming or Test-Driven Development. A schematic representation of the first to options is presented in Figure 13:

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

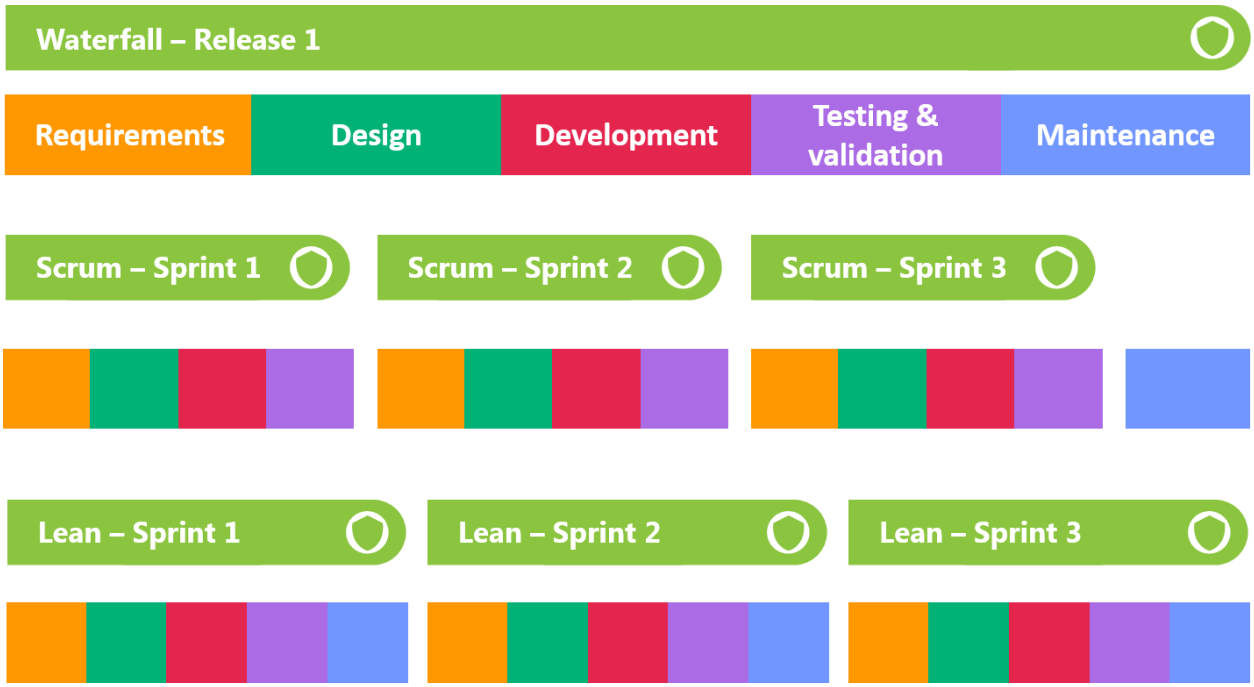


Figure 13: Comparison between software life-cycle development methodologies: waterfall, Scrum and Lean Kanban

Agile methodologies like Scrum or Lean Kanban promote completing smaller development *sprints*, prioritizing the assessment and evaluation that final users, customers and stakeholders can perform iteratively not only of the produced prototypes but also of the technical specifications and models derived from the initial design requirements. They are also very appropriate for the development of complex systems that are composed of multiple components and technologies which must be integrated, which is the case of SecureIoT and specifically of the SECaaS.

As an alternative, DevOps or agile practices promote iterative, incremental, continuous, automated and holistic development procedures. You can see that following Scrum or Lean methodologies, we complete smaller development sprints where the user's feedback is prioritized and where the initial requirements or models are also updated. At the same time, this kind of methodology is really appropriate to develop systems which require the integration of multiple components.

Therefore, the software design-cycle of SecureIoT SECaaS will follow an agile methodology strongly inspired by Lean Kanban. This method has a strong emphasis on continuous processes and avoids overloading the development team. All the tasks identified to be done are part of the backlog of pending activities while the team is focused on the visualization of the ongoing work. Unlike Scrum, it limits the amount of work that is done in each sprint and has a higher level of flexibility to simplify its adoption.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

6.1.2 WP5 timeframe

Following the Lean Kanban methodology, Minimum Viable Prototypes (MVPs) of SECaaS services are released from an early phase of the project and continuous improvements and new features are added and exposed to WP6 use-cases' partners. Nevertheless, to be consistent with the rest of work-packages and to simplify the monitoring and assessment of the WP5, three major releases have been proposed, matching the WP5 milestones of M12, M21 and M30. They are shown in a visual way in Figure 14.

The first release has had an intermediate internal checkpoint to define detailed and complete technical specifications considering the key requirements identified by WP2 in D2.1 and D2.2, the reference architecture of D2.4 and the use-cases described in D6.1. The SECaaS specifications are based on the 4+1 architectural view model, which is consistent with the work developed in the rest of SecureIoT work-packages.

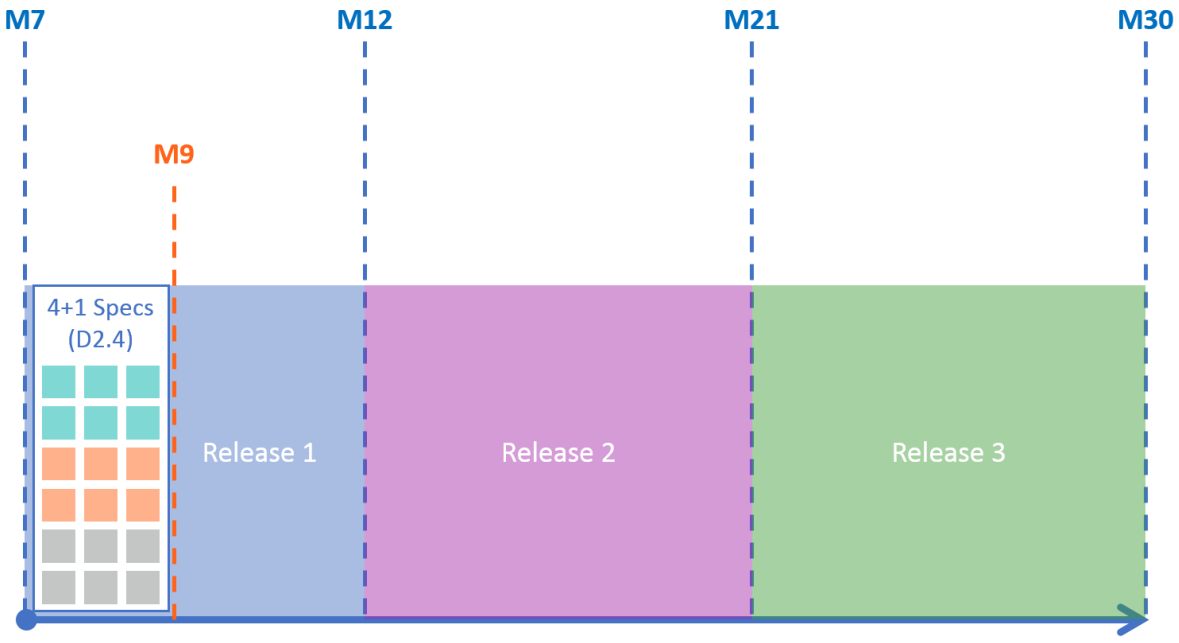


Figure 14: Timeframe for WP5 design and implementation activities

Each one of the releases will be divided into *sprints*, where a small subset of tasks will be addressed and completed. In the case of the first release, the first sprint has had a longer duration of two months in order to do the following actions:

- Analysis of requirements from D2.1, D2.2, D2.4 and D6.1
- The composition of 4+1 specifications, including the identification of involved actors, use-cases and scenarios to model key functionalities to be implemented by each SECaaS; sequence diagrams to describe the interactions between internal components and external components and users; functional architecture or logical view.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

- Identification of background technologies and assets to be reused and integrated, explaining the status prior to SecureIoT and potential extensions and improvement to be done within the project.
- Identification of baseline technologies, libraries and tools to be integrated, taking care of licensing aspects that may limit the future exploitation of SecureIoT SECaaS.
- Description of SECaaS interfaces and APIs.
- The composition of an initial backlog of tasks to complete the implementation and release of the corresponding SECaaS.

The homogeneous and consolidated approach promoted by T5.5 at WP5 level has enabled to maximize the synergies between all the SECaaS and to achieve similar levels of detailed and maturity. The results are visible through the first version of WP5 deliverable, i.e., D5.1, D5.4, D5.7 and D5.10.

The rest of sprints have a duration of one month, as it is proposed by Lean Kanban and DevOps good practices, have a holistic vision: they include not only implementation aspects but also testing, development and updates in the initial technical specifications. Each sprint ends with a work-package level meeting where the outcomes are analysed, extracting key conclusions and also planning the following sprint with the selection of the tasks to be addressed from the backlog. This process is illustrated in Figure 15 below:

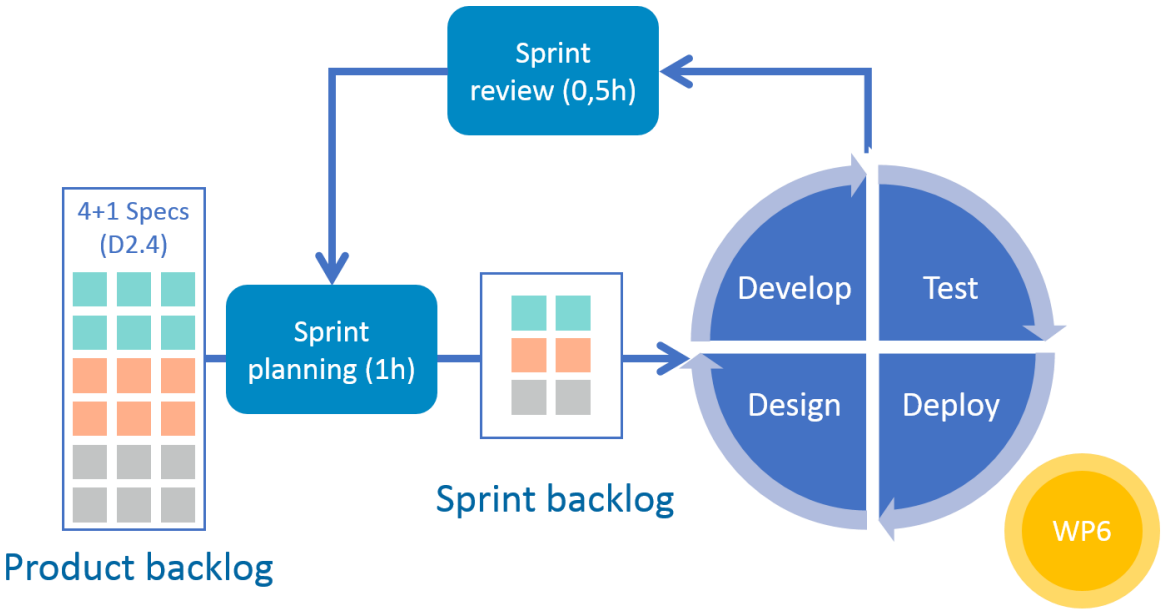


Figure 15: SecureIoT WP5 sprints methodology

6.1.3 Software development guidelines

While it is assumed that each SecureIoT partner and the engineers and developers involved in WP5 are perfectly aware of good and modern development practices, a small set of basic guidelines have been established.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

6.1.3.1 Source code management

- The Source Code Management (SCM) environment used to develop SecureIoT SECaaS will be GitLab community version[11]. In order to keep private the ongoing implementations and to warranty a high availability, secure and stable setup, ATOS provides this service which is accessible in https://gitlab.atosresearch.eu/. Periodic backups are configured in order to avoid losses of data in case of hardware failure.
- SecureIoT SECaaS will follow microservices paradigm proposed by Martin Fowler [4]. Thus, each microservice will have a single *codebase* as part of a single GitLab project or repository. Initially, each SECaaS has been identified as a microservice and the corresponding GitLab project created. However, during the execution of the project, this decision will be revisited according.
- All SECaaS must include a README.md file in the root folder of the repository including the following sections:
  - Background: introduction, mean features and architecture.
  - Install: from sources, use of Docker.
  - Instructions for end-users and developers.
  - APIs description.
  - License statement.
  - Third-party libraries used and licenses.

6.1.3.2 Branching model

Three branching models are proposed: Git flow, GitHub flow and GitLab flow. As it is discussed in detail in [5], Git flow encourages to use *master* branch only to deploy the software in production and developers are encouraged to use the *develop* branch. GitHub flow is a lightweight option since *develop* branch is not used anymore and all changes merged in master are deployed automatically relying on CI/CD tools. These two flows are presented in Figure 16. Finally, GitLab alternative offers a trade-off between the complexity of Git flow and the lack of GitHub flow with respect to the management and deployment of the *codebase* over different environments by introducing production and release branches. Although T5.5 recommends the use of GitLab flow, which is described by Figure 17, WP5 development teams are free to select the branching model that fits better their needs and mindsets. However, it is important to highlight that all SECaaS MUST be implemented following one of them.

[11] https://gitlab.com/

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

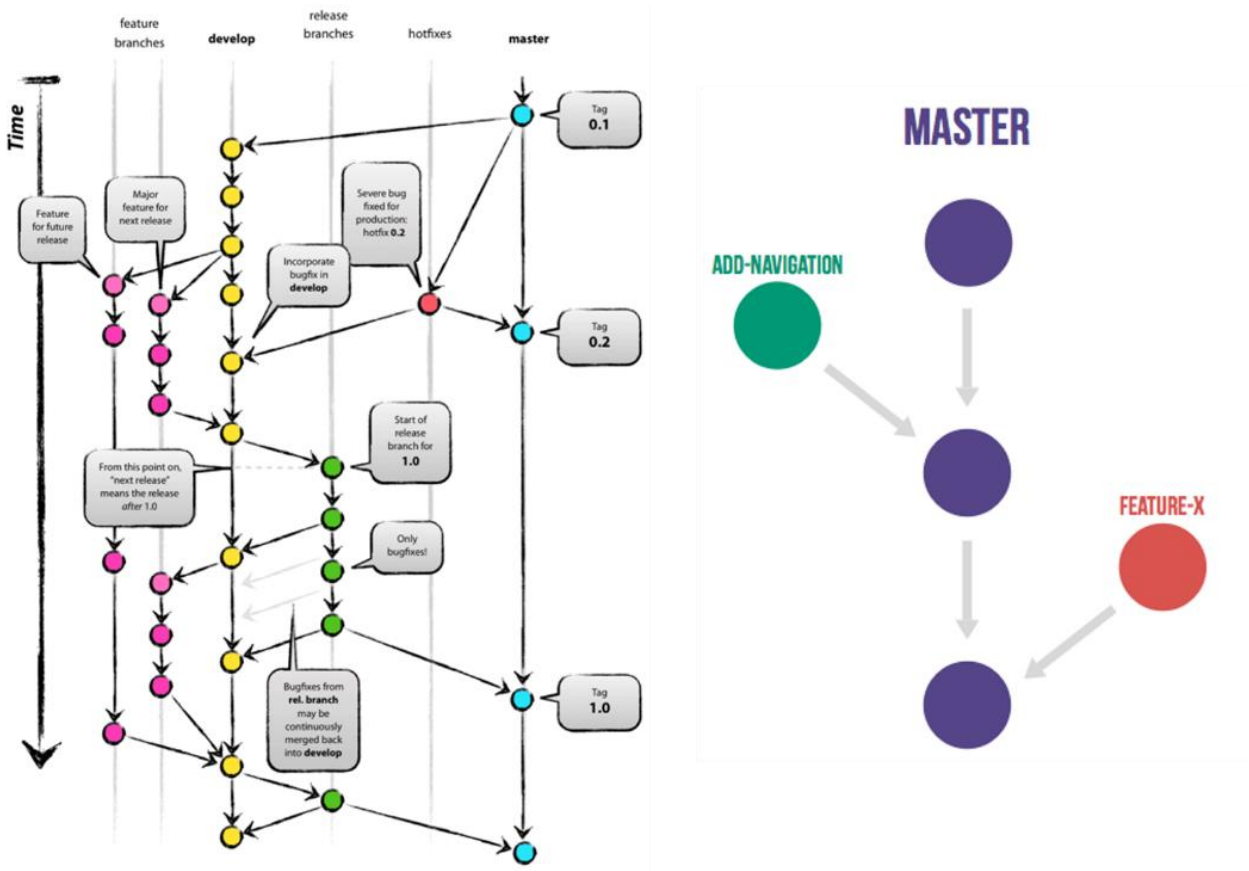
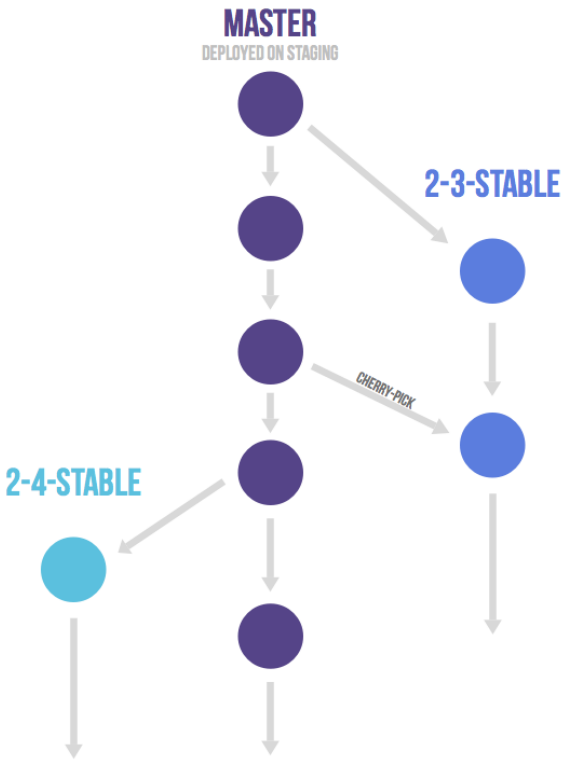


Figure 16: Git flow and GitHub flow [5]



Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Figure 17: GitLab flow [5]

6.1.3.3 Development lifecycle

An exhaustive and detailed list of best practices is documented in different online resources [6][7]. In particular, SecureIoT SECaaS must fulfil the following aspects:

- Master and develop branches will be protected. Thus, direct pushing is not possible.
- GitLab issues are used to document bugs, features to be implemented, improvements, etc. Each one of the items of the backlog must be included as issues.
- GitLab issues will be tagged properly to identify its nature (BUG, IMPROVEMENT, NEW FEATURE) and status (BACKLOG, IN PROGRESS, REVIEW NEEDED, COMPLETED).
- GitLab issues will be linked to milestones to facilitate tracking and monitoring.
- GitLab issues will be solved separate branches: one branch for each issue.
- Contributions to master branch will be done using Merge Request feature. They will contain a description explaining the background and motivation of the work done, the introduced changes, the pending issues (referenced also in the source code with *//TODO* comments) and reviewers if they are needed.
- Merge Request must also include tests and documentation.
- Merge Request cannot be accepted if CI/CT is not successful.

6.1.4 SecureIoT environment for CI, CT and CD

In 2006, M. Fowler proposed in [8] the following description for continuous integration: "*Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly".*

Although today the term Continuous Integrations is typically understood only as an automated mechanism to build software, the concept expressed by Fowler was wider and include the automation of three very relevant aspects of software life-cycle:

- Continuous testing, verification and validation.
- Continuous configuration so that a single codebase can be tailored to fit the specific requirement of different environments.
- Continuous deployment of the same single codebase over multiple hardware infrastructures.

Nowadays, a rich portfolio of commercial and open-source tools enables Continuous Integration, Testing, Configuration and Deployment to be applied and easily integrated with most environments. The next two subsections present the analysis that has been accomplished to select SecureIoT toolchain and the resulting ecosystem that has been set up for the benefit of SECaaS providers and WP6 use-cases' partners.

6.1.4.1 Analysis of the state-of-the-art

The purpose of this subsection is not to include a detailed and exhaustive report about available tools but to highlight which options have been considered and the main reasons that have led to the selection of SecureIoT toolchain.

Table 19: State of the art for continuous integration systems

Continuous integration system		
GitLab CI / GitLab runners	https://docs.gitlab.com/runner/	Virtual machines that allow running the CI/CD/CD jobs defined in a *YAML* file of a GitLab project, e.g., to build a Docker image or to use kubectl.
Jenkins	https://jenkins.io/	Open source automation project focused on CI/CT/CD. It has great flexibility thanks to plugins.
SecureIoT will leverage on **GitLab CI/ GitLab runners** since they provide a flexible and easy to use way of defining CI/CT/CD pipelines that can be tailored to its application to multiple technologies. It is available out-of-the-box in GitLab, so no additional servers and maintenance are needed.		

Table 20: State of the art for continuous testing tools

Continuous testing		
Travis-CI	https://travis-ci.org/	Open-source CI system that uses a YAML file to define how to build and test software, e.g., selection of Node.js version, required services to execute tests, instructions to build software and run tests, etc.
SonarQube	https://www.sonarqube.org/	An open-source automation system that allows running code linter and style checkers for different technologies, extracting quality metrics and KPIs.
Docker / Docker-compose	https://www.docker.com/	Although the scope and potential of Docker go beyond testing, it must be noted that it implies great benefits in this sense. For instance, using Docker and Docker-compose it is possible to set up and to run automatically end-to-end and unit testing.
GitLab CI files will include specific jobs to use **SonarQube** (or specific code linters and style checkers) to validate SECaaS code quality. Unit and e2e tests will be based on **Docker and**		

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

Docker-compose to allow easy setup of required components. The tests should achieve a code coverage percentage of 70-80%.

Table 21: State of the art for continuous deployment tools

Continuous deployment		
Nexus repository manager	https://help.sonatype.com/repomanager3	Artefacts repository that can be used to manage and distribute components relying on different technologies, e.g., C object files, text files, binaries, Java JAR-WAR-EAR, NPM packages or executables. Also, Docker images. Fine access control is one of its main features.
Docker swarm	https://docs.docker.com/engine/swarm/	Natively included in modern versions of Docker. It enables clusters of Docker (swarm) engines to be created and managed. Horizontal scaling and service discovery are supported.
Kubernetes	https://kubernetes.io/	An open-source platform for containers orchestration, deployment and scaling. It is a production technology and available in most widely used cloud platforms like AWS, Google Cloud or Azure.
Rancher	https://rancher.com/	An open-source platform that simplifies the administration and configuration of Kubernetes clusters and abstract the underlying technology.
SecureIoT hardware infrastructure will be managed from **Rancher (based on Kubernetes)** tool. The release of SECaaS will be done using **Docker** images which will be published in an internal **Nexus3** repository.		

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

6.1.4.2 Continuous integration, configuration, testing and deployment

SecureIoT WP5 DevOps environment has been designed to achieve a high-level degree of automation in terms of building, testing, configuration and deployment. The resulting pipeline is summarized in Figure 18:

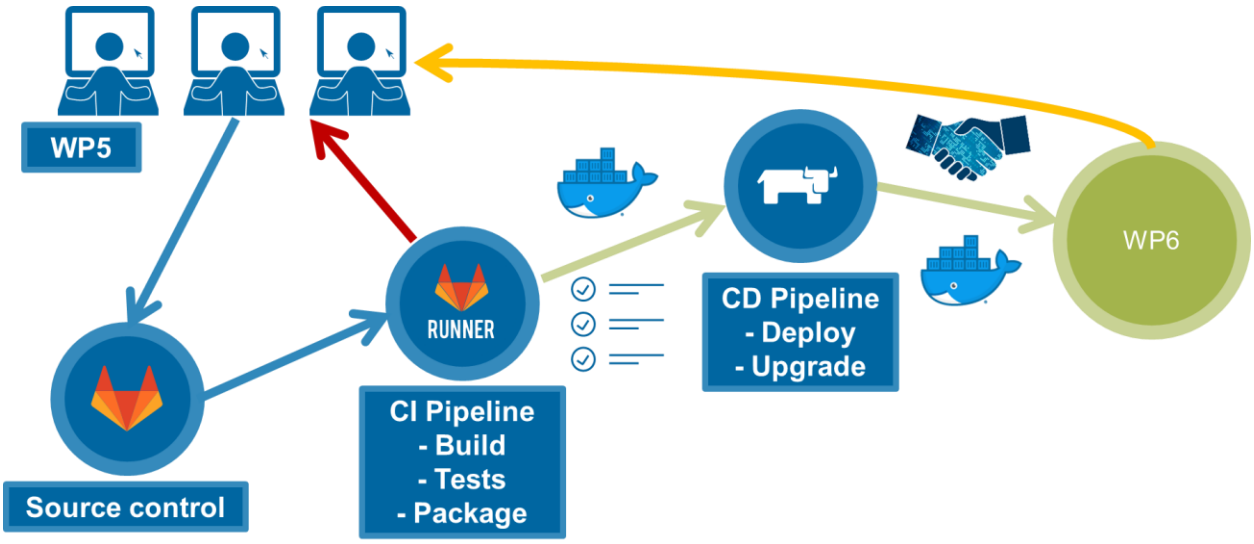


Figure 18: SecureIoT continuous integration, configuration, testing and deployment pipeline and toolchain

1. GitLab runners launch automatically CI pipelines whenever new changes are pushed to any branch of the repositories.
    a. They build the Docker images used to release the SECaaS, warrantying that the new changes do not break building the sources.
    b. They run unit testing using commands included in the Docker image, e.g., npm test for Node.js.
    c. They run e2e integration tests using Docker-compose files to setup all required components, e.g., MongoDB databases.
    d. They run code quality validation against SonarQube.
2. If changes are correct, they can be merged into the master or develop branches depending on the branching model, starting the CD pipeline in a new GitLab runner:
    a. CI pipeline is executed again.
    b. Docker images are published in Nexus3 artefacts repository.
    c. Kubectl[12] is called with the appropriate credentials in order to deploy or update the new Docker images in the corresponding hardware domain.

[12] https://kubernetes.io/docs/reference/kubectl/overview/

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

This ecosystem supposes an evolution and upgrade with respect to the typical infrastructures used in research projects and implies a relevant step towards the industrialization of SecureIoT SECaaS.

6.1.5 SecureIoT hardware infrastructure

SecureIoT hardware infrastructure is initially composed of three different clusters or environments:

- Controllers in order to host the different tools, i.e., Rancher/Kubernetes, GitLab runners, Nexus3.
- Production where stable versions of SECaaS prototypes are deployed. It is a testbed for use-cases.
- Pilots so that assistive robots, connected vehicle and industry4.0 applications can install specific components and technologies, e.g., FIWARE Generic Enablers.

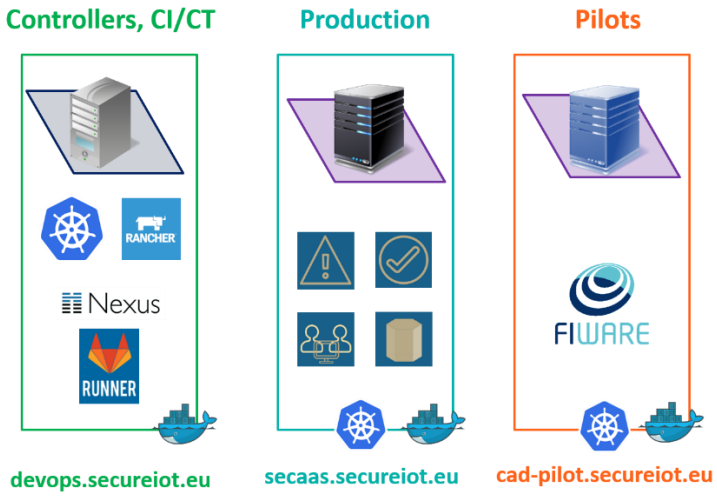


Figure 19: SecureIoT hardware infrastructure

Initially, each one of these clusters is composed by a single Virtual Machine (VM) running Ubuntu 16.04 LTS operating system. They have been provisioned with enough hardware resources in order to cope with demanding applications and services as it is showed in Table 22.

Table 22: Details for SecureIoT hardware infrastructure

Name	IP address	Operating system	RAM	HDD
devops.secureiot.eu	95.211.239.166	Ubuntu 16.04 LTS	8GB	220GB
production.secureiot.eu	95.211.239.168	Ubuntu 16.04 LTS	24GB	1TB
cad-pilot.secureiot.eu	95.211.239.169	Ubuntu 16.04 LTS	16GB	1TB

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

The three clusters are managed from Rancher/Kubernetes tool as it is demonstrated by Figure 20.

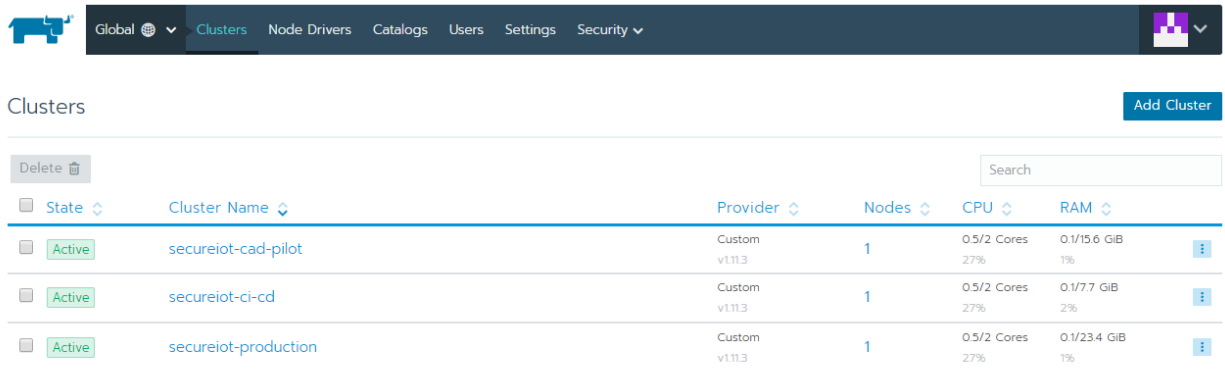


Figure 20: SecureIoT clusters managed by Rancher

Rancher capabilities include the possibility to monitor the use of hardware resources and the management of the containers deployed in each cluster.

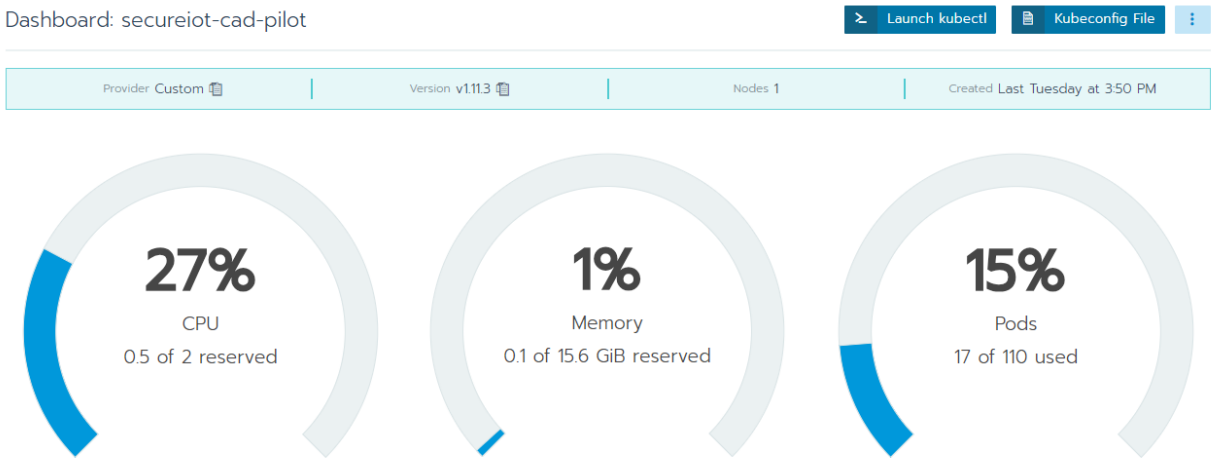


Figure 21: Monitoring of SecureIoT cluster by Rancher/Kubernetes

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

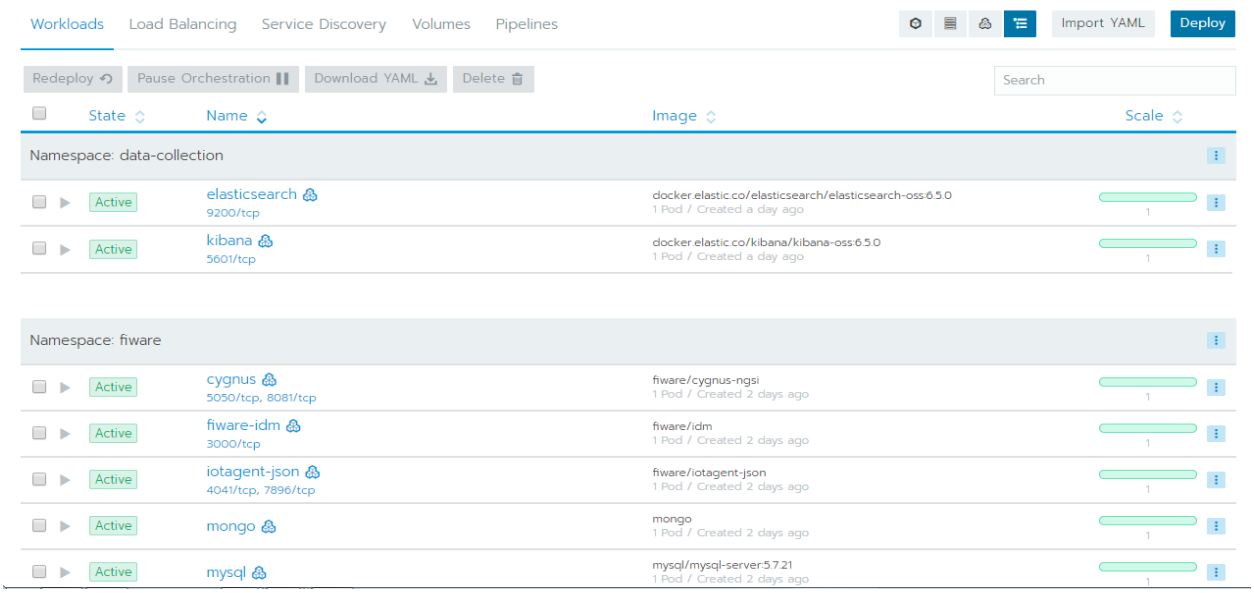


Figure 22: Management of SecureIoT containers with Rancher/Kubernetes

6.2 Continuous integration (CI), testing (CT) and deployment for Risk Assessment and mitigation services

The application and customization of the methodologies and technologies explained in section 6.1 for the Risk Assessment and Mitigation services will be described in detail in D5.2.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

7 Conclusions

This deliverable has specified the IoT risk assessment and mitigation functionalities of the SecureIoT platform, including both configuration and runtime functionalities. Moreover, it has specified the ways in which the risk assessment components of the project are integrated with other modules of the SecureIoT platform, fully in-line with the SecureIoT architecture. The specifications include also details about the APIs for accessing the risk assessment functionalities.

From a practical perspective, the risk assessment and mitigation service of the project provides the means for identifying and grading various risks that are applicable to the assets that comprise an IoT system. It also facilitates the identification and implementation of relevant mitigation actions. The implementation of both assessment and mitigation functionalities are driven by appropriate configurations of the risk assessment component, which are also applicable via proper APIs that are specified in this document.

Key to the operation of the risk assessment and mitigation services is their proper integration with the data collection and analytics services of the project, given that these services drive the identification of risks and of applicable mitigation actions. As part of the present demonstrator this integration is in its early stages at the level of a proof-of-concept and based on early implementations of the data collection and data analytics components of the project. The plan for releasing updated versions of this deliverable (including updated and enhanced demonstrators) involve the integration of the risk assessment component with more advanced version of the data collection and data analytics modules of the project. To this end, SecureIoT has implemented and put in-place a continuous integration methodology that emphasizes frequent integration and delivery of the risk assessment component of the project.

Moreover, we have described the plan to take advantage of the Risk Assessment Engine previously developed by Atos for using it as basis for the component. The current functionalities it offers will be extended and enhanced in order to fulfil the requirements of the use cases and IoT domain as presented in Section 2. Also, integration with the other components of SecureIoT (such as the assurance component, the security knowledge database, etc.) is a key element as we want to provide all services naturally connected with each other in order make SecureIoT a complete solution.

Although we have identified some extension features, we will continue updating the plan when we have a more advanced work in the integration of the SECaaS and its use in the use cases of the project. This way, we will continue updating the work and development of the tool for fulfilling the needs and special characteristics of the use cases.

Overall, the present deliverable represents significant progress in the detailed specification of the risk assessment and mitigation services of the project, including relevant functional specifications and APIs. It is accompanied by an initial demonstrator, which will be however fine-

tuned as part of subsequent versions. The latter will be reflected in deliverables D5.2 and D5.3 of the project, which will be enhanced and more advanced versions of the present deliverable.

Project Title: SecureIoT
Contract No. 779899
Project Coordinator: INTRASOFT International S.A.

References

[1] John Soldatos, Sofoklis Efremidis, et al., "D2.4: Architecture and Technical Specifications", SecureIoT H2020-IoT03-2017 - RIA Project, September 2018

[2] Philippe Kruchten, "Architectural Blueprints - The "4+1" View Model of Software Architecture", Paper published in IEEE Software 12, November 1995, pp. 42-50

[3] S. Gordiyenko, "Software Development Life Cycle (SDLC). Waterfall Model", 2014. . [Online]. Available: https://xbsoftware.com/blog/software-development-life-cycle-waterfall-model/. [Accessed: 19- Dec- 2018]

[4] M. Fowler, "Microservices", 2014. [Online]. Available: https://martinfowler.com/articles/microservices.html. [Accessed: 19- Dec- 2018]

[5] S. Sijbrandij, "GitLab flow", 2014. [Online]. Available: https:// about.gitlab.com/2014/09/29/gitlab-flow/. [Accessed: 19- Dec- 2018]

[6] "GitLab development guides". [Online]. Available: https://docs.gitlab.com/ee/development/.[Accessed: 19- Dec- 2018]

[7] "GitHub best practices". [Online]. Available: https://resources.github.com/videos/github-best-practices/. [Accessed: 19- Dec- 2018]

[8] M. Fowler, "Continuous Integration", 2006. [Online]. Available: https://martinfowler.com/articles/continuousIntegration.html. [Accessed: 19- Dec- 2018]