Title:	*Case studies implementation – Final version*
Authors:	*Arnor Solberg (TellU), Franck Fleurey (TellU), Sergio Jiménez Gómez (INDRA), Rocío Gómez Robledo (INDRA), Jon Colado García (INDRA), Francisco Parrilla Ayuso (INDRA), Erkuden Rios Velasco (TECNALIA), Eider Iturbe Zamalloa (TECNALIA), Angel Rego Fernandez (TECNALIA), Saturnino Martinez Melchor (TECNALIA), Miguel Angel Anton Gonzalez (TECNALIA), Sarah Tiphaine Ada Noye (TECNALIA), Rubén Mulero (TECNALIA), Larraitz Aranburu (TECNALIA), Sheila Puente (TECNALIA), Jose Manuel Olaizola (TECNALIA), Uģis Grīnbergs (BOSC), Modris Greitans (EDI), Oscar Zanutto (ISRAA), Emanuela Capotosto (ISRAA).*
Editor:	*Arnor Solberg (TellU)*
Reviewers:	*Nicolas Ferry (CNRS) and Anne Gallon (Evidian).*
Identifier:	*Deliverable D1.4*
Nature:	*Other (Software) with Report*
Date:	*01 March 2021*
Status:	*Final*
Diss. level:	*Public (with proprietary software available on request)*

The research leading to these results has received funding from the European Community's H2020 Programme under grant agreement n° 780351 (ENACT).

Executive Summary

The objective of the deliverable D1.4 is to describe the development and integration of the final Version (M38) of the ENACT Use Cases environment and scenarios performed within Task 1.2 according to the definitions and requirements described in D1.1 and D1.3.

The Use Cases environment comprises all the IoT infrastructures that are needed to perform Development and Operation of the Use Cases, including IoT and edge devices, the integration platforms, and the cloud services. In this deliverable, the final implementation of the use cases and the application of the ENACT enablers are reported. A main part of the deliverable is the software itself.

D1.5 describes more elaborated DevOps scenarios and validation with respect to KPIs and requirements that is put forward in the ENACT project in general.

The three different Use Cases implemented in ENACT are:

- **Intelligent Transport System (ITS):** The purpose of this use case application is to assess the feasibility of IoT services in the domain of train integrity control, in particular for the maintenance and logistics of the rolling stock and the on-track equipment.
- **Digital Health:** The purpose of this use case application is to provide medical device integration and data transport capabilities to external services such as an alarm centre or an electronic patient journal for stakeholders such as patients, doctors etc., exploiting the potential in IoT, edge and cloud services, adding more devices, actuation, distributed processing and better exploitation of collected data.
- **Smart Building:** The purpose of this use case application is to generate Smart Energy Efficiency applications and Smart Elderly Care applications, which will make use of sensors, actuators and services, in order to ensure the safety of the facilities to perform energy efficiency measures and also to support the care-takers in assessing the wellbeing of users.

TellU is leading this task and is responsible for the eHealth use case, Indra, and Tecnalia are responsible for the development and integration of the ITS and Smart Building, respectively. BOSC and EDI contribute to the implementation of the ITS case study and facilitate the rail test infrastructure and demonstrations. ISRAA has contributed to the evaluation and demonstration related to the eHealth use case.

The research leading to these results has received funding from the European Community's H2020 Programme under grant agreement n° 780351 (ENACT).

Members of the ENACT consortium:

SINTEF AS	Norway
BEAWRE DIGITAL SL	Spain
EVIDIAN SA	France
INDRA Sistemas SA	Spain
Fundacion Tecnalia Research & Innovation	Spain
TellU IOT AS	Norway
Centre National de la Recherche Scientifique	France
Universitaet Duisburg-Essen	Germany
MONTIMAGE EURL	France
Istituto per Servizi di Ricovero e Assistenza agli Anziani	Italy
Baltic Open Solution Center	Latvia
Elektronikas un Datorzinatnu Instituts	Latvia

Revision history

Date	Version	Author	Comments
17.12.2020	0.1	Arnor Solberg	Outline and some introductory text
25.01.2021	0.2	Arnor Solberg, Sergio Jiménez Gómez	Merged version with initial contributions from INDRA and TellU
09.02.2021	0.3	Arnor Solberg, Miguel Angel Anton Gonzalez, Oscar Zanutto	First complete draft with contributions from INDRA, Tecnalia and TellU, ready for internal reviews and inputs from technical providers
15.02.2021	0.8	Arnor Solberg	Merged input from all contributors. Ready for internal review
19.02.2021	0.9	Anne Gallon	Internal review
21.02.2021	0.9	Nicolas Ferry	Internal review
28.02.2021	1.0	Arnor Solberg, Miguel Angel Anton Gonzalez, Sergio Jiménez Gómez	Updated version after internal review
12.03.2021	1.0	Arnor Solberg, Nicolas Ferry	Final updates according to internal review

Contents

Table of Figures

1 Abbreviations and Definitions

Term	Definition
ACM	Actuation Conflict Manager
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
ARM	Advanced RISC Machine
BDA	Behavioural Drift Analysis
BLE	Bluetooth Low Energy
CTC	Centralized Traffic Control
DevOps	Development and Operations
DMI	Driver Machine Interface
DND	Dependency Network Diagram
eGW	Enhanced Gateway
EPJ	Electronic Patient Journals
ERTMS	European Rail Traffic Management System
GNSS	Global Navigation Satellite System
GRDP	General Registry of Data Protection
GSM-R	Global System for Mobile Communications – Railway
GUI	Graphical User Interface
GW	Gateway
HLA	High Level Architecture
HVAC	Heating, Ventilation and Air Conditioning
ID	Identification
IoT	Internet of Things
ISO	International Organization for Standardization
ITS	Intelligent Transport Systems
JSON	JavaScript Object Notation
KP	Knowledge Processor
KPI	Key Performance Indicator
L&M	Logistics and Maintenance
LED	Light-Emitting Diode
MQTT	Message Queue Telemetry Transport
MW	Middleware
OPC-UA	OPC Unified Architecture
OPEX	OPerating EXpense
OSI	Open System Interconnection
OTI	On-board Train Integrity
PLC	Programmable Logic Controller
QoS	Quality of Service
RCA	Root Cause Analysis
RFID	Radio-Frequency Identification
SIB	Semantic Information Broker
SIS	Smart IoT Systems
STCC	Smart Train Composition Coupling
SW	Software

TCP/IP	Transmission Control Protocol / Internet Protocol
TDS	Train Detection Systems
V&V	Validations and Verification
VPN	Virtual Private Network
WSN	Wireless Sensor Network

2 Introduction and Objectives

The goal of this document is to report the work performed in second phase of Task 1.2 towards the final construction and implementation of the three use cases of the ENACT project: Intelligent Transport System (ITS), Digital Health, and Smart Building. The application of the ENACT DevOps Enablers are reported in this document.

2.1 Context

The three use cases implemented in ENACT are the following:

- **Intelligent Transport System (ITS):** the maintenance and logistics of the rolling stock and the on-track equipment. Led by INDRA, and contributed by EDI and BOSC.
- **Digital Health:** medical device integration and data transport capabilities to external services such as an alarm centre or an electronic patient journal for stakeholders such as patients, doctors etc. Led by TellU as part of their remote patient monitoring product, piloted in ISRAA.
- **Smart Building:** Smart Energy Efficiency applications and Smart Elderly Care applications, which makes use of sensors, actuators and services, to perform energy efficiency measures and to support the care-takers in assessing the wellbeing of users. Led by Tecnalia, with a complemented experimental implementation by CNRS.

DevOps is significant for the efficient development and operation of all the three use cases.

For ITS, INDRA, as the system integrator, needs to continuously test and evaluate the subsystems with both software and hardware from their suppliers (i.e., EDI and BOSC in this project), and adjust the design and implementation of their main services accordingly in order to maximize the integration of the subsystems. DevOps guarantees the effectiveness of the integration process and provides real-time feedback to both the integrator and the suppliers as reference for subsequent development activities.

For digital health, TellU needs to continuously add new features to their remote patient monitoring product, involving not only the backend service running in the cloud but also the front-end services running on their hundreds of gateways already distributed to the patients. One example is that during the pandemic, they quickly adapted their product for a local government in Norway to monitor COVID-19 patient quarantined at home[1].

For smart building, KUBIK serves as a testbed for researchers in the whole Tecnalia institute to quickly build proof-of-concept prototypes for their ideas for the future building. A DevOps environment for quickly adjusting the prototypes is critical for researchers to evaluate and demonstrate their ideas.

In the meantime, trustworthiness is of paramount importance for all the three use cases, from different perspectives.

For ITS, the most important aspect is the safety of the train. They need thorough testing of both the subsystems from their suppliers and the integration logic. On the other hand, since the integrated system will be running on many trains eventually, they need to consider and evaluate scalability from the beginning, even before they actually invest into a large number of physical devices.

For Digital Health, the most important aspect is the privacy protection of the patient data, because the gateways are used to collect personal data from elderly people or patients, who are more vulnerable to privacy issues. Robustness and resilience are also important for their product.

[1] https://www.dagensmedisin.no/artikler/2020/11/10/who-vil-vite-mer-om-digital-oppfolging-av-covid-pasienter-fra-bodo/?fbclid=IwAR0wfaztYa222OcQlKR03ws4jDNbFgQxLIVad4jiGvRQv87G1igxCc6pLLQ In Norwegian.

For Smart Building, security is most important, because a large and complex system such as a smart building can easily be a target for attacks, and subsequence may be significant. In addition, Smart Building typically involves large set of actuators, these must be carefully managed.

The three use cases successfully applied ENACT DevOps enablers to support their development and operation activities, and to assure the trustworthiness of the implementations. The rest of this document will present the three implemented use cases, and their application of the ENACT enablers to support the DevOps practice.

2.2 Main Achievements

This document describes the implementation of the three use cases and application of the ENACT enablers for the development of these use cases. The table below lists the main software artefacts that compose each use case implementation, and the ENACT enablers used by the implementation.

Use case	Artefacts	Enablers applied
INDRA use case on SIT	The main developments carried into the ITS Use Case are made in two different areas: • Configuration: This part is made to enable the connectivity and the integration of all the elements that form the architecture and the ENACT DevOps tools. • Software: The Things, Gateways, Metadata services,… are made to reach the Use Case objectives. These developments are carried by EDI and Indra. These both sections are included into the following repositories: The software and configurations developed by Indra: • https://bitbucket.mova.indra.es/scm/iotsafety/its_enact_public.git The software and configurations developed by EDI: • http://git.edi.lv/enact-project	Test and Simulation (3.1.3.2), Root Cause Analysis (3.1.3.3), GeneSIS (3.1.3.1), Security and Privacy Control and Monitoring (3.1.3.6) Actuation Conflict Manager and Behavioral Drift Analysis (3.1.3.5)
TellU use case on Digital Health	The main part of TellU's RPM application is the front-end services running on gateways. The source code can be found here: • https://github.com/TelluIoT/gateway-rpm (production version, proprietary code, access on request) • https://github.com/TelluIoT/gateway-enact (demo version based on the production, open source) Part of the product is built and distributed via different partners: • Health personnel application: https://mao.shepherd.telenor.no/ • Patient app on Android https://play.google.com/store/apps/details?id=no.telenor.dialogg - Patient app on iOS: https://apps.apple.com/no/app/dialogg/id1467874186?l=nb	Risk management (Section 3.2.3.1) Context aware access control (Section 3.2.3.2) ThingML, DivEnact and GeneSIS (Section 3.2.3.3) Test and simulation (Section 3.2.3.4)

Tecnalia use case on Smart Building	The main developments in the Smart Building use case are related to the SMOOL middleware and its adaptation to the building environment and the security in sending orders to actuators: The SMOOL server and the SDK: • https://bitbucket.org/jasonjxm/smool/ (wiki with software download, quick start guide and example videos and presentations) The clients or KPs specifically developed for the ENACT project with the most common sensors and actuators for the Smart Building environment and the integration with ThingML: • https://gitlab.com/enact/smool_enact (Java examples of KPs to connect/share data by using SMOOL middleware as IoT broker) The implementation of secure actuation in KUBIK building using SMOOL, with the EnactKubik (producer) and EnactKubikConsumer KPs, and code of Node-RED applications for user comfort, energy efficiency, Z-Wave IoT message converter to SMOOL middleware and sensor data flows: • https://gitlab.com/enact/kubik_enact (Java adaptations of the SMOOL producer and consumer to make secure actuation and Node-RED IoT applications) The applications developed in the context of the complementary CNRS experimental lab (based on a network with devices from both KUBIK and the lab via SMOOL) https://gitlab.com/enact/smart	Risk Management (Section 3.3.3.1) Actuation Conflict Management and Behavioural Drift Analysis (Section 3.3.3.3) Security and Privacy Monitoring and Control (Section 3.3.3.4) Online Learning (Section 3.3.3.5)

2.3 Structure of the document

The main content of this document is Section 3 that reports on the final use case implementation. Section 3 follows the same structure for each use case describing the overall application, the implementation architecture the actual application of ENACT enablers, short summary of the Software Artifacts and outlook and further work related to the use cases and the application of ENACT enablers. Section 4 concludes this deliverable.

3 Final Use case implementation

This section presents each of the use cases and the implementation of each of the three solutions from a use case perspective, that is, the objectives and general features of each system (ITS, Digital Health and Smart Building) to be delivered to end users. For each use case, the general architecture of the Smart IoT System is presented, the end-to-end functionalities that belong to each use case are described, and the application of the ENACT enablers.

3.1 ITS Domain (Rail)

D1.2 establishes the basis to the developments that are planned into the ITS Use Case. This basis includes both the ITS Use Case developments and adaptations made for the Use Case itself and the ENACT DevOps tools integration. The D1.3 shows the validation results and the accomplishment degree of the KPIs defined in the D1.1 at mid-term of the project. In this deliverable, the full use case has been implemented and delivered, leveraging the ENACT enablers.

The validation and verification procedures will be described in the D1.5 ***"Final evaluation and validation Report"*** serving this D1.4 as an input for that purpose. Along this chapter: (i) the implementations and integration description, (ii) the improvements of the ITS Use Case functionalities obtained by using the ENACT enablers, and (iii) further resources that show the work developed as part of the Use Cases will be described.

A summary of the main achievements and impact of the ITS Use Case both in terms of business level achievements as well as of technical and validation achievements are summarized in the following:

- The Use Case itself, taking as a reference the initial stage of the ENACT project, has evolved from a to be integrated early prototype to a functional system ready to be deployed on a real rail infrastructure. The internal objective for the ITS Use Case, from the INDRA perspective, was improving the prototype connectivity with third parties that includes the ENACT DevOps tools through a Cloud/Gateway infrastructure using the AMQP protocol and in the edge layer with the EDI Things and the ENACT DevOps enablers, if it would be required, by MQTT. These protocols are intended to provide flexibility to the ITS Use Case infrastructure and keep the security and safety policies required for the rail environment.
- Another main achievement for the ITS Use Case is the integration of an open source Cloud platform that has been fostered by the European Commission (EC) as a reference Cloud platform. The integration has been accomplished and the FIWARE platform used to store and to represent data collected into the ITS Use Case.
- As a final main achievement for the ITS Use Case itself, several functionalities have been enabled inspired by the ENACT enablers requirements, to enhance the systems that form the use case. In this point, it is important to make an emphasis on the ITS Gateway. This gateway has been improved during the ENACT project to make it ready for a production and deployment in real scenarios stages. It is expected for the next years to deploy similar infrastructures, as the one brought to ENACT, in real scenarios for Rail Operators and Managers.
- INDRA has exploited several of the ENACT enablers. In particular:
    - The Docker and Kubernetes technologies have been studied as software deployment technologies for the ITS Gateway and the Cloud infrastructure, the GeneSIS tool, which extend over these technologies with platform independent reliability mechanisms, has been included as a main component for the deployment. These technologies are enabling a fast and an agile deployment on the devices in real environments.
    - A Testing and Simulation tool has been integrated with the ITS Use Case infrastructure in order to evaluate the performance of the ITS Gateway. This implies a remarkable milestone as it provides information about the ITS Gateway capacity to handle the different information that it manages. Moreover, this evaluation of the ITS Gateway

provides information about its scalability and how many ITS Gateways are required in a scenario.

- The Root Cause Analysis (RCA) tool has been studied as a main tool to identify failures in the ITS infrastructure network. This tool has successfully identified two types of failures into the ITS Use Case infrastructure network. The important point is that this tool has proved the potential to identify more failures that may appear in the ITS Use Case infrastructure in real scenario deployments.
- Other ENACT DevOps tools have been integrated to enhance these agile and fast set up real of environment scenarios. These tools are Behavioural Drift Analysis (BDA), Actuation Conflict Management (ACM) and Security & Privacy Monitoring and Control (S&P Mon&Control).

3.1.1 Overall Application Description

The ITS Use Case is in charge of providing a valid verification and validation scenario for the ENACT tools. The functionalities and features that the Use Case brings to the project are intended to fulfil specific objectives that are aligned with the ENACT objectives. The further functionalities and the architecture are aligned with different points of the DoA as follows:

- *DoA Point 1: "The Internet of Things has made us to think of services unimaginable just a few years ago. However, due to the particularities of the rail market - large infrastructures and rolling stock are needed to cover safety certifications, expensive field tests, etc"*

  Large infrastructure issues are solved providing a flexible set of components in the Cloud and the Edge and IoT layers, referred to as the Gateway layers in the following, and presented along the ITS Use Case section. The Train Integrity and Logistics and Maintenance (L&M) functionalities, explained in the following paragraphs, are intended to align the IoT and the rail progress. These functionalities are based on an architecture that has as a main component the ITS Gateways. The ITS Gateways are entities flexible to be connected to each other and to any layer. These components can be scaled as desired as the atomic components of each layer are functionally independent and designed to be connected in a safety and secured manner. Moreover, the transversal services (authentication, naming…) that they require are allocated in a Cloud platform, therefore, they are equally scalable.

- *DoA Point 2: "IoT services in the domain of train integrity control, in particular for the maintenance and logistics of the rolling stock and the on-track equipment."*

  As part of the use case, several IoT services running across the Cloud-Edge-IoT continuum, which provide the main and backup functionalities, were developed to track the train composition and each element, besides their completeness status and other maintenance parameters. At the time being, the project just includes energy consumption as a maintenance parameter; however, the system is designed to include any maintenance parameter by adding the sensor hardware required to make the measurement and a minor configuration file.

- *DoA Points 3, 4 & 5: "Constructing a powerful environment for simulation and testing of smart Rail IoT services, in order to guarantee the trustworthiness of these services before deploying them to the on-board system." & "Enabling predictive maintenance and increasing safety of Rail IoT services using the context data that cannot be obtained by wired sensors, based on the ENACT trustworthiness toolkit." & "Increasing the operational efficiency by permanently and remotely monitoring the status and position of key elements of the rolling stock and the on-track equipment, perform proper reaction based on notifications and actuators end exploiting cloud resources for advanced processing and learning."*

  Apart of the physical hardware to test and show the functionalities, a virtual environment has been created to integrate the ENACT DevOps tools remotely as the partners facilities are located

in different countries and to bypass the COVID-19 restrictions. For example, this infrastructure is used by the Testing and Simulation (T&S) tool to record a real train behaviour and to simulate scalability application and the functionalities themselves before they are deployed. This virtual infrastructure is also used by the RCA tool. This tool records the gateways status to evaluate the failures that may appear.

Aligned with these objectives, this section will introduce first the functionalities provided by the use case by its own and after the role of the ENACT enablers for the Development and Operation of these functionalities as well as for the infrastructure that forms the Use Case.

Two basic services have been developed into the ITS Use Case framework (see Figure 1). These services are intended to provide data to monitor the position of the train and maintenance data using IoT systems. These services are (i) Train Integrity and (ii) Logistic and Maintenance (L&M):

- **Train Integrity:** The train Integrity service provides an indicator describing the completeness of the train. In other words, it is a service that allows the user to know if the wagons and the loco that form the train compositions are travelling following rail safety conditions or not. This service is based on a Things infrastructure that monitors the acceleration, the position of each wagon/loco, and the distance between them continuously. These data are stored and processed to generate this Train Integrity indicator.
  It must be remarked that both the Train Integrity indicator and the bare data (wagons acceleration, distance between wagons and their positions) can be provided to any authorized user integrated into the ITS Use Case infrastructure (see D1.1 for more details). This service is considered Safety, it is required to run the train mission.

- **Logistic and Maintenance:** The L&M service provides information about specific magnitudes that can be measured into the train composition (both wagons and loco). These magnitudes cover a variety of aspects such as wagons temperature, position, energy consumption, humidity, pressure at the axis, etc. In this use case, the energy consumption and wagons position values are brought. This service is not considered Safety as the train mission can be run in case of this service is not working properly.

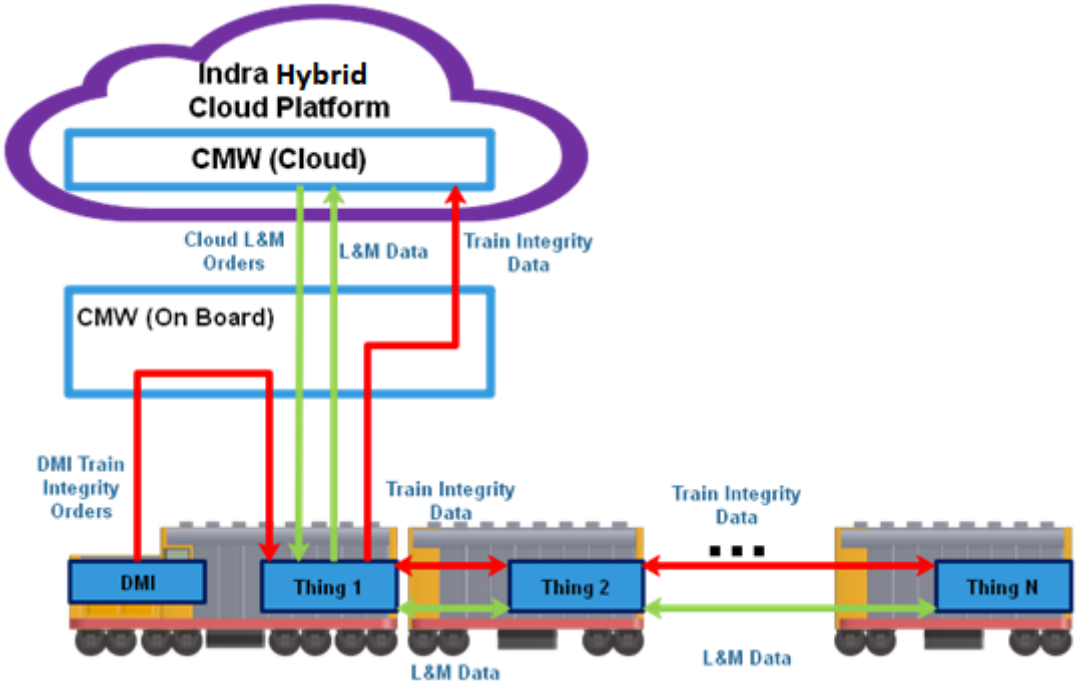


Figure 1: Train Integrity and L&M Overview

The extra functionality marked as pending point in D1.2 has been added. The ITS gateway has a new important function apart of routing the messages from the other entities that is providing data about its own state. They are able to provide data about their state in two different aspects: Physical and routing status. These reports can be used by ITS Use Case authorised users integrated into the ITS Use Case. More precisely, these reports contain:

- Physical data: CPU usage, RAM usage, power consumption, disk usage.
- Routing data: Number of gateway connections, number of gateway up connections, published message size, received message size, number of messages published per second, number of messages received per second, and messages delay.

With all these functionalities, the ITS Use Case own functionalities are covered to accomplish the ITS Use Case challenge and the integration with the ENACT tools.

3.1.2 Architecture and main scenarios of final implementation

The architecture section is intended to describe all the elements that form the ITS Use Case, how they are related and the reason behind those links.

The architecture that appears as a consequence of the developments, described in the last sections, carried in the 3 years project can be shown in the next Figure 2:

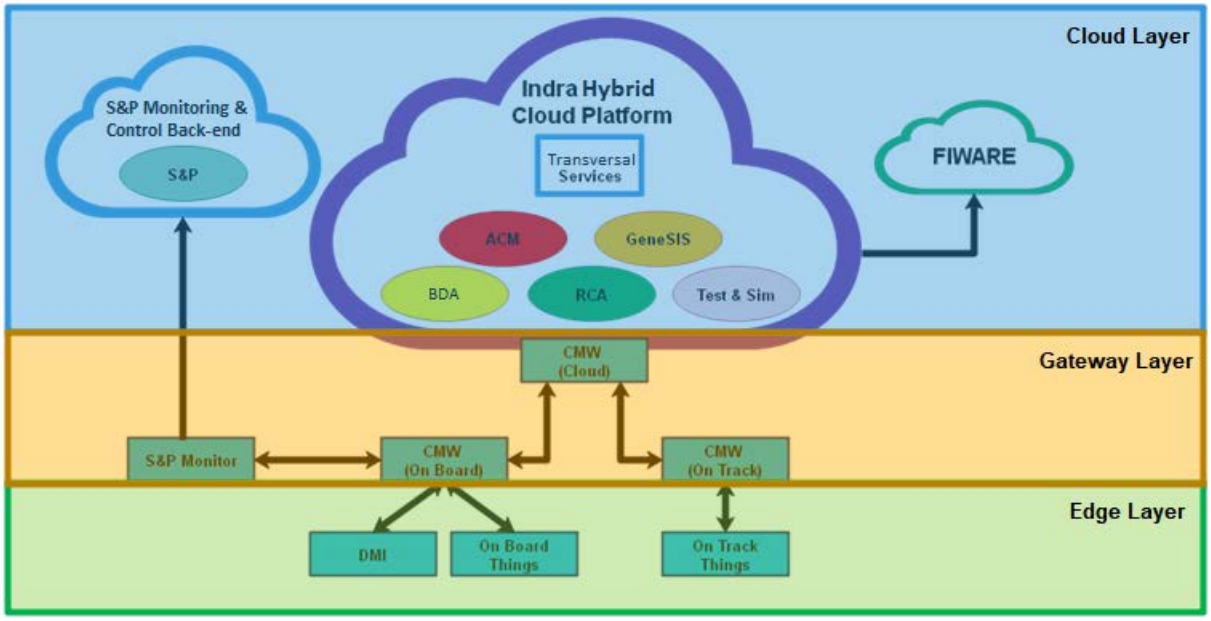


Figure 2: ITS Architecture

The architecture is divided in three layers. Further details about the elements that form these layers are given:

- **Edge Layer:** The edge layer is in charge of providing the bare data to the ITS Use Case. The following elements form it:
  - Things: The key element into this layer is defined under the name of Thing. The Things provide the L&M and Train Integrity parameters described above.

    The Things are defined as a group of nodes which are coordinators, sensors or actuators. In such a way, the Things have several capabilities: gather, actuate and communicate.

    It must be emphasized that the layer covers the On Board and On Track sections, the Train Integrity parameters are exclusively treated On Board; however, the On Track section participates into the L&M data provision with RFID tag that reports information

from the On Track infrastructure. These Things (see Figure 3) are powered up by an energy harvesting system to cover the case that no electrification systems are equipped On Board a train. These Things have been developed during the project by EDI and the energy harvesting system by BOSC. The following shows the Things involved into the ENACT project.

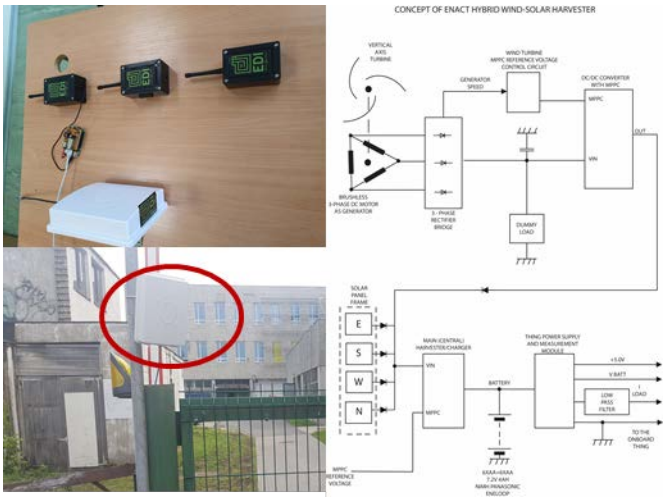


Figure 3: On Board and On Track Things

The Things of the system consist of:

Sensors:

- On Board: an accelerometer, a Received Signal Strength Indicator (RSSI) sensor, a Global Navigation Satellite System (GNSS) receiver, and Radio Frequency Identification (RFID).
- On Track: Radio Frequency Identification (RFID) reader.

Actuators:

- On Board: Light-Emitting Diode (LED)s and displays.

The Actuators listed are not considered in any functionality related with neither the ITS Use Case (Train Integrity and L&M) nor ENACT tools functionality into the ITS Use Case. The reason for it is that the actuators equipped in the Things are not suitable with the messaging architecture brought to the ENACT ITS Use Case.

- DMI: The DMI (Driver Machine Interface) is included into this layer and is in charge of managing the Things into the train by the driver for the Train Integrity functionality and to calculate the Train Integrity indicator based on the data reported by the On Board Things.

- S&P Mon&Control: The S&P Mon&Control is in charge of two main objectives: (i) security monitoring of the traffic that is carried by the gateway layer including notifications and situational awareness, and (ii) launching reaction mechanisms in case anomalies in the traffic behaviour or intrusions are detected. It must be emphasized that the network traffic analysis is focused on traffic security and not on the business data related to the ITS functionalities.

  This enabler is composed of several differentiated elements:

  - Hardware: A hardware host device offers the computing capacity to the monitoring agent software (see details below), the connectivity with the Communication MiddleWare (CMW) via Ethernet and the connectivity with the S&P Mon&Control back-end (see details below) through a 3G/4G interface.

The piece of hardware itself was provided by INDRA and it can be seen in the Figure 4:

Figure 4: The S&P Mon&Control monitoring agent host

- Monitoring agent software: The monitoring agent developed by Tecnalia and installed in the hardware host sniffs the traffic from the CMW in order to be treated and to be sent to the S&P Mon&Control back-end in the Cloud for a further analysis.
- User revocation software: The communications analysis by the S&P Mon&Control back-end could throw as a result that the users that generate the traffic being monitored are intruders, and therefore, an immediate notification is published to this entity which contains the information on the user identification. This software developed by Indra revokes the permissions to the user in the ITS Central Authentication services as part of the control of the S&P Mon&Control Enabler.

- **Gateway Layer:** A gateway, so called CMW (Communication Middleware), is the main gateway layer component that connects all the ITS Use Case layer components (Cloud services, Things,…), it collects and gathers the edge data and provides it to the Cloud. Moreover, it gathers the Cloud order to the Things to start the L&M service. This CMW is based on Indra developed solution certified for rail environments. The ITS CMW can be seen in the Figure 5.

  The basic functionalities that have been developed previous the ENACT project, but refined into the project, in a general perspective of a CMW are listed below:
  - Routing the messages from the different services to be sent to the different architecture entities. This routing follows a preconfigured topology to guarantee the provision of the information in a safe and secured manner.
  - Synchronization tasks to keep all the devices working with the same time reference.
  - Authentication tasks at different OSI levels to enable the connectivity of the edge elements and to enable the connection of the CMW with the Cloud layer.

Figure 5: ITS CMW/Gateway

As explained in D1.1, the scalability of the solution is an important concern for the use case as a whole. Indeed, in a usual high-speed train line around 1300 wagons must be covered in several single, double, and mixed compositions. Therefore, the number of Things and CMWs can be increased significantly.

As detailed previously, the CMW being a core component linking the Things layer and the Cloud it is considered as a key point of failure in the system. On the one hand, it is critical to understand the extent to which a gateway is able to scale (increase in number of sensor and actuator data). On the other hand, it is equally important to understand if the ITS system is able to handle a large number of gateways. For these reasons we decided in the ENACT project to focus on scalability issues at the CMW.

As the CMW equipment cost is high to provide several of these devices to the project, considering the budget, the scalability tests will be done in a single physical CMW and several virtual ones will be provided for testing. These virtual gateways are equally valid as they simulate most of the physical gateway characteristics:

- Same functionalities deployed.
- Same physical resources (RAM, CPU,…) enabled.
- The same number of interfaces are enabled.
- The main difference is that the power consumption is not an issue for VMs.

Therefore, the general architecture is formed by a single physical CMW located On Board and the several virtual CMWs.

- **Cloud:** The Cloud layer is formed by the Cloud platforms that participate on the ITS Use Case and how they are related. Three different Cloud platforms are considered: FIWARE, Indra Hybrid Cloud Platform, and S&P Monitor and Control Back-end.

    - Indra Hybrid Cloud Platform: The cloud is a hybrid solution that combines a private and public part. The private part is in charge of the internal ITS task such as edge data storage, edge elements authentication, and external partner's authentication. The public part is in charge of integrating external elements such as tools or other Cloud platforms. This is the integration point made for all the Cloud resources and DevOps tools that requires it. All the DevOps tools, except the S&P tool, are integrated in this layer to accomplish their functions. Moreover, it is the integration point also for the FIWARE Cloud platform.

    - FIWARE: FIWARE is an open source platform that the European Union supports as a future platform to provide Cloud services. For this Use Case, we use FIWARE to provide users with several dashboards to track the Use Case functionalities outputs (making use of the Orion context broker and QuantumLeap as the integration components and Grafana as the presentation tool).

    - S&P Monitor and Control Back-end: It is a Cloud platform to the ENACT project to evaluate the S&P DevOps tool data. The Safety and Security data treated in the S&P monitor is sent to this Cloud platform to be evaluated and to provide to the Use Case user an interface to interact with the Safety and Security rules that are desired for the On-Board installation and the functionalities.

3.1.3 Application of the ENACT enablers

The impact of the ENACT enablers application is based on the ITS Use Case functionalities described in the previous section. Further details about the impact, benefits and achievements are described in the following section. As described in the D1.1 the ENACT enablers integrated in the ITS Use Case are:

- **Orchestration and Continuous Deployment Enabler:** This tool, also known as GeneSIS, supports the continuous deployment of SIS across the Cloud-Edge-IoT space. It has been used in the context of this use case to deploy the software components on the CMW. Deployment is triggered automatically, when a new version of the software is released, following the DevOps principles. A deployment is triggered only when the train is in a state considered as appropriate for it. Deployment is performed using the rolling deployment strategy to ensure maximum continuity of service.
- **Test, Emulation and Simulation Enabler:** This enabler supports the simulation of sensors and actuators infrastructures, in particular it provides mean to record data from the real system (and thus from real sensors/actuators) and to replay it. While replaying the data, the enabler can inject failures and attacks enabling to test the system against these. In addition, the data can be replayed multiple times to test the capacity of a system to scale. In the context of this use case, the test and simulation enabler has been used in particular to test the ability of the SIS to scale with a focus on the CMWs.
- **Context Monitoring and Actuation Conflict Management Enabler:** BDA and ACM are the tools developed into this enabler framework. BDA provides mechanisms to understand to which extent a smart IoT system is effective in reaching its goals (a behavioural drift). BDA also allows to analyse drifts with the production of new behavioural models learned from the observed behaviours. As part of the use case, it is used to observe the behaviour of simulated trains (e.g. kinematic behaviour of the train in terms of speed/acceleration/pitch and in terms of passenger's safety on allowing doors to open only when stationary, etc.). ACM supports the identification and management of actuation conflicts – i.e., actuation orders with conflicting goals. This tool has been used on a simulated train to manage conflicts arising when information are emitted by the applications to the driver interaction equipment’s.
- **Security and Privacy Monitoring and Control Enabler:** The S&P monitoring and control tool is the tool used associated to this enabler. The tool is in charge of enhancing the ITS Use Case security and safety characteristics for the On Board devices. The On Board devices network parameters are monitored in the S&P Monitor device (cf. 3.1.3.6) and sent to the S&P Monitor and Control Back-end to detect predefined security threats. In the use case, the tool did not focus on the business data managed in the On Board devices, only in the network parameters.
- **Run-time Quality Assurance and Root Cause Analysis Enabler:** The RCA is the tool used associated to this enabler. RCA collects the monitoring data concerning the hardware usage (e.g., CPU, RAM, power consumption, disk usage) and connectivity status (e.g., number of gateway connections, number of gateway up connections, published message size, received message size, number of messages published per second, number of messages received per second, and messages delay) and visualizes them by the charts on dedicated dashboards. RCA needs to first learn the images of the system reflected in aforementioned monitoring data when it is properly functioning. It is then possible to actively inject the failures/ problems with known causes to the ITS so that those incidents’ patterns can be extracted and learned by the RCA. This leads to the construction of a historical database of proper functioning events and observed incidents with known causes, impacts and mitigation actions to be taken. Afterwards, RCA is monitoring the system in real time and calculating the similarity score of the actual state and each of the learned events (normal states and incidents) in the historical data. In consequence, when the system is seen too different from observed normal states, the administrator/ DevOps engineer can be notified so that the potential current abnormal behaviour can be learned and updated to the historical data. Similarly, the repetition of a failure/ problem reflected by a high score can be alerted and corresponding root-causes, impacts and mitigation actions can be suggested to react quickly to the incident.

In the following sections we detail how these tools were used and applied in the context of the use case.

3.1.3.1 Automatic Deployment - GeneSIS

The tool is in charge of managing and deploying the software that is running in the ITS Use Case, i.e., it is also in charge of deploying and ensuring that the software that run into the gateways are correct and deploy the access credentials to the tools that require to be integrated into the ITS infrastructure (Figure 6).

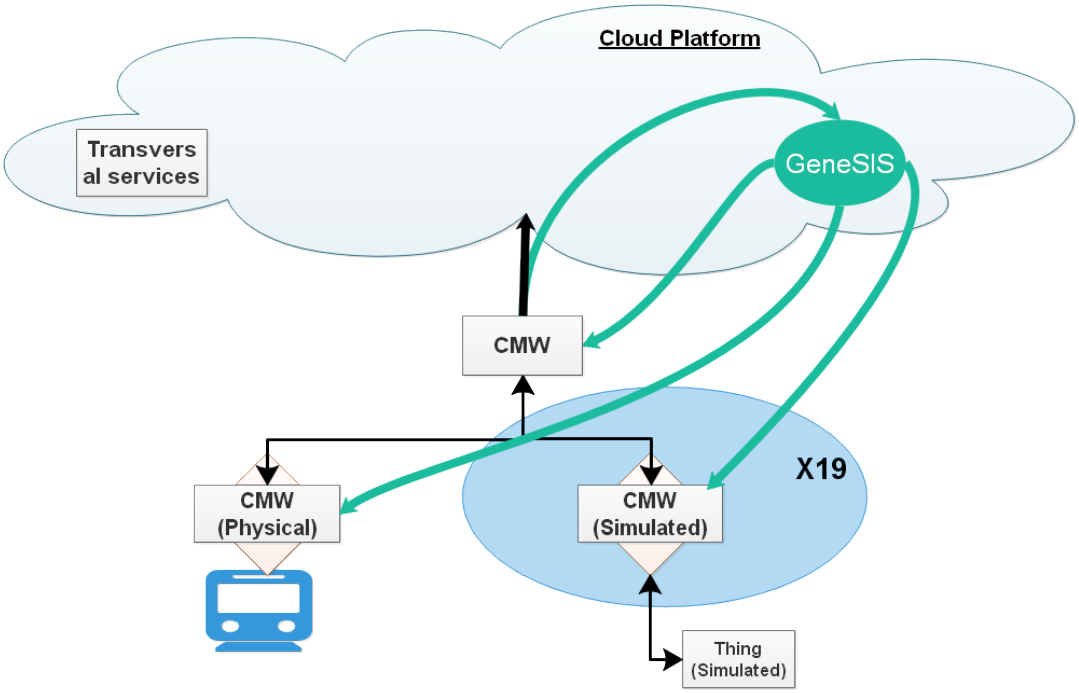


Figure 6. Orchestrated Deployment V&V Scenario

The management of the software running on the CMWs had to respect three constraints: (i) the software can be deployed or updated only when the train is in a state where running such processes is considered as safe, (ii) it should report if a deployment is successful or not, and (iii) maximum availability should be guaranteed.

For the first constraint, GeneSIS is provided with details about the status of the Use Case infrastructure to check if it is possible making a deployment or not. In case the system is running the Use Case functionalities, the deployment cannot be performed. More precisely, the following set up has been defined. On the one hand, GeneSIS is connected to the ITS infrastructure using a secure MQTT connection. GeneSIS receives messages about the state of the CMW (three states as depicted in Figure 7). On the other hand, GeneSIS is integrated as part of the DevOps pipeline and, when a new version of the software is released, its deployment is triggered. The deployment is pending until a message is received from the ITS infrastructure that it is permitted.

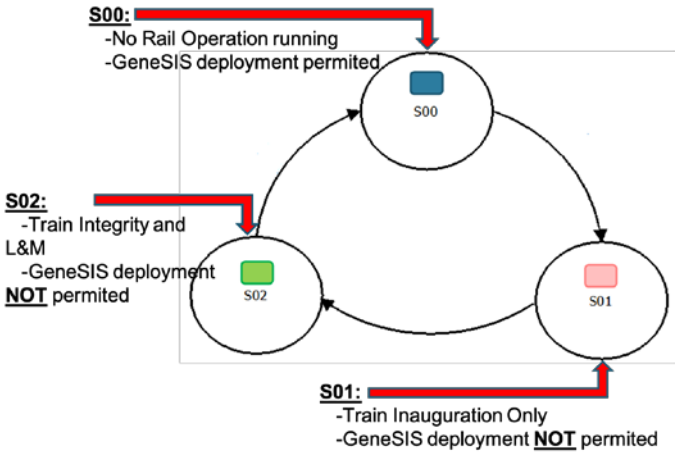


Figure 7. Main states of the CMW. Deployment is only permitted in S00.

As depicted in Figure 8, GeneSIS was used not only to deploy software on the CMWs but also to deploy the FIWARE based dashboards. The ITS-app comes as a Docker image, available in the ENACT/ITS-APP Gitlab repository (with ID registry.gitlab.com/enact/its-app:0.5.0).

For the second constraint, the tool is able to evaluate if the deployment is correctly performed and if it has a conflict with previous deployments made in the same device. As depicted in the Figure below, the GeneSIS models@runtime based engine provides users with a deployment model enriched at runtime with information about the status of a deployment. Errors and successful deployments are reported in the model and are further complemented by the GeneSIS deployment logs (i.e., report of a deployment).

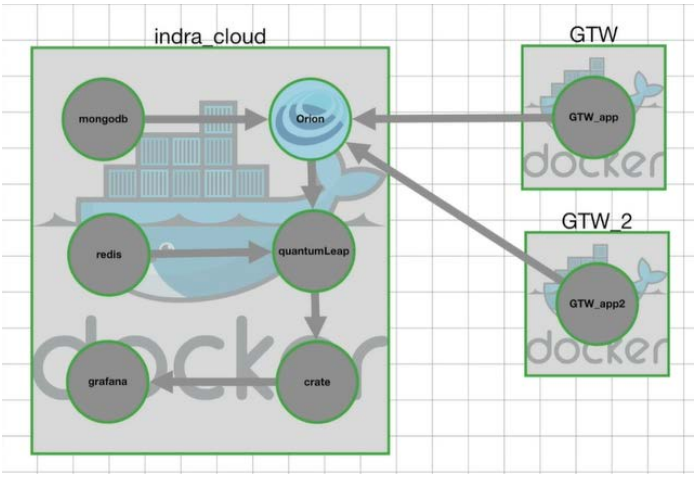


Figure 8. Deployment model, including deployment of the FIWARE-based dashboards.

For the third constraint, to ensure availability, we equip the deployment of this ITS-App with an availability policy. This availability policy let us control the number of replica (concurrent instances of the applications), the upgrade policy (i.e., when a new version is deployed), and underlying implementation. We present below an excerpt of the ITS-App deployment model detailing the availability strategy:

```
components: [
    {
        "name": "its_app",
        "properties": [],
        "availability": {
            "strategy": "Builtin",
            "healthCheck": null
            "replicaCount": 1,
```

```
            "zeroDownTime": false,
        },
    },
]
```

This policy ensures that GeneSIS deploys the ITS-App on top of Docker and uses built-in mechanisms to monitor and upgrade it (the alternative "Docker Swarm" strategy would apply as well). Because this application does not support multiple instances running in parallel on the same host, we limit the number of replicas to one, and we disable the zero downtime updates. While there is only a single instance running at a time, GeneSIS monitors its health using the given health check script, and restarts it as soon as it detects a problem. Besides, to minimize service interruption during updates, GeneSIS prepares all the needed resources (Docker image and container), before to stop the older one and to active the newer one.

Once the deployment is complete, we can open an SSH console on the ITS gateway (simulated as a virtual machine) and check that (i) GeneSIS has installed Docker in remote mode, pulled the needed images, and installed the ITS-App alongside the needed availability mechanisms, that is a NGinx proxy and a watchdog monitoring the running ITS-App.

```
$ ssh -i ./INTPBEC2SDE_SINTEF.ppk -p 2336 34.255.17.2
root@ubuntu:~# docker images
REPOSITORY                          TAG       IMAGE ID       CREATED         SIZE
registry.gitlab.com/enact/its-app   0.5.0     82be8f1422be   3 minutes ago   906MB
fchauvel/enact-nginx                latest    88c27354e4d5   3 minutes ago   153MB
```

We see that GeneSIS pulled both the ITS-App from ENACT Gitlab registry, as well as the ENACT NGinx proxy from DockerHub. We can now look at the containers running as follows:

```
root@debian:~# docker ps -a
CONTAINER ID   IMAGE                                        COMMAND
CREATED              STATUS          PORTS                              NAMES
699693af5831   registry.gitlab.com/enact/its-app:0.5.0    "/docker-entrypoint.…"
59 seconds ago       Up 58 seconds   80/tcp, 0.0.0.0:5000->5000/tcp   its_app-1
ffbbc8bfb493   fchauvel/enact-nginx:latest                "/docker-entrypoint.…"
45 seconds ago       Up 44 seconds   80/tcp, 0.0.0.0:5000->5000/tcp   its_app-proxy
```

We see that two containers are running, namely its_app-proxy, the NGinx instance, and its_app-1, the unique ITS-App. We can also connect via to the NGinx proxy and see the watchdog logs as follows:

```
root@debian:~# docker exec -it a_component-proxy /bin/bash
root@ffbbc8bfb493:/enact# ls -l
ls -l
total 44
-rw-r--r-- 1 root root   144 Feb 19 20:35 docker.sh
-rw-r--r-- 1 root root   155 Feb 19 20:36 endpoints.dat
-rw-r--r-- 1 root root 11690 Feb 16 14:13 endpoints.sh
-rw-r--r-- 1 1000 1000   302 Feb 19 20:35 healthcheck.sh
-rw-r--r-- 1 root root  6667 Feb 16 14:12 restart.sh
-rwxr-xr-x 1 root root  1827 Feb 16 08:55 watchdog.sh
-rw-r--r-- 1 root root   742 Feb 19 20:42 watchdog_its_app-1.log
root@ffbbc8bfb493:/enact# cat watchdog_its_app.log
cat watchdog_its_app.log
Fri Feb 19 20:36:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:36:01 UTC 2021 INFO: Endpoint 'its_app-1:5000' is back!
Endpoint already known.
Fri Feb 19 20:37:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:38:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:39:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:40:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:41:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:42:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
```

We can now simulate a failure of the ITS-App by stopping the container underlying the service and check that it is automatically restarted. From a second SSH connection to our host, we proceed as follows:

```
root@ubuntu:~#docker stop its_app-1
```

```
its-app-1
root@debian:~# docker ps -a
CONTAINER ID   IMAGE                                          COMMAND
CREATED         STATUS                         PORTS
NAMES
ffbbc8bfb493   fchauvel/enact-nginx:latest                    "/docker-entrypoint.…"   8
minutes ago   Up 8 minutes                  80/tcp, 0.0.0.0:5000->5000/tcp
its_app-proxy
699693af5831   registry.gitlab.com/enact/its-app:0.5.0   "/docker-entrypoint.…"   8
minutes ago   Exited (137) 40 seconds ago
its_app-1
```

Waiting about a minute for the watchdog to detect the failure, we see in the watchdog logs that it has noticed the failure and provisioned a replacement.

```
root@ffbbc8bfb493:/enact# cat watchdog_its_app-1.log
cat watchdog_a_component-1.log
Fri Feb 19 20:36:01 UTC 2021 INFO: Heath check for its_app-1:5000' returned '0'
Fri Feb 19 20:36:01 UTC 2021 INFO: Endpoint ' its_app -1:5000' is back!
Endpoint already known.
Fri Feb 19 20:37:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:38:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:39:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:40:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:41:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:42:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:43:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '0'
Fri Feb 19 20:44:01 UTC 2021 INFO: Heath check for 'its_app-1:5000' returned '1'
Fri Feb 19 20:44:01 UTC 2021 INFO: Endpoint 'its_app-1:5000' has failed!
Configuration loaded from '/enact/docker.sh':
 - Host: '192.168.1.162:2376'
 - Image: 'registry.gitlab.com/enact/its-app:0.5.0'
 - Command: "/docker-entrypoint'
 - Network: 'GeneSIS-its_app'
Container 'its_app-1' stopped.
Container 'its_app-1' removed.
New container 'its_app-1' from image 'registry.gitlab.com/enact/its-app:0.5.0'
created!
ID: "73684efd1b7f73873a0f5238bb350c4090f552cc9e19f14e0915ee044d36be53"
Container 'its_app-1' connected to 'GeneSIS-its_app'
Container 'its_app-1' started!
```

We can now update our deployment in order to deploy a new version of the ITS-App, say for instance version 0.5.1, which is also available on the ENACT Gitlab container registry. When we log onto our ITS gateway, we can see that a new container has been created but only starts once the older one has been stopped.

```
root@debian:~# docker ps -a
CONTAINER ID   IMAGE                                   COMMAND                  CREATED
STATUS         PORTS                NAMES
c4bea0c3ede2   registry.gitlab.com/enact/its-app:0.5.1   "/docker-entrypoint…"   2
seconds ago   Created              80/tcp, 0.0.0.0:5000->5000/tcp   its_app-2
73684efd1b7f   registry.gitlab.com/enact/its-app:0.5.0    "/docker-entrypoint.…"
12 minutes ago      Up 12 minutes   80/tcp, 0.0.0.0:5000->5000/tcp   its_app-1
ffbbc8bfb493   fchauvel/enact-nginx:latest                "/docker-entrypoint.…"
8 minutes ago       Up  8 minutes   80/tcp, 0.0.0.0:5000->5000/tcp   its_app-proxy
```

Waiting a few more seconds, we see that the older container gets deleted and that the new one starts as a replacement, minimizing the downtime of the ITS-App, to a few seconds.

```
root@debian:~# docker ps -a
CONTAINER ID   IMAGE                                   COMMAND                  CREATED
STATUS         PORTS                NAMES
c4bea0c3ede2   registry.gitlab.com/enact/its-app:0.5.1   "/docker-entrypoint…"   28
seconds ago   Up 4 seconds   80/tcp, 0.0.0.0:5000->5000/tcp   its_app-2
ffbbc8bfb493   fchauvel/enact-nginx:latest                "/docker-entrypoint.…"   8
minutes ago   Up 8 minutes   80/tcp, 0.0.0.0:5000->5000/tcp   its_app-proxy
```

3.1.3.2 Test and simulation (T&S)

The T&S tool has two roles into the ITS Use Case. The first one is monitoring the infrastructure in order to validate that the ITS infrastructure is running properly and the second one is simulating failures into the infrastructure to enhance the validation value against issues that may appear into the operation of the Use Case itself (Figure 9). The main benefit that this tool adds is the possibility to evaluate the ITS Gateway performance and check the information volume that it can manage. Once this information is obtained, the number of gateways to be deployed can be known by the developers and Rail Operator to optimize the deployment costs.

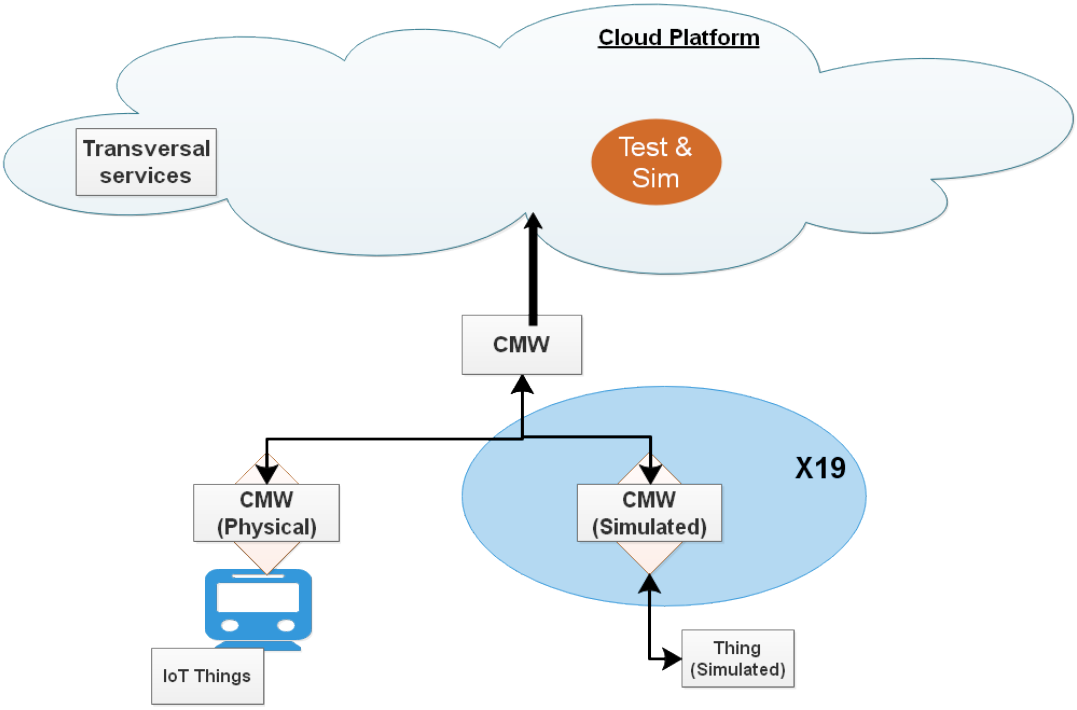


Figure 9. Test and simulation V&V Scenario

The Test and Simulation tool collects all the data gathered by the gateway layer to monitor its workload when it is operating the rail functionalities' tasks. In this case, all the traffic from the Edge and Cloud layers is monitored, in other words, the 100% of the traffic managed during an operation by the gateway.

The second role is using these data, the tool is able to replicate a single gateway, in a simulated virtual environment, taking as a basis the behaviour of the monitored gateway. Replicating the gateway as many times as desired provides the tool's user the possibility to generate a virtual scenario with all the gateways desired; hence, the scalability of a scenario can be proved. Moreover, as the virtual infrastructure is generated, the tool is able to simulate certain situations that may compromise the infrastructure and check its reliability. The scalability is tested introducing the recorded data for a single train multiplied by a scalability factor to check the information volume that the gateway can handle. The Figure 10 shows the traffic peak reached with the different scalability factors, further results will be shown in the D1.5.

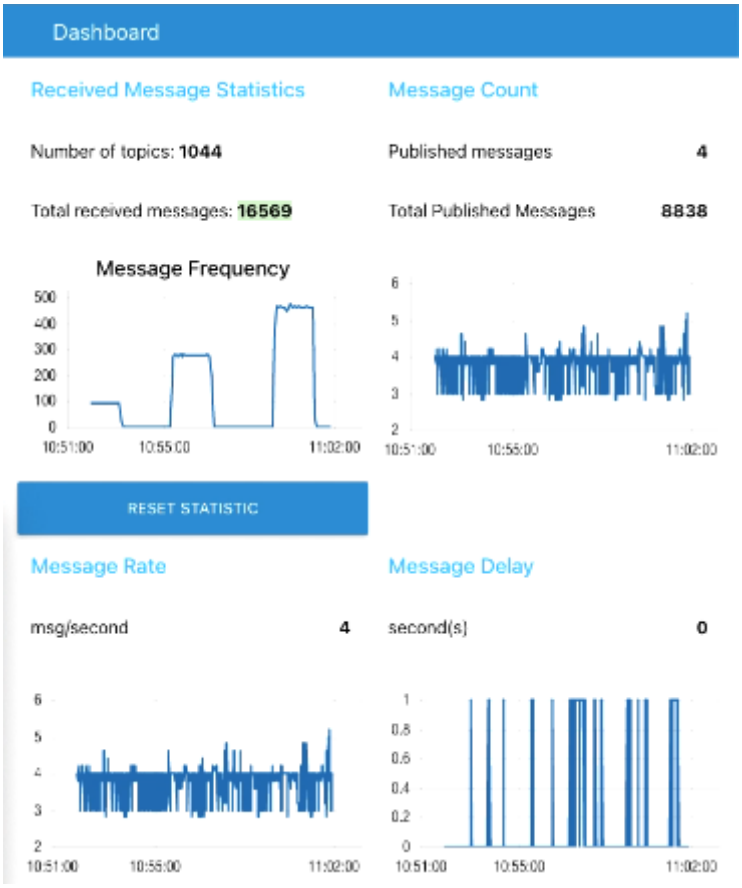


Figure 10. Test and simulation – ITS Use Case demo example

This simulated environment by its own is not enough to test the system's scalability. This infrastructure must publish several metrics to evaluate the simulated gateways are required. A specific report about the gateways status is required to know how the gateway is dealing with that monitored data. As it is mentioned in the D1.5, the gateways report specific data about their physical and routing status to evaluate their operation. Therefore, the tool is able to infer, using these reports, the behaviour of this simulated infrastructure in any situation proved.

3.1.3.3 Root Cause Analysis (RCA)

The RCA tool is in charge of checking the security perspective of the ITS infrastructure from the gateway to the cloud layer finding the reasons for a failure that may occur (Figure 11). This tool serves to identify the cause of network mistakes that in a scalable system is critical due to the quantity of devices deployed.

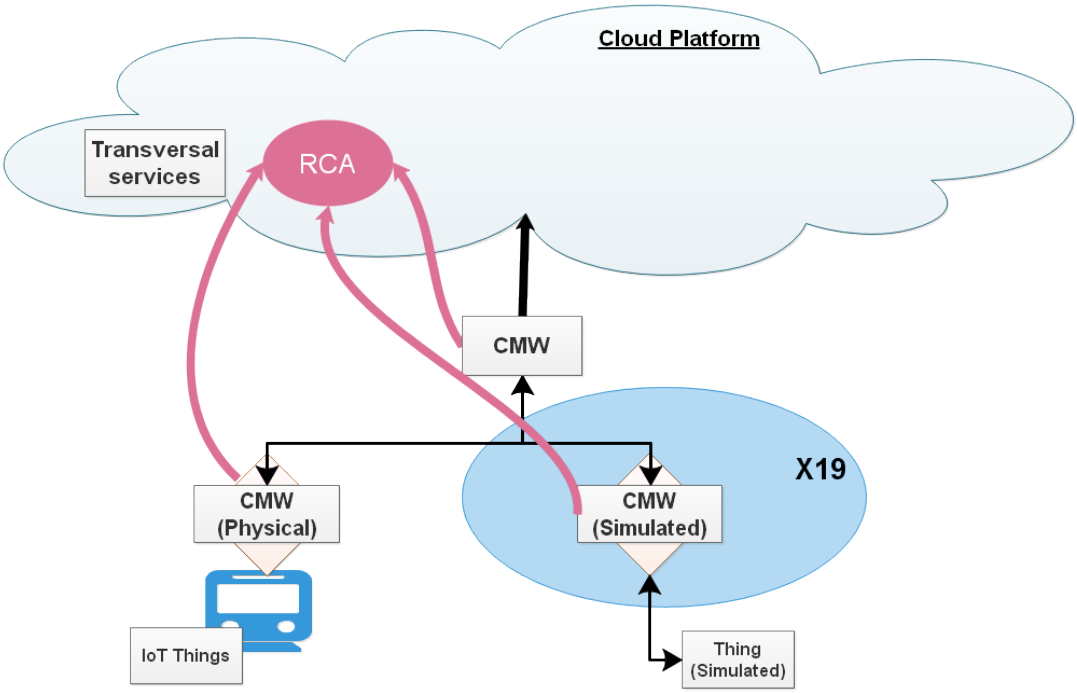


Figure 11. Root Cause Analysis V&V Scenario

Several failure scenarios are simulated and recorded by the RCA tool in order to store the behaviour of the infrastructure in case of those failures appear. The data monitored is the physical and routing data reported by the gateways.

3.1.3.4 Actuation Conflict Management (ACM)

ACM has been used to detect and manage actuation conflicts (actuation orders with conflicting goals) that may arise when information is emitted by applications to the equipment used to interact with the driver. This conflicts identification helps to the rail environment to avoid, in non-safety applications, the proper prioritization of the orders to be all executed properly.

It is worth noting that, because performing and testing actuation conflicts on a real train was not feasible, and because the hardware infrastructure developed within the project mainly focused on sensors with limited number of actuators, thereby preventing complex actuation conflict configurations to be validated, this enabler has been tested against the OpenBVE[2] advanced train simulator. Nevertheless, the interactions between the train and the enabler were performed in an "as-if" scenario - i.e., in a setup similar to what could have been performed on a real train (Figure 12).

More precisely, the following scenario was considered. Several audio information such as alarm, horn and messages can be sent to the driver. All these audio events are occurring within the same physical space and, as a result, the occurrence of one event may hinder the effectivity of another one (e.g., a horn during a message may prevent the driver from understanding the message). Several strategies can be implemented to handle such conflicts. ACM allows to introduce arbitration strategies between the information emitted by the applications and their actual transmission through end-user interaction equipment's. Several strategies can be implemented, for instance, motivated by reducing the driver's cognitive load in the event of an excessive information flow or, more simply, by maintaining consistency between the information transmitted (exclusivity of the audio message).

[2] https://openbve-project.net/

Figure 12 depicts the implement of one such strategy, which consists in giving priorities to the interaction modalities (alarm > horn > message).

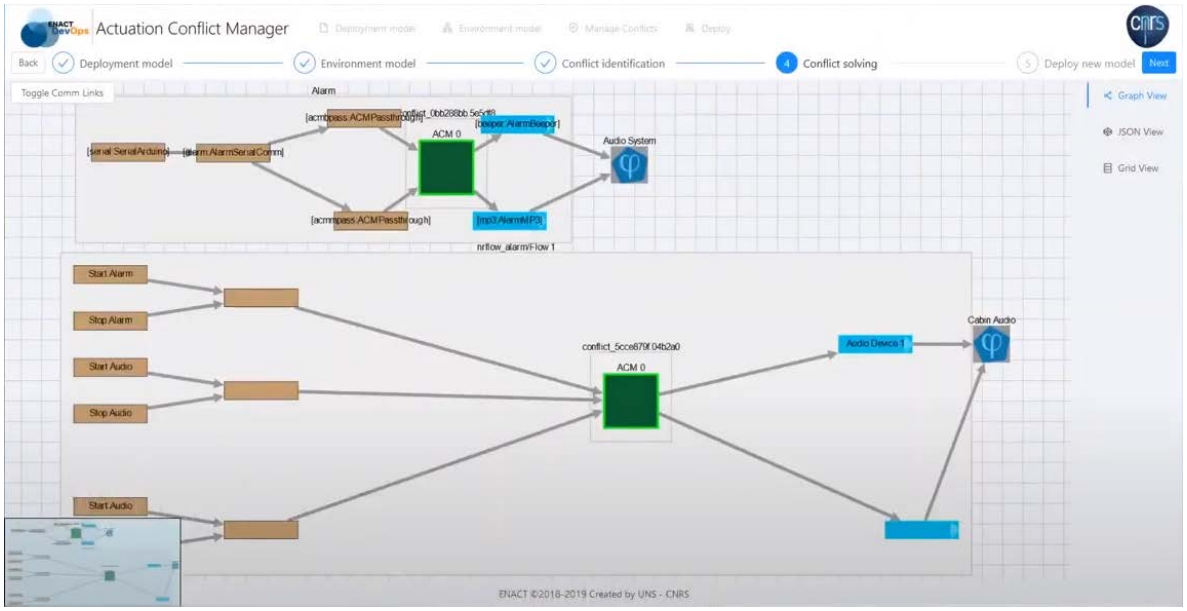


Figure 12. ACM - Rail Use Case Integration synoptic.

3.1.3.5 Behavioural Drift Analysis (BDA)

BDA is in charge of monitoring and assessing SIS concrete behaviour. The tool is fed with business data published in real time by devices deployed at the edge of the infrastructure (Figure 13). On the basis of this data and a model of the SIS legitimate behaviour, the BDA tool computes a score which, when less than one, denotes a behavioural drift and, when equal to zero, denotes an unexpected and potentially harmful behaviour.

The BDA tool integration has been validated with EDI on a use-case where data are gathered from a gateway attached to a model train equipped with accelerometer sensors. This approach was motivated by the impossibility of making the behaviour of a real train to drift because of the obvious risks this entails for both humans and equipment.

For the same reasons, a train simulator has been used to assess the BDA tool on more complex yet realistic scenarios. The train simulator is OpenBVE. This tool focuses on realism by simulating the brake system, friction, air resistance, acceleration, etc. It also provides real-time status of some actuators (e.g. speed, doors opening, etc.) and the traffic signs along the track. This makes the tool particularly relevant for the purpose of validating the BDA tool. On the basis of this tool, two legitimate behavioural models have been developed, monitoring and assessing train and passenger integrity. The first model (shown in Figure 14) defines the doors opening states the train must comply with depending on the context (e.g., train moving, stopped in a train station,...). The second model (shown in the Figure 15) defines the ranges of speed permitted for the train, according to the speed limits given by the traffic signs along the track. Several scenarios were played using the OpenBVE simulator and drift could be observed. Examples of drift can be seen in the following videos: https://youtu.be/mokA22r20a4

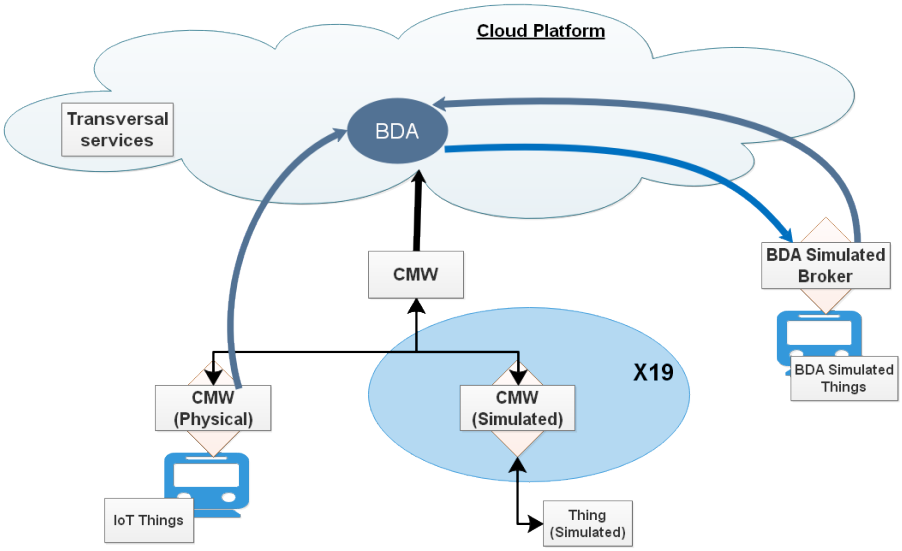


Figure 13. Behaviour Drift Analysis V&V Scenario

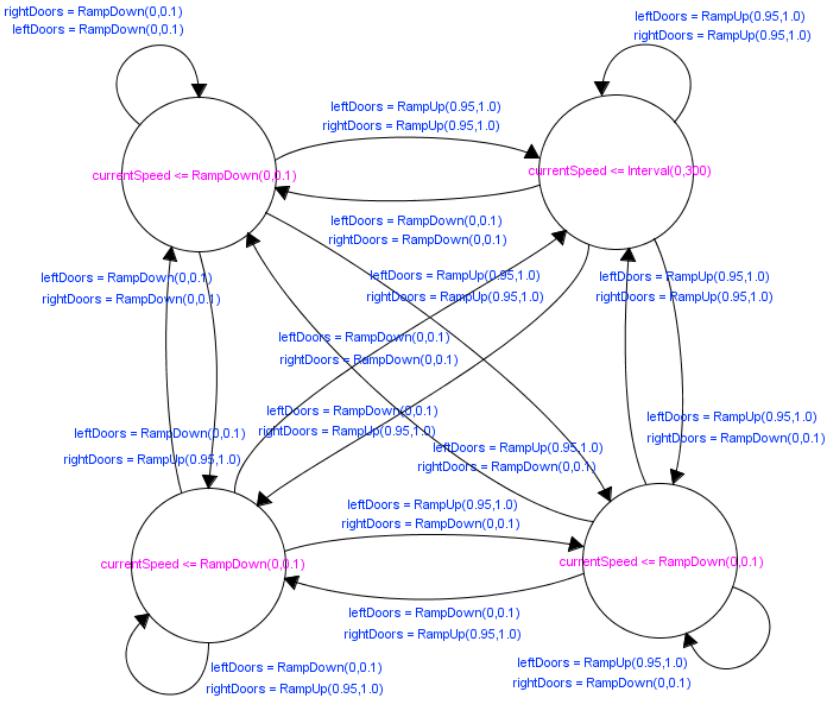


Figure 14: Train door behavioural drift analysis model

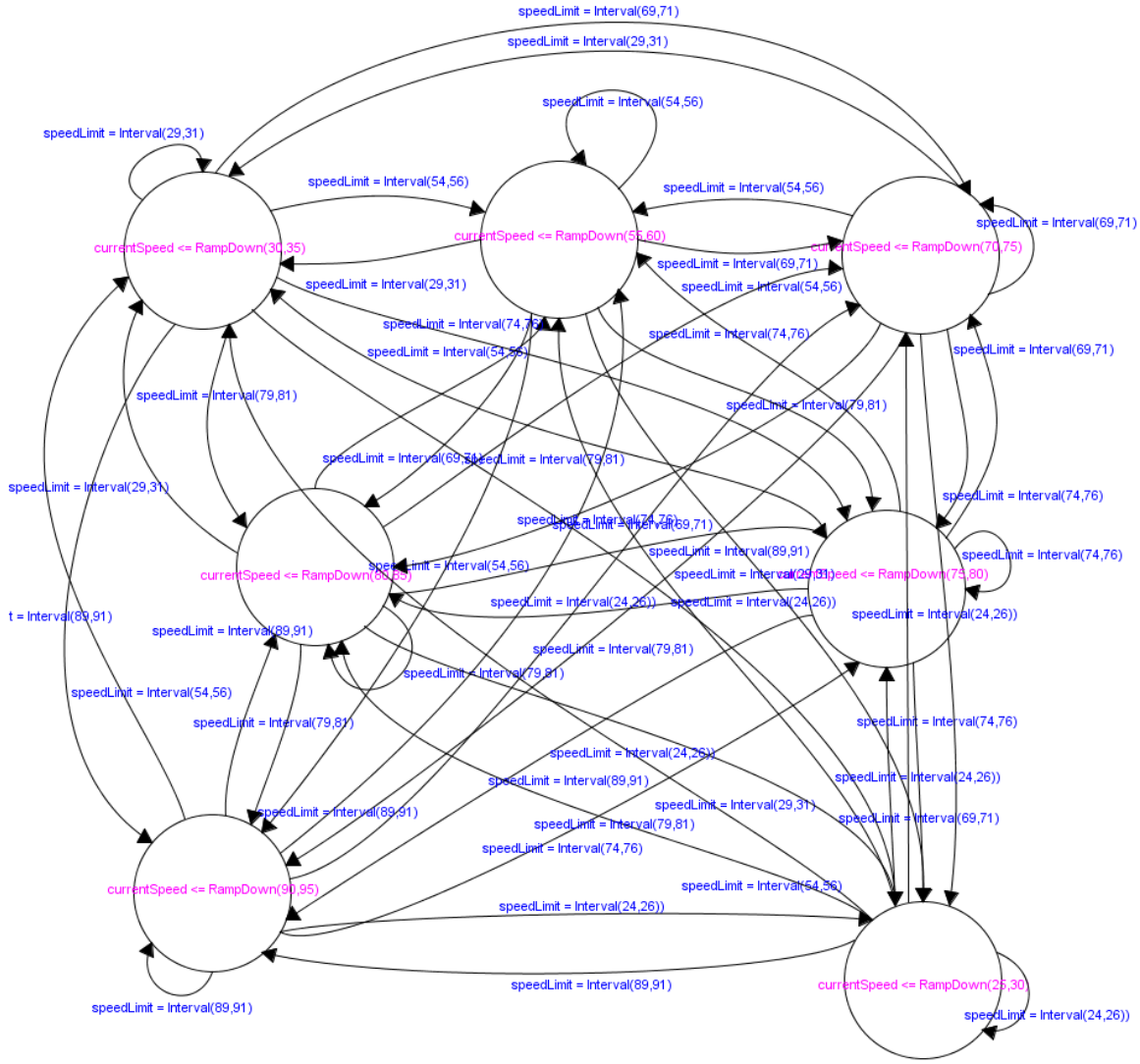


Figure 15: Speed limit behavioural drift analysis model

3.1.3.6 Security and Privacy Monitoring and Control (S&P)

The S&P tool is in charge of providing the security solution that is required at the edge as IoT technologies can suffer vulnerabilities at this layer.

Since there is no private data transmitted in the on-board train system (nor in the rail track system), the main purpose of using this tool is to supervise the security of the communications on board so as the train data is protected from confidentiality, availability and integrity issues. The tool is fully integrated with the edge (Gateway) in the train and therefore it is used to detect any security incident or anomaly in the on board network traffic, as well as to raise alarms and enact controls in these communications as reactive mechanisms.

The S&P Mon&Control tool integration follows a different approach compared to that of other ENACT Enablers, since in order to cover the security at the edge and to ensure it does not get compromised, the tool needs to be as independent of the system it is being monitored as possible. For this reason, the tool backend is not integrated within the Indra Hybrid Cloud Platform, and resides in an independent Cloud in Tecnalia premises.

The tool offers two main services for the use case, the security monitoring and the protection or control services. The monitoring service is in charge of collecting the data that are transmitted through the On-Board gateways and detecting any issue or anomaly there. This data includes the Things and the DMI

data only. These data are captured by the S&P Mon&Control sensor agent deployed in a dedicated hardware device installed on board, and sent to the S&P Mon&Control tool back end in the Cloud to be analysed (Figure 16).

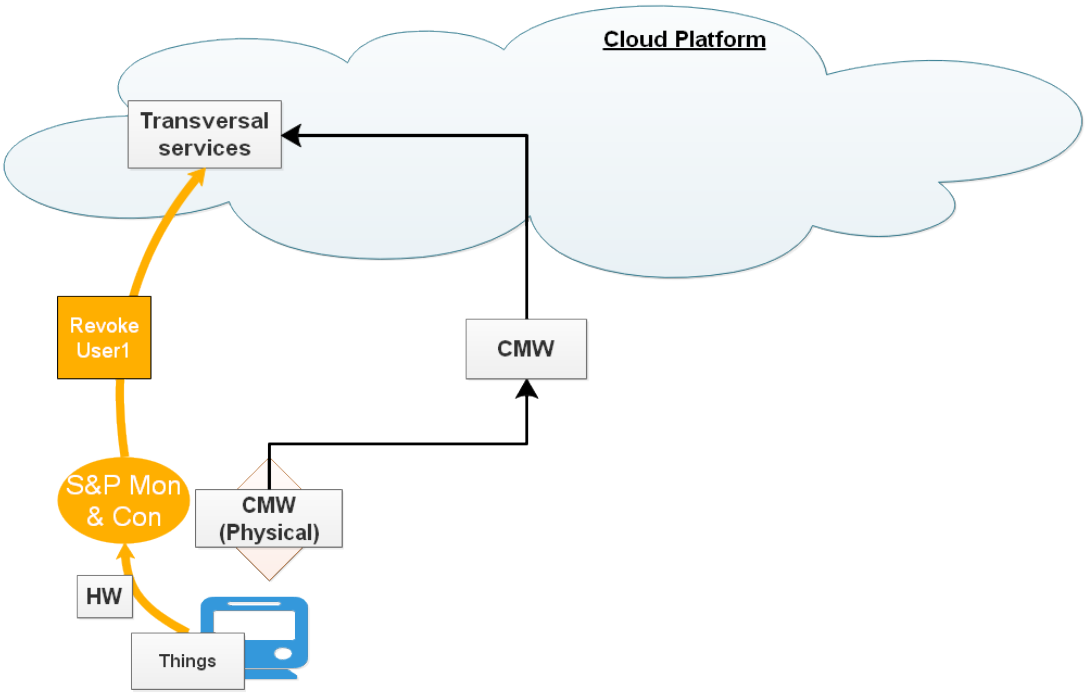
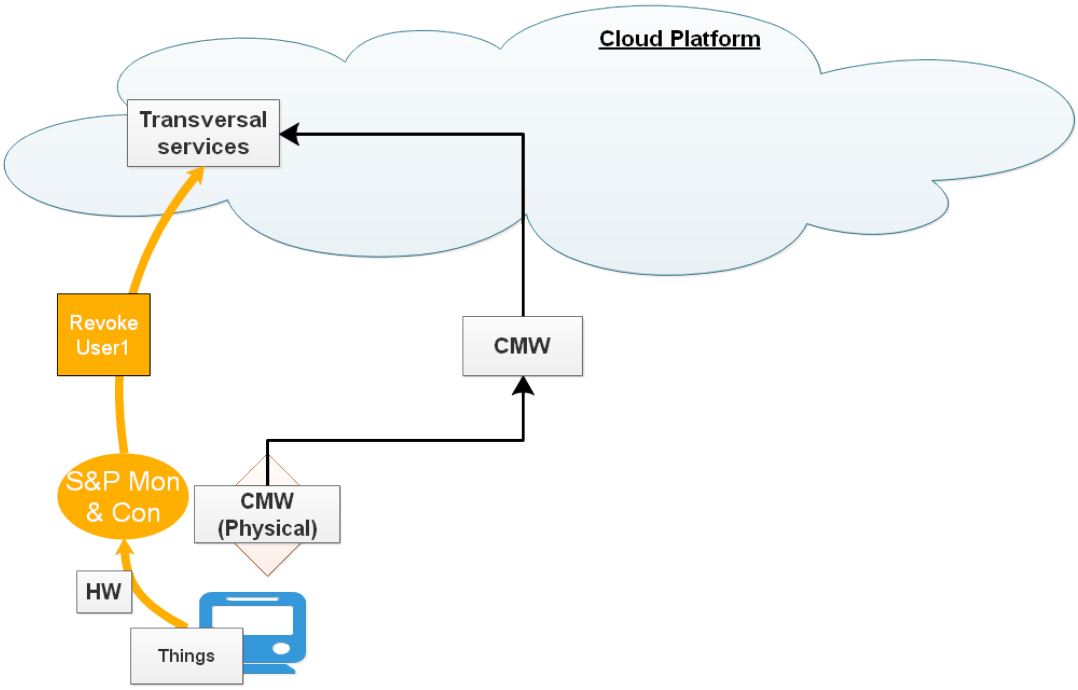


Figure 16. Security and Monitoring V&V Scenario

The S&P Mon&Control tool is integrated in the train on board scenario with twofold objective:

- Traffic behaviour monitoring and reaction: The edge layer traffic behaviour is considered as one of the characteristics that defines the rail performance is regular. The safety and security requirements that define this kind of functionalities require a regular traffic, with specific data packet types being transmitted at defined intervals, with defined frequencies and alternations among them. This normal traffic may be affected by either incidents or intruders attacking the edge. The S&P Mon&Control monitoring is employed to detect any deviation in the regular traffic and the control service is used to revoke the user, from the Use Case central authentication servers, that generates such deviations.

- Intrusion detection and control: By supervising all the traffic of the edge layer, the S&P Mon&Control tool can learn which user is connected at any time and verify whether it corresponds to the permission granted users, which are those systems that can publish/subscribe to the IoT Platform and are therefore authenticated in the central Use Case authentication server. These users need to be registered also in the S&P Mon&Control tool



in the central Use Case authentication server, but not in the S&P Mon&Control tool registry, the control service in the S&P Mon&Control tool will react by revoking that user or/and MAC in the Use Case central authentication server, and thus, its connectivity to the whole IoT Platform will be stopped.

3.1.4 Software

These pieces of software have been developed in the first and second periods of the project and as summarized in the following list:

- **Background:** First period developments to tackle the ITS Use Case needs.
  - Gateways developments
  - Cloud development
  - Things development
- **Software Improvements:** Second period developments to tackle the proper integration of the ENACT tools.
  - ITS Virtual Gateway Infrastructure
  - S&P User Revoke Software
  - Gateway Status and Operational Stage Provision
  - Routing Topologies

All this code for the gateways, including the extension for the Cloud, the Things, and the improvements is site are in the following repositories:

- Gateways/Cloud and their related adaptations:

https://bitbucket.mova.indra.es/scm/iotsafety/its_enact_public.git

- Things:

http://git.edi.lv/enact-project

Further descriptions and evaluation of the components defined into these repositories are stated in the D1.5.

3.1.5 Outlook and further work

The future of the railway market involves digitalization, automation, connectivity and the use of intelligent systems that continue to add value to society by improving management, operation and user experience, in order to face the challenge posed by the European Green Deal as evidenced in the different Strategic and Innovation European agendas, as well as in the plans and reports of public governs and relevant public organizations and Programs (such as the Shift2Rail European Innovation Initiative, the Innovation Plan for Transport and Infrastructures launched by the Government of Spain, the French-Swedish Strategic Partnership for innovation and green solutions in the transport sector, etc)

The digitalization of the railway market as well as the automation and deployment of new intelligent systems, requires the design and implementation of new systems to be deployed in the railway ecosystem; systems that will pose a technological challenge to achieve the objectives set in the European agendas for the coming years, and that will entail major changes as well as the deployment of a large number of devices and subsystems both On-Board and On-Track. These systems and devices, key to a progressive and changing digitalization of the sector, must be prepared for agile developments and deployments, where their control and monitoring provide the necessary mechanisms to guarantee an efficient, robust, safe, secure and updated operation, according to the demands and challenges to be met. The ENACT results emerge as a facilitator to meet the proposed challenge, in line with the strategic lines and programs described for the future digitalization of the rail market providing DevOps enablers.

The ITS platform defined into the ENACT project framework serves as an example to enable further integration with the DevOps cycle and remarks the fact that it is actually possible. For the ENACT project, the Train Integrity and Logistic and Maintenance services have been brought only; however, several more rail services can be included into this philosophy.

3.2 Digital Health

Demographic trends as well as the increasing incidence of chronic diseases are already challenging the sustainability of existing health systems worldwide, this has escalated dramatically by the ongoing pandemic in a way that has not been encountered before. In this context, digitalisation, including remote monitoring and tele medicine, is seen as key mean to resolve these challenges and as a key offering for healthcare providers to reduce the load from chronic diseases and for monitoring elderly living at home. From the societal perspective, people want to live their everyday life at home as long as possible, also when they are suffering from health issues and reduced capabilities. The Digital Health use case represents an innovative solution in this domain as it leverages an integrated system that can support people with a variety of health issues and are meant to be easily evolved as the patient needs evolve.

A summary of the main achievements and impact of the Digital Health use case both in terms of business level achievements and technical and validation achievements are summarized in the following:

- The use case itself has evolved from being a prototype development to being at production ready system recently sold to customers and already deployed for remote supervision of several hundred patients suffering from the following chronical diseases: COPD, Kidney and Diabetes. Moreover, it has been applied for remote supervision and following up of Corona patients in several municipalities in Norway. The application has got attention from the World Health Organisation (WHO) and will be presented at a WHO conference this spring by one of our customers (a Norwegian municipality). This again leads to attention from Norwegian media houses putting this on their news (including TV news). The support and exploration we have been able to do in ENACT has been significant to reach this success, in particular, the Personal Health Gateway, which is a core component managing the IoT and edge part of our service, the ENACT project has been significant for its development.

- During ENACT TellU has experienced significant growth, both related to exploiting the ENACT use case to build a production ready remote patient monitoring service and by evolving its already existing remote supervision services in the welfare domain (camera-based and sensor based supervision), which evolution has also been explored as part of ENACT. In figures TellU has grown its revenue 10 times during the ENACT project period (from 0.7 M€ in 2017 to 7 M€ in 2020) and has grown from 5 employees in 2017 to about 40 to date.
- TellU has or are investigating to exploit several of the ENACT enablers. In particular:
    - TellU is already exploiting the ThingML part of the ENACT Orchestration and Continuous Deployment Enabler in its DevOps process in particular, for the efficient development and management of code deployed on edge and IoT devices. Moreover, the ENACT GeneSIS tool is explored to support the continuous deployment on the edge and IoT level.
    - We have started to exploit the DivEnact framework (which we consider as part of the ENACT orchestration and continuous deployment bundle) that supports the management of large number of largely distributed but similar edge and IoT deployments. TellU foresees to evolve our exploitation of DivEnact further to be able to better manage our large-scale deployments of services we deploy in people's homes related to our Telecare services. These services need to be efficiently and securely managed across the IoT, edge and cloud space. This is planned and prepared and is also further funded by a new innovation project supported by the Norwegian Research Council.
    - In our use case we have evaluated the Context Aware Access Control (CAAC) tool, which is part of the ENACT Security and Privacy Monitoring and Control Enabler. In particular this provides interesting opportunities in terms of, for example, providing access to IoT and edge devices such as surveillance cameras related to various context. For example, in case of a fire alarm it may be an opportunity to provide access to the firefighters, while in the normal state the access is purely granted to authorised health personnel.
    - As part of our use case we have explored the Risk Driven Decision Support enabler to evaluate its capability to support our risk analysis and risk management procedures related to our services, and to facilitate the reporting for ISO 27001 compliance.

3.2.1 Overall Application Description

The industrial-based use-case from TellU that was developed in ENACT is a Digital health system for supporting and helping various patients staying at home to the extent possible during treatment and care. One type of "patients" is elderly people, where the Digital health system will feature elderly care to allow the elderly to live at home as long as possible. Another type of patients that we approach with our services is patients with chronical diseases such as Diabetes, Kidney, COPD and patients with temporary diseases that need to be regularly followed up such as Covid-19 and cancer. These need to be followed up also in terms of their medical condition related to their disease. For example, Diabetes patients that need to follow their glucose level and regularly provide measurements and questionnaire reports that will be regularly followed up by health personnel. A general set up of a TellU eHealth use case for medical telecare services are illustrated in Figure 17.

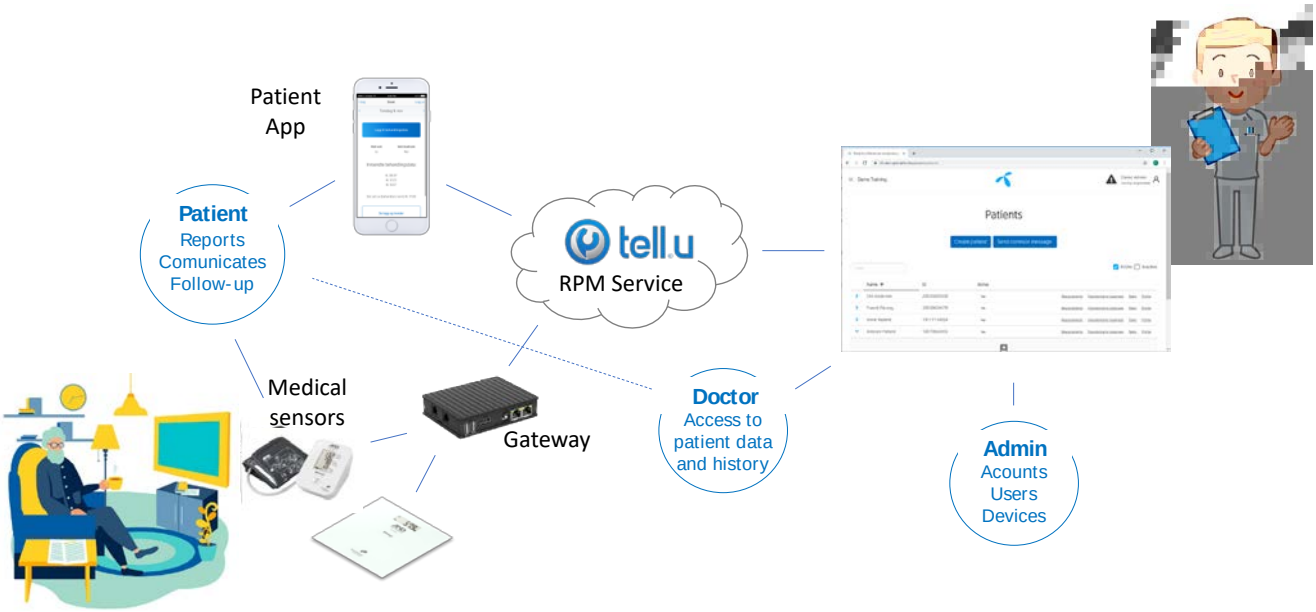


Figure 17: Digital Health applications.

The digital health system both control sensors that are deployed for remote supervision such as bed sensors, motion sensors, sensors for indoor and out-door location, video based supervision etc, and control various types of medical devices and specific sensors supporting the care and wellness for the specific patient (e.g., blood pressure meter, scales, glucose meter, medicine reminder). In addition, the system needs to integrate with other systems, for instance to provide information or alarms to response centres, care-givers, physicians, family etc., to feed information to medical systems such as electronic health record systems etc.

3.2.2 Architecture and main scenarios of final implementation

In terms of managing distribution and the IoT and edge space, the central component is what we denote "the personal health gateway", which integrates the sensors and devices as well as controls the edge and ensures the right data are provided to the various stakeholders and to the cloud-based system. Thus, the handling of large numbers of largely distributed personal health gateways and their connected sensors has been a main focus in this case study for the validation and exploitation of the ENACT technologies. In particular, we have explored the potential in IoT, edge and cloud services, by having smooth integration of heterogeneous devices, DevOps process for the development of the edge components, as well as secure and trustworthy connections and data transfers.

This Use Case is set up with a local/edge infrastructure consisting of a set of devices and a home gateway. A set of such local infrastructures are then connected and aggregated into a cloud-based infrastructure. The overall architecture of use case is depicted in Figure 18.

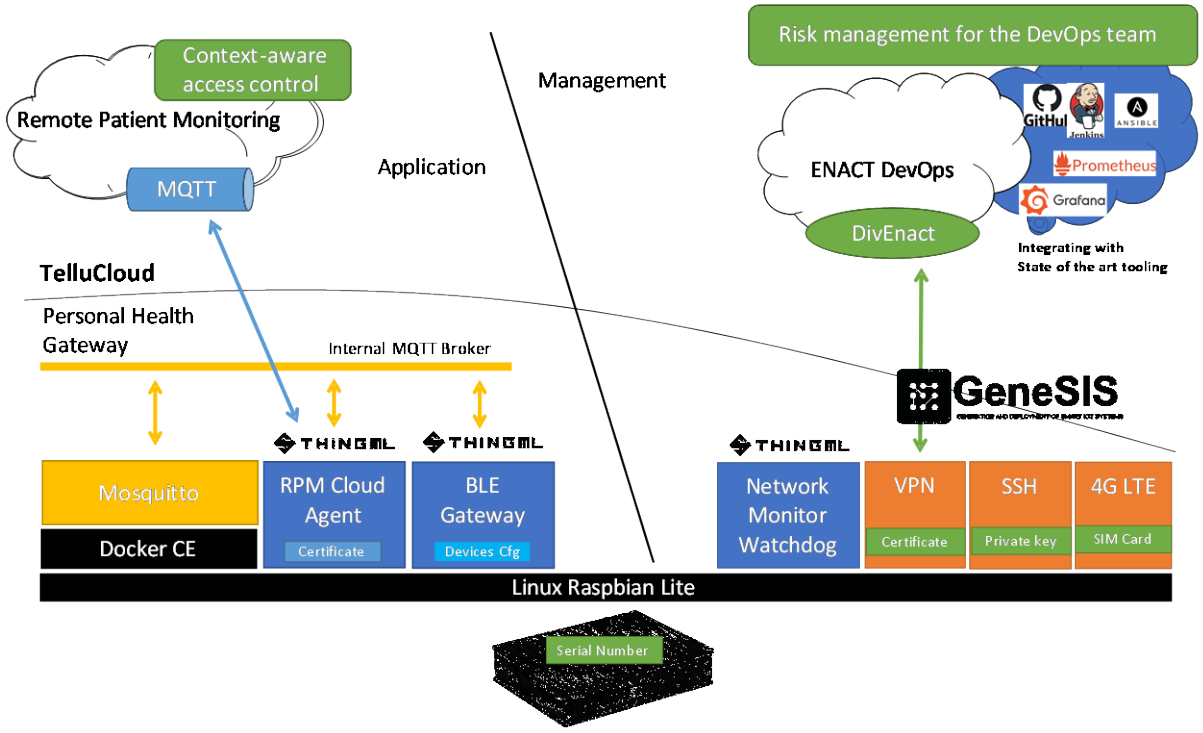


Figure 18: Overall architecture with the GW architectural components.

The Personal Health gateway architecture is depicted in the lower part of the figure, and is the part that is controlling the edge and connecting devices and sensors in the IoT and edge space while the TelluCloud eHealth system resides in the cloud. The total system encompasses a complex ecosystem spanning across IoT, edge and cloud. The Personal Health Gateway consists of a set of microservices to manage the various interactions with the devices and cloud services. The service interaction at the application and management levels are completely separated from each other. This is partly to ensure strict security and privacy requirements. The BLE gateway component manages Bluetooth Low Energy (BLE) enabled devices such as blood pressure meter, scale, glucose meter etc. The RPM cloud agent includes the application logic that resides on the edge level and interacts with the cloud-level service. A set of microservices support the management and DevOps process providing access to system level operations of the Personal Health Gateway through secure channels. Moreover, it includes the monitoring component providing system and application-level monitoring required for the continuous operation of the service.

The Gateway includes an MQTT broker and support for standard internet communication protocols. Most components run on docker containers.

The application of the ENACT enablers are indicated in the overall architecture figure:

- The Context Aware Access Control is explored to provide more advanced application-level functionality in order to dynamically provide access to different stakeholders based on the context. This can be an escalated situation/crisis situation, for example, in case of a fire alarm it may be important to provide access to the camera applied in remote camera based supervision for the firefighters, while in the normal state the camera will only be possible to be accessed by authorised health personnel.
- ThingML is fully exploited for the efficient coding and DevOps support of the Personal Health Gateway.

- GeneSIS together with DivEnact are explored for the efficient management and continuous deployment of potentially large-scale deployments of our telecare service where the IoT and edge devices are managed through the deployed Personal Health Gateways (PHG) residing in people's homes. Note that the PHG is the software stack as depicted in the overall architecture figure above i.e., the software managed in the DevOps process. In general, it can be deployed on various computing devices such as a stationary GW or on mobile gateways (e.g., smart phones). The most recent release of our PHG can be deployed on Android and iOS based smart phones, enabling the patient to do medical measurements on travel. For this, we need efficient and automatic management of variants and versions of the software stack, which is challenges that are addressed by DivEnact.
- The ENACT Risk Driven Decision Support tool is explored as part of our DevOps process that needs to be compliant with standards such as ISO 27001, where risk analysis and risk management is required to be an integral part of the DevOps process.

3.2.2.1 Scenarios

Part of TelluCloud's competitive edge is to be flexible – i.e., to be able to customize a deployment to the needs of individual users, integrating the specific sensor devices, gateways, etc., which are relevant to the user needs and setting up tailored service logic. The basic premise of the ENACT use case is that a primary user has a need for some form of medical or personal monitoring in their daily life living at home. TellU does business with care providers, producers of devices and providers of relevant services to deliver complete digital health services. Moreover, a central challenge related to the case study is the privacy and security of data. The use case potentially involves various forms of sensitive data, and TellU exploitation in real business cases will manage sensitive data that requires proper treatment. These can include personal sensitive information registered in the system, indoor or outdoor location of persons, alarms, and medical measurements such as blood pressure and blood glucose levels. These sensitive data, and the system need to both adhere to governmental rules and regulations and have the full trust of the users. The General Data Protection Regulation (GDPR) EU regulation needs to be adhered to. TellU role is normally to act as a Data Processor in GDPR terms, implying that information security and protection and proper treatment of sensitive data needs to be in place. This implies to ensure proper risk management throughout the DevOps process and to have evidence and traceability of the risk management as part of the DevOps process.

In the following table, a summary of some main requirements and scenarios that have been explored by the Digital Health Use Case is presented.

No	Overall Requirements and Scenarios	ENACT enablers
S1	Information security (e.g., according to ISO 27001) needs to be preserved in TellU eHealth systems. Moreover, a longer term need is to have certified components and applications to be able to deliver not only data transfer but process data according to requirements for Medical Device standards in the digital health domain (e.g., according to International Organization for Standardization (ISO) 13485). In particular, it is a goal to prepare a certified Personal Health Gateway. To accomplish certification, a risk driven, and formal software development process is required as part of an overall quality management system. This includes proper risk management that includes evidence of risk mitigation and traceability of risk	Risk Driven Decision Support enabler, ENACT Risk Management Tool

	management in the DevOps process needs to be in place. To accomplish this in an efficient way, automation of risk management should as far as possible be inherent in the DevOps, for example by integrating security, privacy and data protection engineering functionalities into existent, mainstream DevOps tools. Moreover, a model-based approach and generative techniques as promoted by ENACT is a way to better document and test according to Medical Device standards. Trustworthiness in terms of security, privacy and safety is key requirements in the digital health domain. To further strengthen trust and robustness we see agile and DevOps support as key, for example, in order to quickly respond to unexpected behaviours.	
S2	Related to deployment of code in a secured and reliable way, the digital health use case has elaborated the ENACT DevOps framework to perform the efficient and safe deployment of code on devices, the Personal Health Gateway and cloud resources, and ensure the safe interoperation of the distributed software pieces, where the various pieces of software need to be deployed in a distributed fashion across the IoT, edge and cloud infrastructure. This includes the handling of main scenarios supported by GeneSIS: • "Initial deployment". deploys the system following the availability tactics selected by the user. • "Internal Fault". A fault occurs in the system and it has to be dealt with preferably making it not visible to the end-user. • "Zero-downtime upgrade", the user requests the deployment of a new version of the system and this should be leveraged to minimize service disruption.	ENACT Orchestration and Continuous Deployment Enabler, in particular ThingML, GeneSIS integrated with state-of-the-art DevOps tools (Ansible, NodeRed)
S3	The Fleet management of a large number of Personal Health Gateways and their set of connected devices that typically can vary from patient to patient needs to be tackled in a DevOps lifecycle. Managing variations in the software stack and specific configuration also needs to be in place, and there is a need to properly cope with the complete system across IoT, edge and the backend system in the cloud.	ENACT Orchestration and Continuous Deployment Enabler, in particular DivEnact, ThingML and GeneSIS
S4	Allow dynamic access according to the actual context in a controlled manner for example providing access to IoT and edge devices such as surveillance cameras related to various context. For example, in case of a fire alarm it may be an opportunity to provide access to the firefighters, while in the normal state the access is purely granted to authorised health personnel	ENACT Security and Privacy Monitoring and Control Enabler in particular the Context Aware Access Control tool

S5	Testing systems that is distributed across the IoT, edge and Cloud space is particularly challenging as the production environment is typically not at hand as the largely distributed nature and the combination of hardware and software components are normally not possible, or very costly, to set up in a realistic manner. This is also the case for the Digital Health use case as it is largely distributed and the environment in people's homes are hard to test in a physical test environment. Thus, in order to test such systems, simulation is key, and the scenario is to properly simulate this kind of IoT based systems in a DevOps setting.	ENACT Test and Simulation Tool

3.2.3 Application of the ENACT enablers

In this section we present the main applications of ENACT enablers in the Digital Health use case.

3.2.3.1 ENACT Risk Management

In the following we present application of the Risk Management enabler applied for the Digital Health use case. In particular, we have applied it for risk analysis and management of our eHealth platform. An extract of the overall operational architecture of the eHealth platform is illustrated in Figure 19.

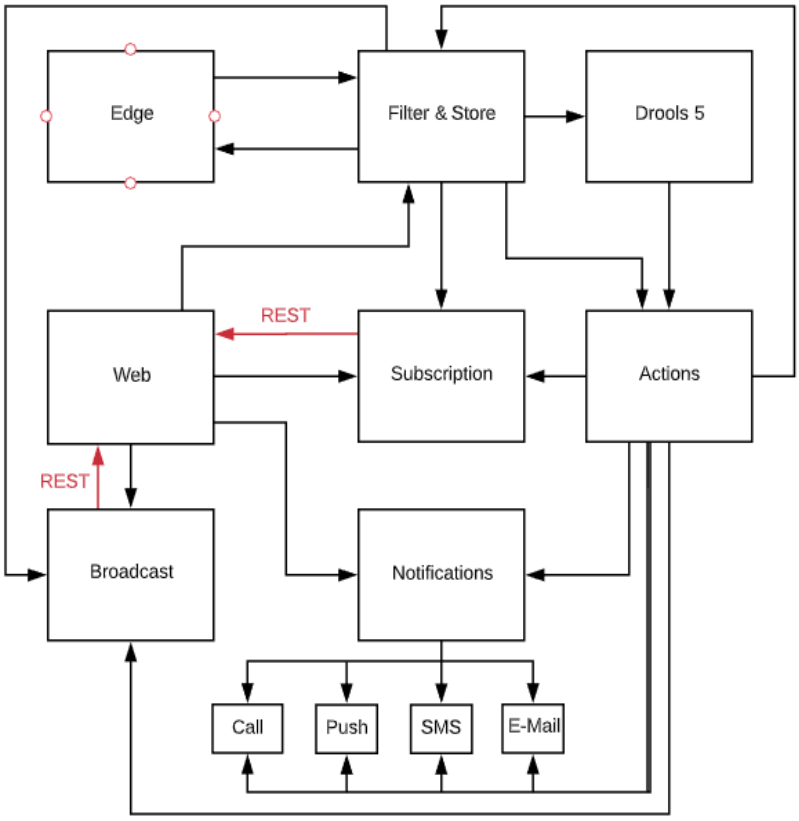


Figure 19: eHealth platform Architecture

The eHealth platform enables to build structures to allow to catalogue and store data from IoT-devices and to perform actions based on data coming from these devices.

The Edge represents bidirectional communication with the devices, and the Web is an interface for managing and reading data. The Drools based rule engine processes the data and determines whether or

not any actions should be triggered based on the data. These actions can be creating an alarm, notifying a user, creating an external event, updating a state etc.

The platform provides four channels for communicating with users: Push Notifications, Short Message Service (SMS), E-Mail and Calls, with SMS.

Below is an extract of applying the ENACT risk management tool related to the risk management of the SMS component of the eHealth platform.

Examples of Risks included for the SMS component are:

- Incorrect configurations being pushed to the component
- Third party SMS Provider is down, or not responding
- A Customers daily or monthly SMS threshold limit is exceeded

A risk model for related to a component perspective is illustrated in Figure 20:

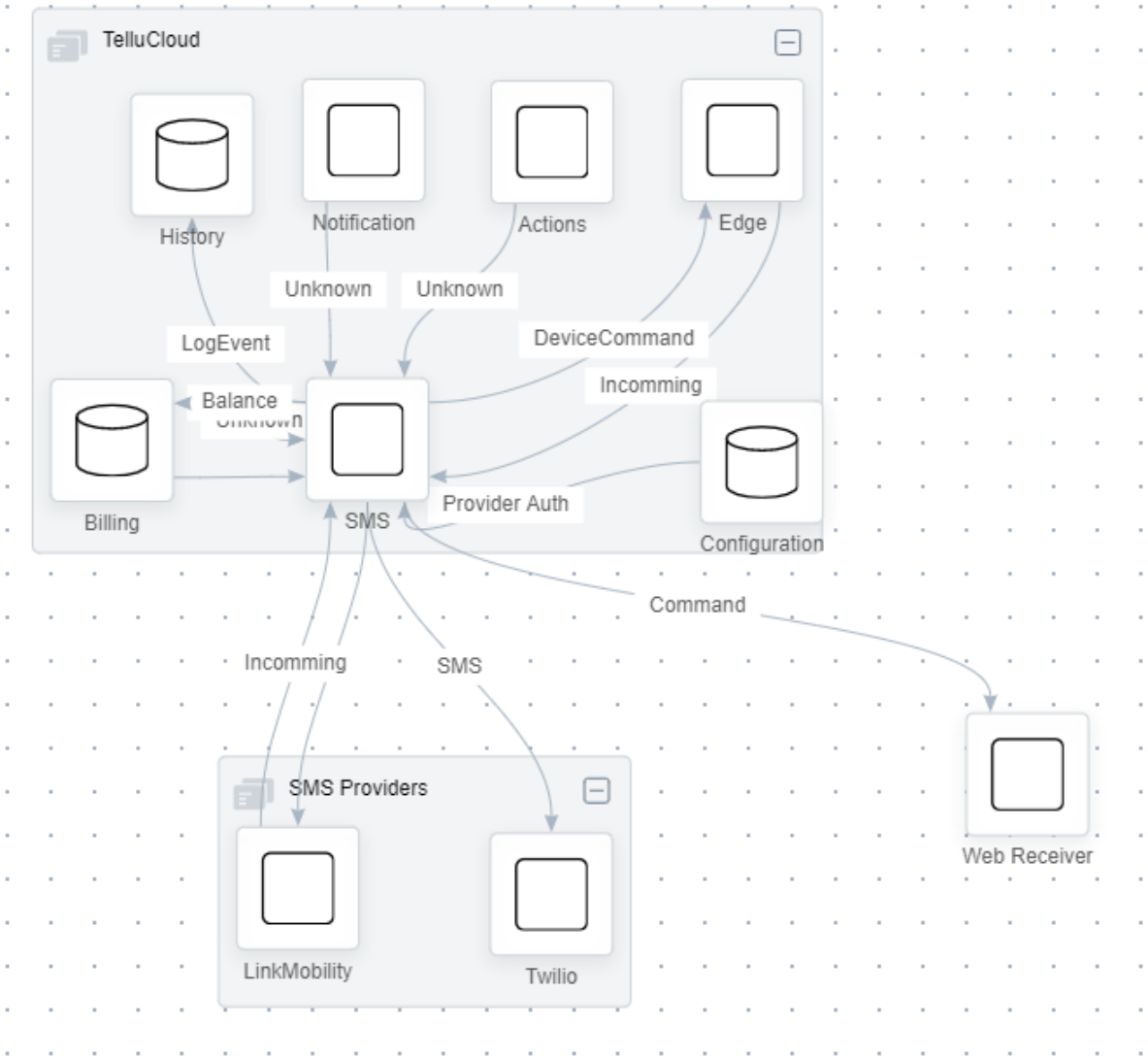


Figure 20: Risk model for SMS component Application View

The figure shows the SMS component, and how it fits in to the system and the approach is to analyse risks per component. For example, a vulnerability for the component configuration is that a corrupted configuration is deployed to the SMS component, which may render the component in a state where it is not able to send SMS's at all.

Another perspective that should be analysed for a component are risks related to the process where the component takes part. E.g., there is a threshold of max SMS that is meant to prevent the SMS system of going out of control, however, this can prevent a customer of getting important SMS's delivered that should be delivered (if the threshold is reached). The scenario is that a threshold limit set for customers, indicating maximum SMS's per day and month, has been exceeded. This can be modelled in a Data Flow Diagram in the ENACT risk management tool as illustrated in Figure 21:

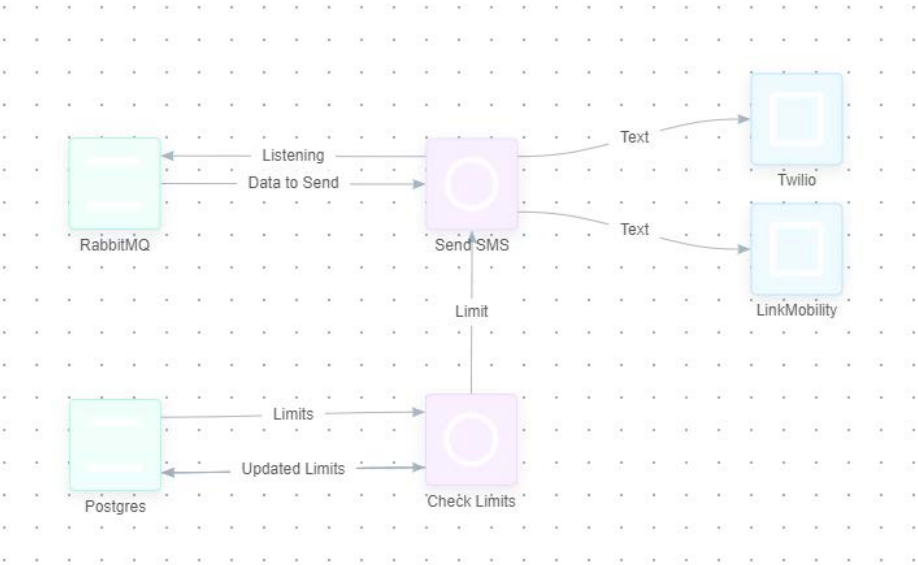


Figure 21: Data Flow Diagram for SMS

The treatment for this use case is to have an hourly database check, whether the thresholds have been surpassed, and notify developers if they have as illustrated in the ENACT risk management tool below (Figure 22):

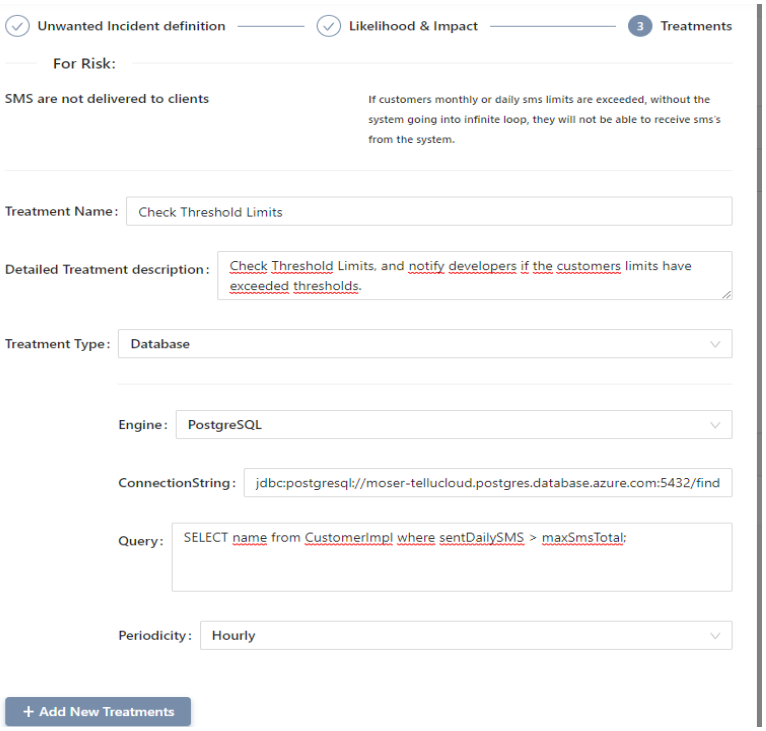


Figure 22: Risk Editor Treatments

Treatments may also be connected to Repositories or Issue Managing tools such as Jira.

The ENACT Risk management tool provides functionality to generate risks and vulnerabilities after providing an analysis.

For example, for our Postgres database in the context of SMS as depicted in Figure 23:

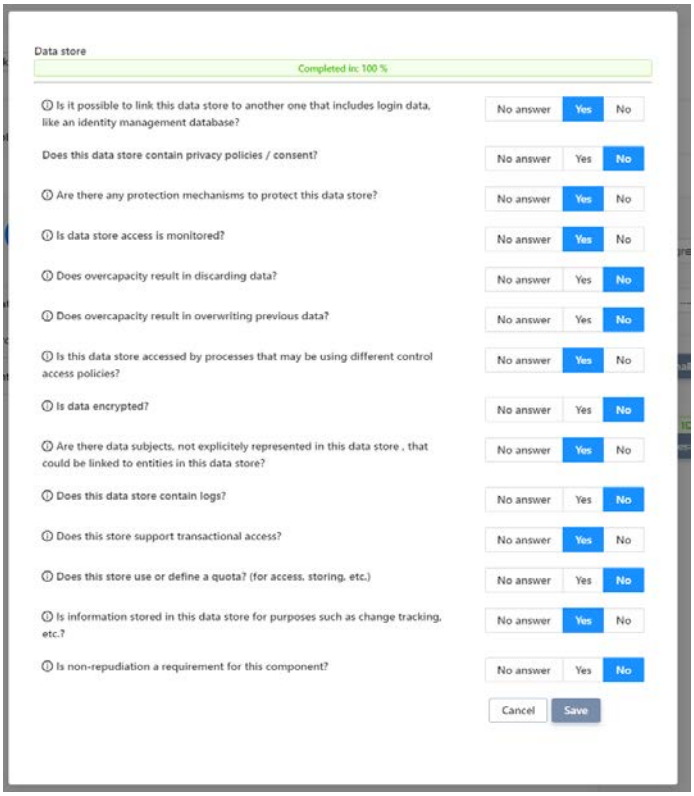


Figure 23: Risk management tool extract of risks and vulnerabilities

This generates vulnerabilities per component type. For this component the tool identified 19 Vulnerabilities, 82 Risks and 133 Treatments.

3.2.3.2 Context Aware Access Control

As already mentioned, the Context Aware Access Control is interesting in the Digital Health use case to achieve some more advanced functionality allowing access to features and edge and IoT devices based on context. For example, surveillance cameras in some elderly people's homes is normally only allowed to be accessed during scheduled time periods (planned supervision) in order to respect and adhere to privacy requirements. However, in emergency situation, e.g., when a safety alarm button is activated, the access to watch video from camera can be permitted even though it is outside the scheduled time periods for planned supervision. Thus, the context-awareness allows privacy rules to be overridden when, for example, the person is in danger. What happens is that if the alarm button is activated, the risk level associated with the user is lowered to grant more access to the devices belonging to this user. In general, we will be able to set up that the more critical the contextual information, the lower the risk, and the greater the access to resources.

The specification of the implementation is depicted in Figure 24.

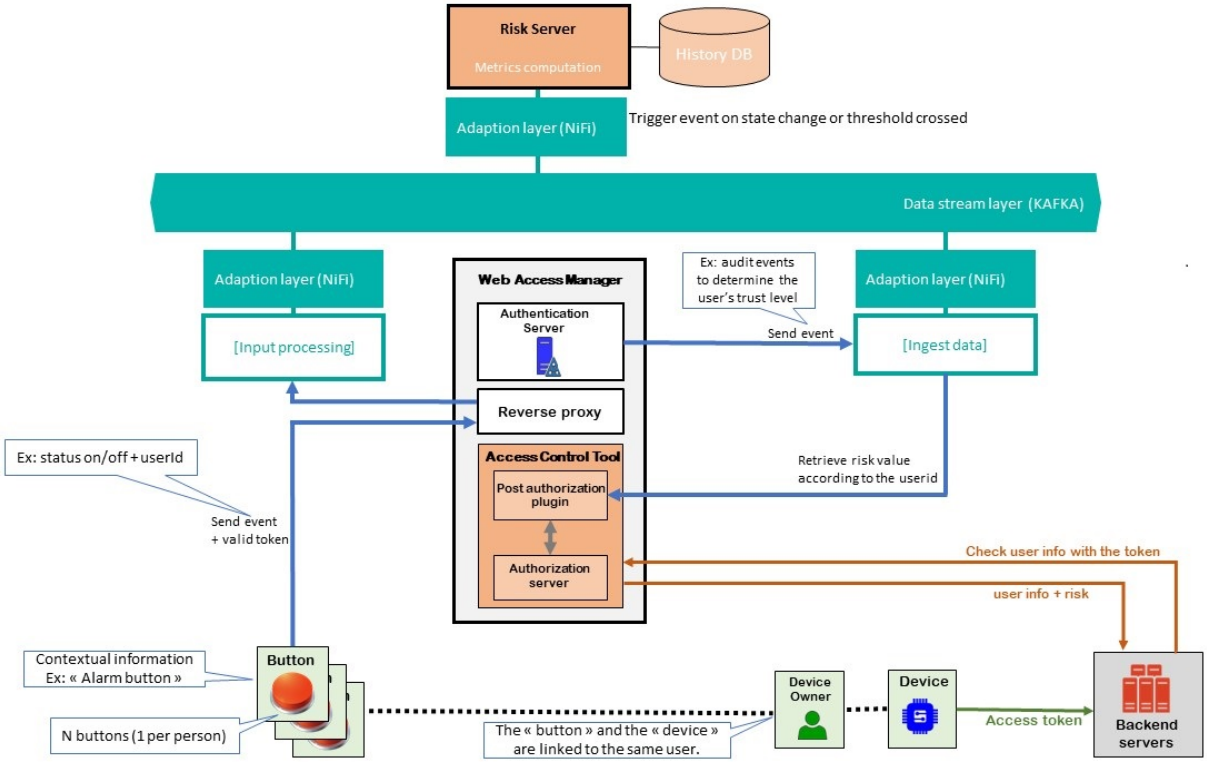


Figure 24: Specification of the application of Context aware access control.

The Web Access Manager is used as OpenID Connect Identity Provider in this infrastructure and various context is monitored, e.g., when an alarm button is pushed the event is transmitted together with some status information and the userID that allows linking the event to the concerned person. The risk server then calculates the risk level associated to the concerned person, and the Access Control tool receives this risk level that will be taken into account in authorization granting. The user information, associated with the user contextual risk level, is obtained by the Digital Health backend server that will then allow access accordingly. In the same way, when a device requests to access resource on the backend server, it requests the server by sending it its access token obtained during its enrolment. The Digital Health Server then requests the Context-Aware Access Control to check this token and get the corresponding user information, including the user's risk level, in order to use it to modulate the accesses accordingly.

3.2.3.3 Orchestration and deployment bundle (ThingML, GeneSIS, and DivEnact)

TellU exploits ThingML part of the Service Orchestration and Deployment Enabler for its DevOps process related to the Personal Health Gateway (PHG). All the low IoT level code (e.g., C code) of the PHG is abstracted by a model based, technology and language independent specification in ThingML Then the ThingML compilers are applied to derive executable code for the distribution to the actual devices.

An extract of ThingML code for the BLE Gateway of the PHG is shown in the Figure 25 and the complete code can be found on Github here: https://github.com/TelluIoT/gateway-enact/blob/master/src/4_BLEGateway/BLEGateway.thingml

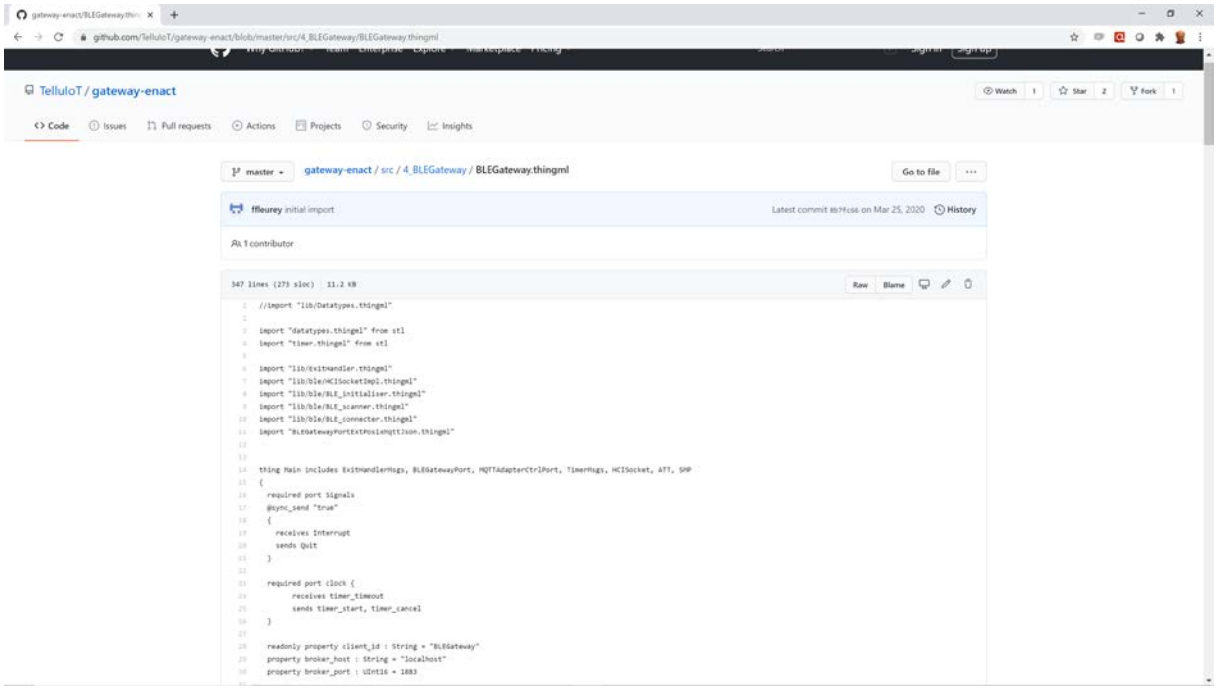


Figure 25: Extract of ThingML code for the BLE Gateway of the PHG

A strength of ThingML and other ENACT enablers is that they can be seamlessly applied together with state of the art DevOps tools such Ansible and Promotheus and these can be applied as an integral part of the ENACT DevOps framework and ENACT DevOps process. For the PHG we apply Ansible for initial set up of the gateway and deployment of the executables and we apply Promotheus as part of the PHG monitoring.

GeneSIS supports the automatic deployment of a SIS within a local subsystem, for example the deployment on a single home. In such case, it is possible to directly interact with the GeneSIS engine hosted on the local edge device, in our case the PHG, and use it as the bridge to further deploy required code to the associated IoT devices within the user's home. GeneSIS integrates with ThingML in particular for the deployment of software on resource limited devices. In the development phase, developers define a deployment model in the GeneSIS modelling language specifying which software artefacts should be deployed onto which devices. An extract of the GeneSIS model Digital Health Use case is shown in . We also apply NodeRed in conjunction with GeneSIS to further enable continuous DevOps management of the PHG.

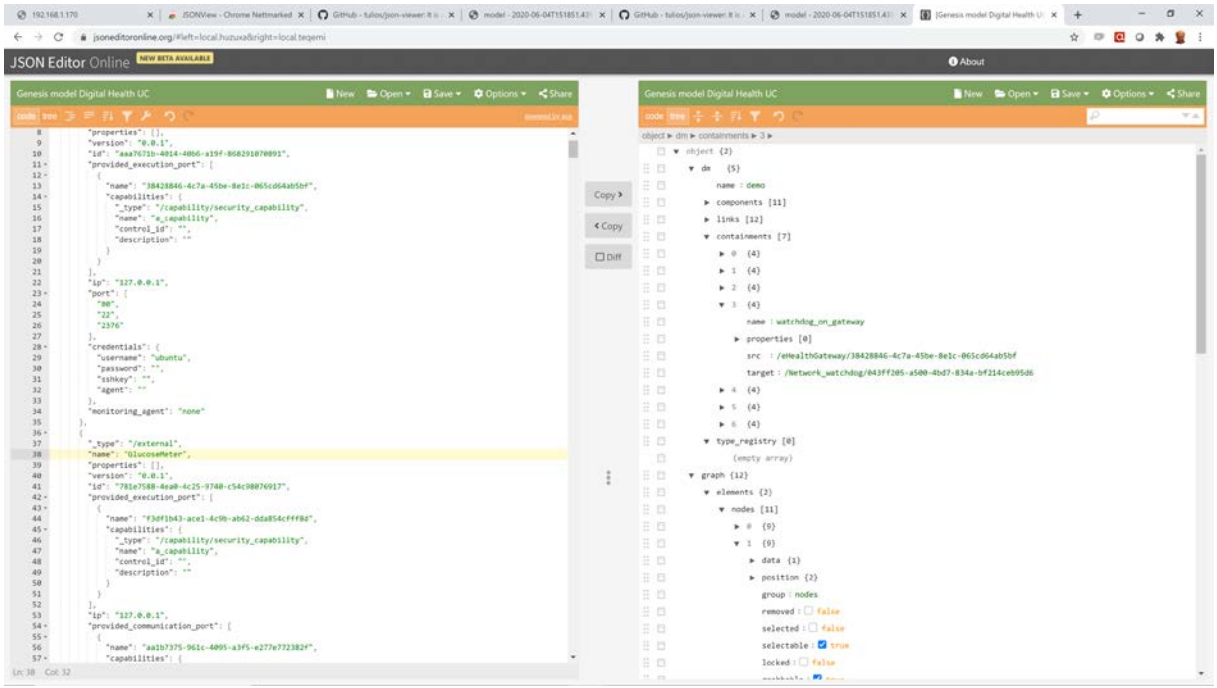


Figure 26: Extract of GeneSIS model for the Digital Health Use case

This approach of hosting a GeneSIS engine on the local edge device is relevant as it provides us with a solution to deploy and maintain software on devices with no direct access to the Internet and only accessible via local network and connections. However, while the solution of having developers to interact with the GeneSIS engine hosted on the local edge device is useful and practical in a test environment or when only a few devices are on the field, this approach does not scale when the fleet of devices grows.

In ENACT we have recognized that the automatic deployment of multiple software variants into a fleet of many IoT/Edge devices is a fundamental challenge in the DevOps for the typical IoT/Edge systems in their production phase, when there are consistently new edge devices adding to the fleet. The main objective is to support the heterogenous devices with proper software variants on each of them. DivEnact provides a concept of Fleet Operation, which means the support of the operation of the fleet as a whole, in order to deploy and manage software in a fleet, without caring of each of the individual device Gateways and devices. The DivEnact concept interoperates with other ENACT tools, in particular the GeneSIS.

The conceptual set up of the DivEnact that we have explored in ENACT is depicted in the figure below.

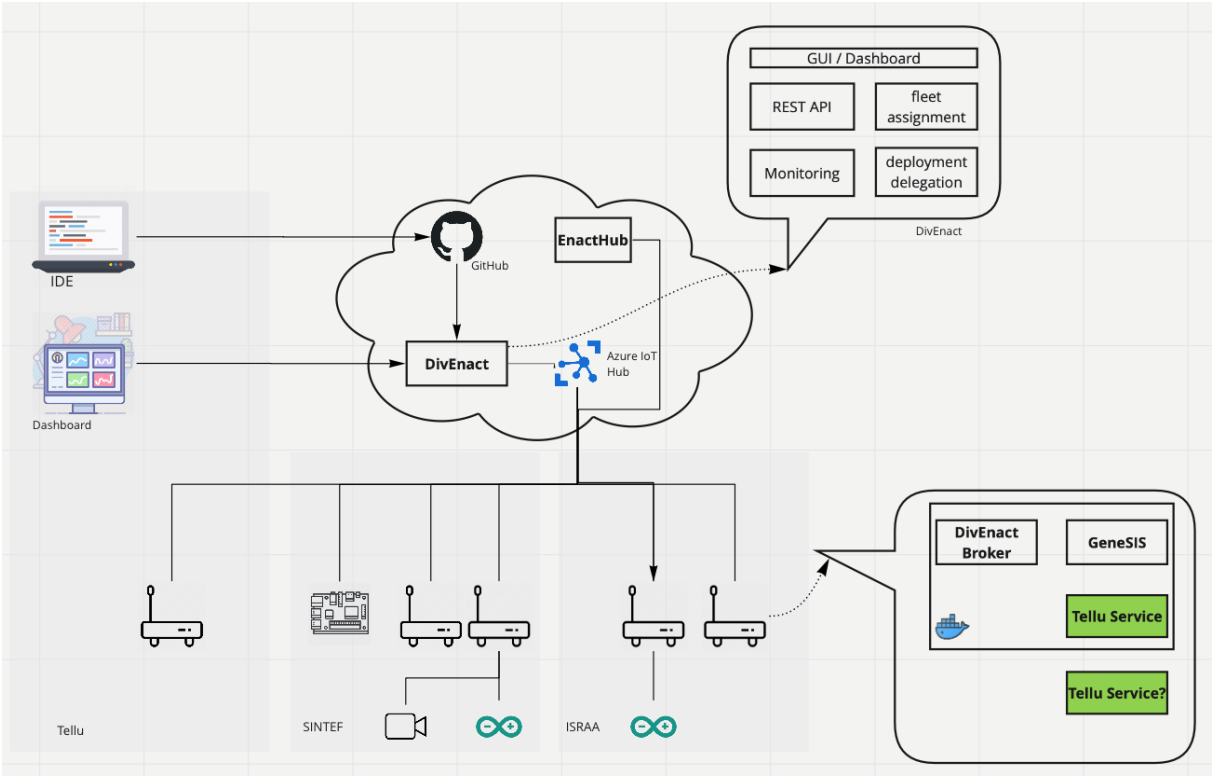


Figure 27: Set up of DivEnact for the eHealth use case

In a typical DevOps loop, developers change code to the IoT/Edge software, normally using an IDE. The change leads to a new version of the software, which they can choose to deploy to the device fleet. This can be done manually through the DivEnact Web GUI, or automatically through a Continuous Delivery pipeline, such as GitHub Action. The operations in the fleet level are:

- Monitoring the list of devices and their contexts
- Add the new version of software as preview
- Release the review version into production
- Rollback specific devices into a previous version

Receiving these commands, DivEnact will invoke the fleet management and assignment modules in the backend to re-allocate the software among the fleet of gateways. The detailed scenario and mechanisms for this assignment can be found in D4.3, using the same eHealth use case as an example. For those gateways that are assigned with a different software version, the backend service will send the new deployment model to the DivEnact agent running on the gateway. The DivEnact agent on the relevant gateway can further invoke the GeneSIS engine to actually deploy the software, both on the gateway and on the devices associated to the gateway.

In the Digital Health use case, the set-up is that elderly or patients are living in their own residences and we provide a Personal Health Gateway as a small computer, together with a set of medical sensors, cameras wearable emergency beepers etc. Each gateway collects measurements such as blood pressure, glucose and oxygen levels, etc., via Bluetooth, processes and aggregates the data, and sends them to the back-end cloud services. The patients and their nurses have access to the data via a Web interface and a mobile App. For some patients, technicians mount the gateway on the wall, while for other patients get the gateway delivered from the manufacturer and install it themselves. Some patients choose battery-powered portable gateways to have a possibility to carry them with the essential sensors whenever they are outside their houses, e.g., for walking, taking medical examinations, or travelling. TellU provides different variants of the front-end software that fit different contexts of the gateways, and continuously upgrade these variants. We need to manage this in an agile DevOps fashion and constantly need to maintain multiple variants of the services.

For the experimentation and application of the Digital Health Gateway in the ENACT project we have provided a PHG distribution that resides in GitHub here: https://github.com/TelluIoT/gateway-enact, which also includes a user guide for the Technical developers of ENACT Enablers to be able to set up, experiment and validate their enablers in collaboration with us as use case providers.

3.2.3.4 Test and Simulation tool

Software testing is a crucial step in software development processes. Providing a test environment that is a production-like environment and that reproduces the same condition and executions traces as the actual production environment is usually an impossible task. This is even more so in IoT environments where there is a need to test the applications and application context that encompass both hardware and software components. For IoT based systems setting up a production-like environment will both be very costly and, in many cases, impossible. Simulation of the environment is a way to accommodate for this. The Digital Health use case has been applied to evaluate the ENACT Test and Simulation tool. For the development and evaluation of tool according to various requirements the Digital Health use case was applied. The overall architecture of the set up for the use case based evaluation is depicted in Figure 28. The Data Recorder records the data from a production gateway and stores the recorded data into the Data Storage, those recorded data are used as testing datasets for non-regression testing. The simulated sensors simulate some sensors in the gateway and send data to the CloudAgent via Internal Broker. The simulated actuators generate actuated data to control the actuators in Arduino, the actuated data sent via Sensor Bridge. The Sensor Bridges were developed in ThingML See https://github.com/TelluIoT/gateway-enact/blob/master/src/6_Sensor/Sensor.thingml (Arduino sensors) and https://github.com/TelluIoT/gateway-enact/blob/master/src/7_SensorBridge/SensorBridge.thingml (Bridge that puts messages on the MQTT broker). The Regular and Malicious Data Generator generates dataset to be used as input in different testing scenarios such as: the scalability of the message queue, the resilience of the system with abnormal behaviour of the sensors, and the possibility to handle some cyber-security attack (DoS attack).

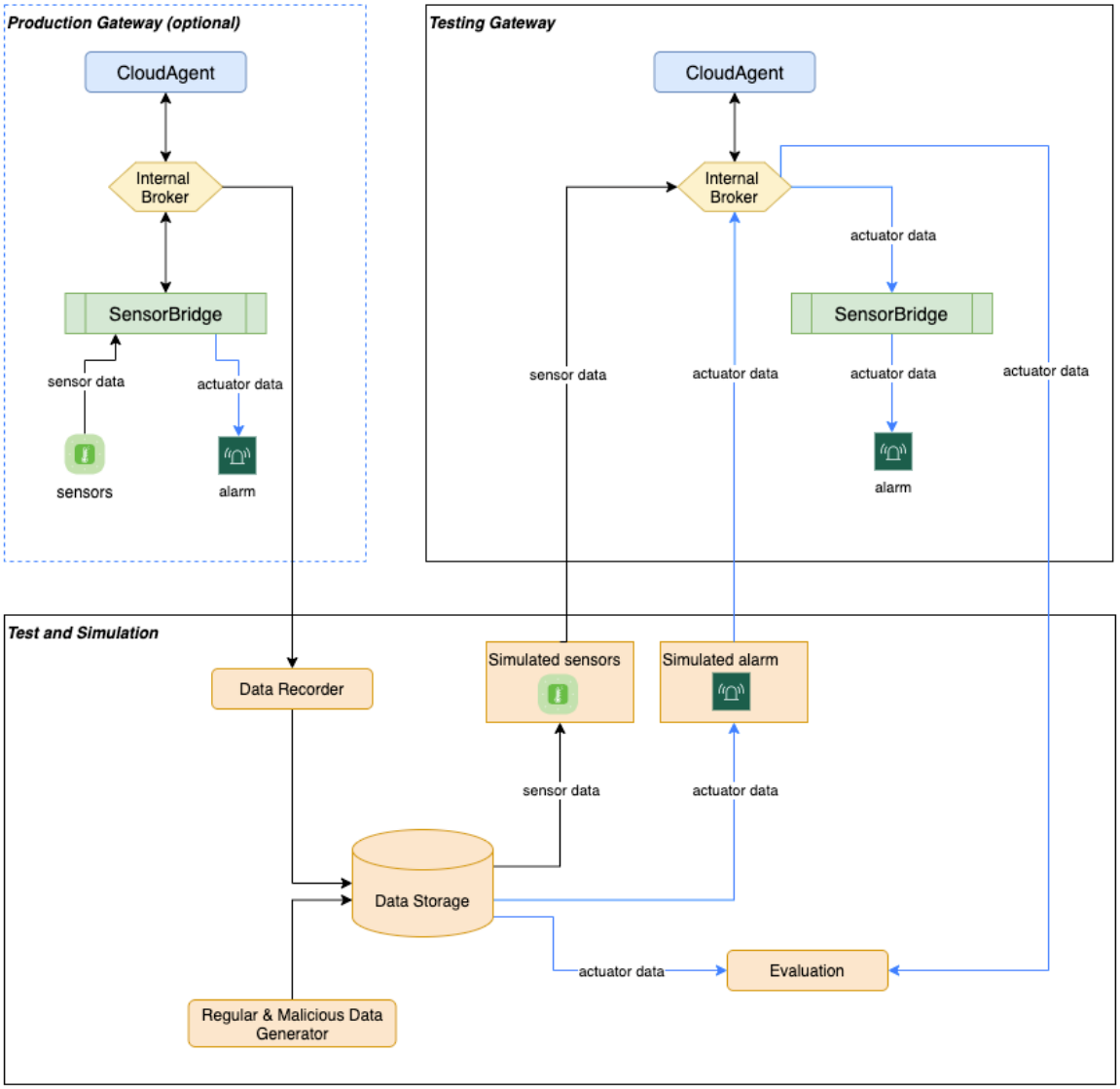


Figure 28: Test and Simulation in eHealth use case

3.2.4 Software

For the application and evaluation of the ENACT enablers various software code is applied. These are referenced in the following

Personal Health Gateway (PHG)

The development of the Personal Health Gateway and the exploration performed in ENACT related to edge and IoT management the source code has been provided in an Enact project Github site here: https://github.com/TelluIoT/gateway-enact

For the production ready PHG the code resides in a private Github repository: https://github.com/TelluIoT/gateway-rpm (version, access on request)

The eHealth Use Case application

The source code of the Digital Health use case and the application itself resides in TellU's Github, as it is proprietary code. The Digital Health application as it is applied in pilots in ENACT (for example in

the pilot deployed in ISRAA) is accessible through these links (Note: The application is not publicly available and user name and password is needed to get access and run the application):

- Health personnel application: https://mao.shepherd.telenor.no/
- Patient app on Android https://play.google.com/store/apps/details?id=no.telenor.dialogg
- Patient app on iOS: https://apps.apple.com/no/app/dialogg/id1467874186?l=nb

Application of the Context aware access control for the eHealth application
The code of the Context-Aware Access Control framework applied in the eHealth case study is available here:
https://enact-caac-my.evidian.com

ThingML
ThingML open source framework is owned and maintained by TellU. The framework has been applied and evolved in ENACT to be part of the ENACT framework as an ENACT enabler. The source code of ThingML tool is accessible from here: https://github.com/TelluIoT/ThingML

3.2.5 Outlook and further work

TellU business relies on exploiting the potential for efficient scaling of our services across the IoT, edge and cloud space. Moreover, trustworthiness is essential as our services include sensitive health data and need to comply with strict privacy and security requirements. Thus, TellU needs to exploitation and evolve applying technologies such as those provided in ENACT. Moreover, the case study itself has gained significant attention in the company and in the market, thus, TellU has developed a commercial RPM system evolving from the ENACT case study. This includes a production ready Personal Health Gateway and commercial remote patient monitoring services. This production ready system was brought back into the ENACT project to further integrate explore and validate ENACT technologies and to set up the pilot system in ISRAA. Status as well as outlook and further work for some of the main ENACT enablers and tools as well as for the case study itself are summarised in the table below.

ENACT Enabler/Tool & eHealth Case Study	Status	Outlook and further work
ThingML	Integrated and applied in TellU DevOps toolchain, in particular for the development and evolution of the Personal Health Gateway	TellU plans to further invest and evolve its exploitation of ThingML as part of its DevOps tool chain, and to further contribute to the ThingML Open Source framework. TellU is part of Several projects that have been funded, where the plan is to evolve and further exploit ThingML. In particular, the SMILE project funded by the EC, where the plans include to evolve ThingML to support the HL7 FHIR recent contribution to standardise Personal Health Gateways in general, and the Fleet project funded by the Norwegian Research Council, where the plans include to further exploit and evolve the ENACT enablers DivEnact, GeneSIS and ThingML in a DevOps tool chain to better cope with DevOps across the IoT, edge and cloud space.

Orchestration and deployment bundle (DivEnact+GeneSIS)	Integrated and applied in TellU experimental tool chain for the ENACT eHealth case study	TellU plans to further invest to better cope with large scale deployments of Personal Health Gateways. For this we will further explore the DivEnact and GeneSIS enablers. Together with SINTEF we have received funding from the Norwegian Research Council (NRC) for a new project under their Program "Industry Driven Innovation Project". The project is denoted Fleet. In this project we will build from the developments and exploitation performed in ENACT to further evolve and exploit the ENACT enablers and tools DivEnact, GeneSIS and ThingML.
Risk Management tool	Integrated and applied in TellU experimental tool chain for the ENACT eHealth case study	TellU is certified according to the ISO 27001 standard on Information Security. A central part of coping with our Information Security and to ensure trustworthy services to our customers is to continuously perform risk analysis and manage risks as an integral part of our DevOps process. This has been the focus of our evaluation of the Risk Management Enabler. TellU will continue to invest in Information security, risk analysis and risk management and will continue to improve its seamless integration and automation of risk management tasks.
CAAC	Integrated and applied in TellU experimental tool chain for the ENACT eHealth case study and explored as part of the production version of the TellU RPM system	Context-aware access control is relevant in several scenarios related to our services, both in terms of controlling access from a device on behalf of a user (e.g., a device or service are granted access based on context information) or for particular users to get access to a service based on the context. These use cases need in general to be handled as part of TellU Services and TellU plans to explore further the application of the CAAC enabler in future services.
ISRAA pilot case	ISRAA is currently exploring a beta version of the production ready Remote Patient Monitoring System in a pilot conducted by ISRAA in one of their elderly home facilities. This is also included as part of the experimental set up for the evaluation of the DivEnact enabler	TellU and ISRAA plan to explore new opportunities related to the case study both in commercial setting and as part of initiatives and projects in the eHealth domain both in Italian setting and European setting. For example, ISRAA and TellU collaborated to present ENACT results and the ENACT case study in an Event with ECH Alliance [3]. Moreover, we will further explore opportunities to deploy commercial ready versions of the case study based on the current set up we have with ISRAA (see https://www.enact-project.eu/newsletter17.php)

[3] See https://echalliance.com/ and https://viewstripo.email/efb89ff2-23d0-469a-ae46-0d293502d4001610360940040

For the DevOps process more in general TellU has exploited its participation in ENACT to evolve and improve its DevOps practices and implemented several of the ENACT concepts in its DevOps process and in the tool chain currently applied. A significant benefit we see for the ENACT approach is that its tools and concepts can be well integrated with current state of practice tools such that we already apply in hour DevOps process such as Jira, Github, Ansible, Jenkins, Promotheus and Grafana.

3.3 Smart Building

The Smart Building use case focuses on developing Smart Energy Efficiency (SEE) applications and Smart User Comfort (SUC) applications to be deployed directly inside the KUBIK building to validate the effectiveness of the ENACT DevOps framework. The continuous development, deployment, and operation of simultaneous applications in the execution environment presents several challenges that can jeopardize the trustworthiness of the whole IoT environment. The ENACT tools demonstrated their ability to address some of these challenges with the use of GeneSIS (i) to continuously deploy the applications and, when combined with the S&P control enabler to tune security mechanisms when undesirable behaviour is observed, (ii) for the proper management of actuation conflicts and behavioural drifts, and (iii) the assurance of trustworthy, security and privacy of the communications at runtime.

Figure 29 shows a general architecture of the Smart Building use case. The use case is formed mainly by two different systems that interoperate using the Interoperability SMOOL IoT middleware that has a semantic broker for connecting heterogeneous devices or sources of information. The Building Management System also centralizes all the information of new wireless IoT devices (System 1) and the legacy building control systems of the building (System 2) using a Scada software.

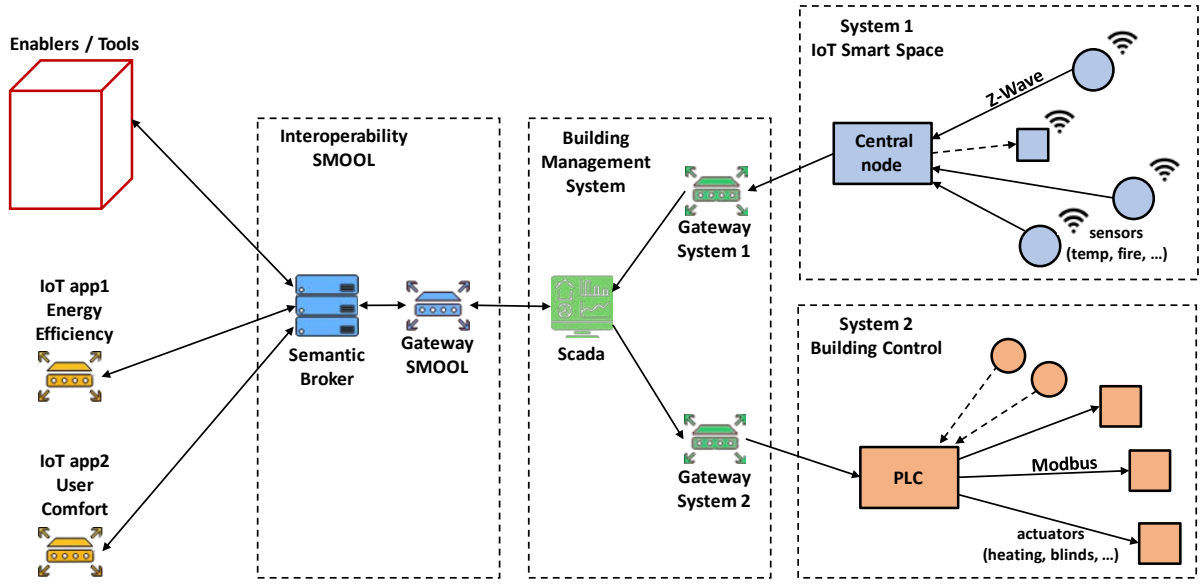


Figure 29: General schema of the Smart Building use case. Source: TECNALIA.

The *System 1* uses a wireless sensor/actuator network using the Z-Wave protocol. These communication channels allow the interoperability of the deployed sensors in the building to read online data and send it directly to a central node which manages the acquired data to be showed in the Scada interface. This type of deployments using Z-Wave protocol are useful in wirelessly environments, e.g., houses or tiny spaces. The advantage of this system is that the deployments costs are minor compared to other solutions because it is not necessary to physically connect each device to the central node and there is no need into buying a centralised hardware, e.g., an industrial PC, to control the connection for each device connected in the system.

The *System 2* is a legacy network which uses PLCs (or industrial PCs) that use building automation protocols (KNX, DALI, PROFIBUS, etc.) and direct control of the devices through relays or

analogic/digital outputs. This system is basically composed by three PLC devices. Each one controls and gathers information from a set of different sensors deployed in the KUBIK experimental building. These PLCs act as a middle agent to understand the information provided by each physically connected device and send it to the gateway system. The sent information will be lately displayed in the SCADA interface to further understand the acquired data. The same PLC device send actuation orders to the actuators to change the behaviour of a target device when necessary. These types of communications are done by using the standardised MODBUS communication protocol system.

A usage example of this PLC systems are the weather sensors deployed in the building roof. One of this PCLs manages all weather sensors with the aim of measuring the weather conditions of the building, e.g., the Global Horizontal Irradiance (GHI), the external temperature or the wind speed. These sensors are directly linked to this PLC device which provides the basic communication between these sensors and the gateway system through MODBUS communication protocol.

In the project execution, the integration with the different enablers has been done into KUBIK experimental building by making the required low interface programming and the integration with the SMOOL middleware. In addition, the KUBIK building has been used as a validation tool for each deployed enabler to ensure that they are working as expected.

3.3.1 Overall Application Description

The case study for the Smart Building system is the management of the different installed devices in a building to improve the comfort rate of building occupants while optimising energy consumption. In this case, different applications try to manage the different actuators using the sensor data to improve the user comfort and at the same time optimise the use of energy by the climate control devices such as heat pumps, fan coils...

In this regard, the use case aimed to satisfy two objectives: 1) the implementation of a complete IoT system to allow the intelligent management of the different deployed devices inside the building; 2) the implementation of a system which allows the intervention of different devices to apply optimal policies which allow to send proper actuations when required.

The implementation of this architecture, involves the necessity of implementing different elements to take into account several consideration such as: 1) actuation conflicts in which different applications send contradictory orders to an actuator or interfere when trying to control a physical variable, such as indoor temperature; 2) risk management when adding a new device that can compromise the whole trustworthiness of the IoT environment; 3) a monitoring of the communications to protect the data readings and actuations from tampering, including an authentication of the command orders by using a secure token when acting on a sensitive actuator.

In addition, the case study aims at making a direct integration of different sensors and actuations inside the KUBIK building using the SMOOL middleware. These sensors and actuators are mainly managed by two different communication protocols, in this case, Z-Wave and a legacy MODBUS communication standard.

Thus, in overall, the following infrastructure has been developed:

1. An intelligent control management system to control the HVAC of the KUBIK building to ensure the thermal comfort levels of the occupants.
2. An intelligent shutter control which acts based on the current irradiance levels and light requirements. At this use case (described lower) 6 shutters where controlled, 3 are inside the Kitchen area of the building and 3 are in the living room.
3. An alert system to inform when the operating comfort levels are not optimal in the target room.
4. The backend software to allow the data acquisition and actuations processes.

3.3.2 Architecture and main scenarios of final implementation

As explained in the introduction of this section, the KUBIK experimental building and its internal components are the main scenario used to integrate and test the ENACT Enablers supporting SIS in Smart Buildings. The final pilot implementation conducted inside this building enabled to test different enablers that addressed different issues that were studied for the first time such as actuation conflict management and security monitoring of the communications in the building.

Figure 30 shows a high-level architecture of the components deployed in the KUBIK building. As explained before, the sensors are connected to a SCADA interface which shows the information captured by the System 1 and System 2. Each set of sensors connected in System 1 and 2 sends the information using their protocol (Z-Wave, MODBUS,) to the PLC or node service to send it to a middle Gateway service (Gateway 1 and 2). These Gateways allow the intercommunication between nodes/PLCS and the SCADA interface to read/write the data.

Two IoT applications are simultaneously running in the Smart Building. The IoT user comfort application aims to maintain the temperature of the building according to the preferences of the inhabitants of the building. The IoT energy efficiency application, on the other side, aims to keep the room at a reasonable temperature while minimizing energy consumption. Both applications collide when the temperature preferences of the inhabitants are far beyond the appropriate temperatures for energy efficiency.

It is interesting to note that the IoT platform used in this building is the SMOOL platform, which is based in SOFIA, and fully described in ENACT deliverable D4.3.

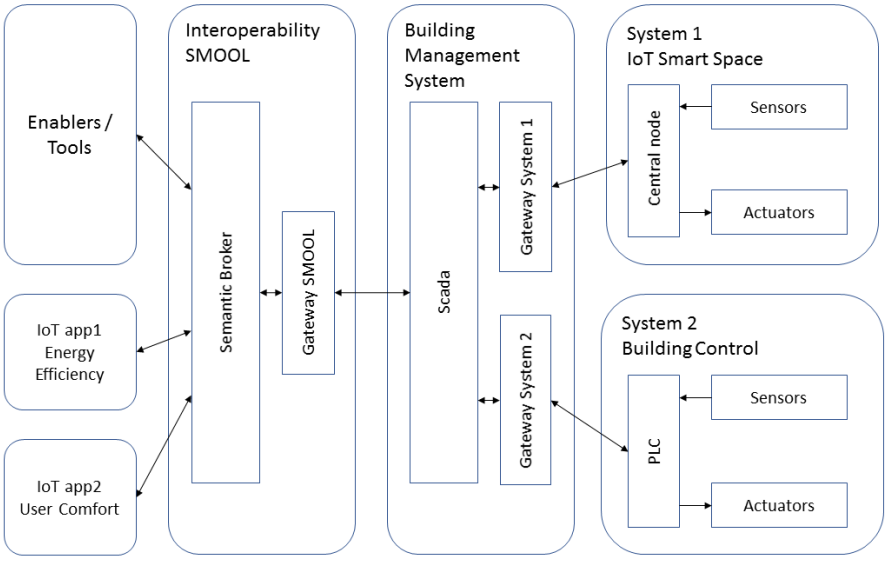


Figure 30: Updated High Level architecture of the Smart Building use case. Source: TECNALIA.

The ground floor of the KUBIK building is the main location for the Energy Efficient Building functionality. This ground floor has been divided into different rooms which mimics an apartment as shown in Figure 31: bedroom, kitchen, living room and corridor.

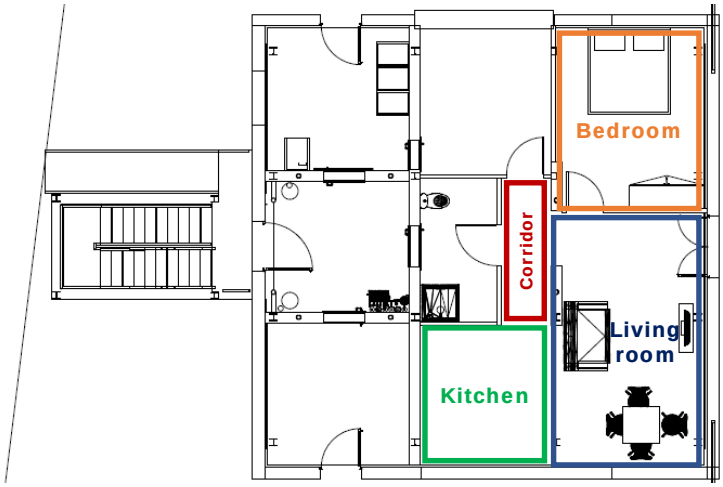


Figure 31: Rooms on the Ground Floor of the KUBIK Building. Source: TECNALIA.

The floor plan and the reflected ceiling plan of the ground floor of KUBIK with the ENACT sensors and actuators in their approximate location are shown in Figure 32 and Figure 33.

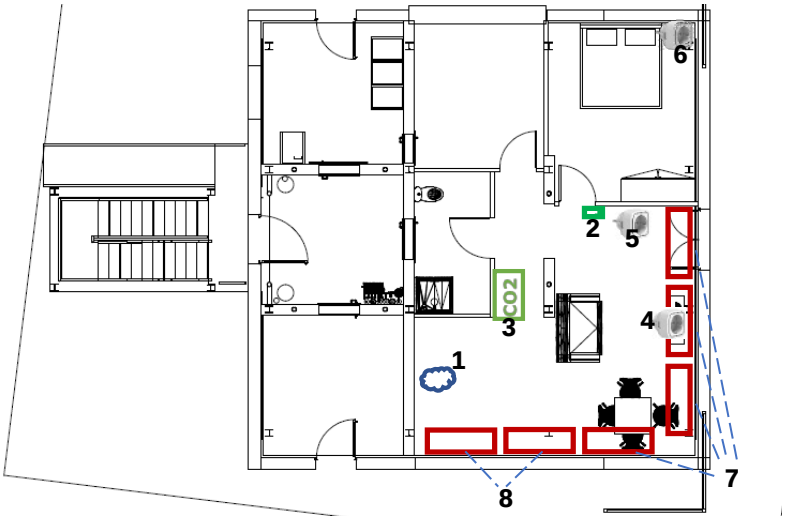


Figure 32: Floor Plan of the Ground Floor of KUBIK. Source: TECNALIA.

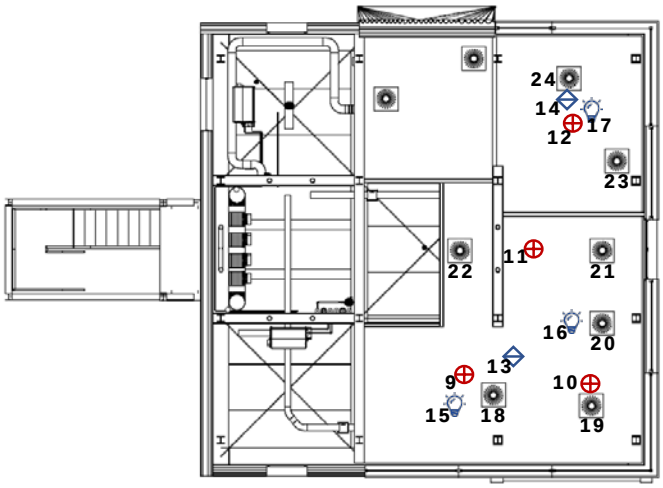


Figure 33: Reflected Ceiling Plan of the Ground Floor of KUBIK. Source: TECNALIA.

Figure 34 shows a detailed description of each device represented in the floor plan of the ground floor of KUBIK (Figure 32), its location in a specific room, its belonging to the IoT Smart Space or the wired Building Control group and, finally, the measures or commands it provides. The same is done in Figure 35 for the devices shown in reflected ceiling plan of the ground floor of KUBIK (Figure 33).

Ground Floor - Floor Plan										
Device				Location				System		Signal
ID	Sensor	Actuator	Device type	Bedroom	Kitchen	Living Room	Corridor	IoT Smart Space Z-Wave wireless	Building Control PLC wired	Measure/Command type and Unit
1	X		Water Flood Sensor		X			X		Flood alarm state (binary): ON/OFF
2	X		Door Multisensor 1	X				X		Position (binary): OPEN/CLOSED
3	X		CO2 Sensor 1		X	X	X	X		CO2 level: 0 ppm to 2000 ppm
4		X	Remote Socket 1			X		X		Switch state (binary): ON/OFF
4	X		Remote Socket 1			X		X		Electric energy consumption: Watts
5		X	Remote Socket 2			X		X		Switch state (binary): ON/OFF
5	X		Remote Socket 2			X		X		Electric energy consumption: Watts
6		X	Remote Socket 3	X				X		State (binary): ON/OFF
6	X		Remote Socket 3	X				X		Electric energy consumption: Watts
7		X	4 Motors for Blinds			X			X	Position (binary): UP/DOWN
8		X	2 Motors for Blinds		X				X	Position (binary): UP/DOWN

Figure 34: Devices/Signals on the Ground Floor and Floor Plan of KUBIK. Source: TECNALIA.

Ground Floor - Reflected Ceiling Plan										
Device				Location				System		Signal
ID	Sensor	Actuator	Device type	Bedroom	Kitchen	Living Room	Corridor	IoT Smart Space Z-Wave wireless	Building Control PLC wired	Measure/Command type and Unit
9	X		Ceiling Multisensor 1		X			X		Motion (binary): YES/NO
9	X		Ceiling Multisensor 1		X			X		Temperature: degrees Celsius
9	X		Ceiling Multisensor 1		X			X		Light: 0 lux to 1000 lux
9	X		Ceiling Multisensor 1		X			X		Relative humidity: 20% to 95%
10	X		Ceiling Multisensor 2			X		X		Motion (binary): YES/NO
10	X		Ceiling Multisensor 2			X		X		Temperature: degrees Celsius
10	X		Ceiling Multisensor 2			X		X		Light: 0 lux to 1000 lux
10	X		Ceiling Multisensor 2			X		X		Relative humidity: 20% to 95%
11	X		Ceiling Multisensor 3			X	X	X		Motion (binary): YES/NO
11	X		Ceiling Multisensor 3			X	X	X		Temperature: degrees Celsius
11	X		Ceiling Multisensor 3			X	X	X		Light: 0 lux to 1000 lux
11	X		Ceiling Multisensor 3			X	X	X		Relative humidity: 20% to 95%
12	X		Ceiling Multisensor 4	X				X		Motion (binary): YES/NO
12	X		Ceiling Multisensor 4	X				X		Temperature: degrees Celsius
12	X		Ceiling Multisensor 4	X				X		Light: 0 lux to 1000 lux
12	X		Ceiling Multisensor 4	X				X		Relative humidity: 20% to 95%
13	X		Smoke Detector 1		X	X	X	X		Smoke alarm state (binary): ON/OFF
14	X		Smoke Detector 2	X				X		Smoke alarm state (binary): ON/OFF
15		X	Ceiling Light 1		X				X	Light state (binary): ON/OFF
16		X	Ceiling Light 2			X			X	Light state (binary): ON/OFF
17		X	Ceiling Light 3	X					X	Light state (binary): ON/OFF
18		X	Fan Coil 1		X				X	Operation state (binary): ON/OFF
18		X	Fan Coil 1		X				X	Temperature setpoint: Celsius
19		X	Fan Coil 2			X			X	Operation state (binary): ON/OFF
19		X	Fan Coil 2			X			X	Temperature setpoint: Celsius
20		X	Fan Coil 3			X			X	Operation state (binary): ON/OFF
20		X	Fan Coil 3			X			X	Temperature setpoint: Celsius
21		X	Fan Coil 4			X			X	Operation state (binary): ON/OFF
21		X	Fan Coil 4			X			X	Temperature setpoint: Celsius
22		X	Fan Coil 5				X		X	Operation state (binary): ON/OFF
22		X	Fan Coil 5				X		X	Temperature setpoint: Celsius
23		X	Fan Coil 6	X					X	Operation state (binary): ON/OFF
23		X	Fan Coil 6	X					X	Temperature setpoint: Celsius
24		X	Fan Coil 7	X					X	Operation state (binary): ON/OFF
24		X	Fan Coil 7	X					X	Temperature setpoint: Celsius

Figure 35: Devices/Signals on Ground Floor and Reflected Ceiling of KUBIK. Source: TECNALIA.

3.3.3 Application of the ENACT enablers

3.3.3.1 Risk driven decision support Enabler

In case of Risk Management Enabler, Tecnalia has developed a process to detect the required components by using the GeneSIS model and performs a list of mitigation actions to avoid any type of vulnerabilities or threats in the data management processes. Each process is described in the following sections.

3.3.3.1.1 Risk analysis of the architecture components

In case of Risk analysis of the architecture components, the usage of a Data Flow Diagram (DFD)[4] was critical in order to detect what are the most suitable components to be analysed and taken into account to create the required conditions to apply the list of mitigation actions based on the possible detected threats.

To accomplish this task, first a Data-Flow Diagram has been proposed based on GENESIS model to detect what are the possible elements which can be seriously compromised by a Smurf attack (which causes a service denial) or an improperly control of the different connected systems. Example of the KUBIC GeneSIS model loaded to the tool is illustrated in the Figure 36.

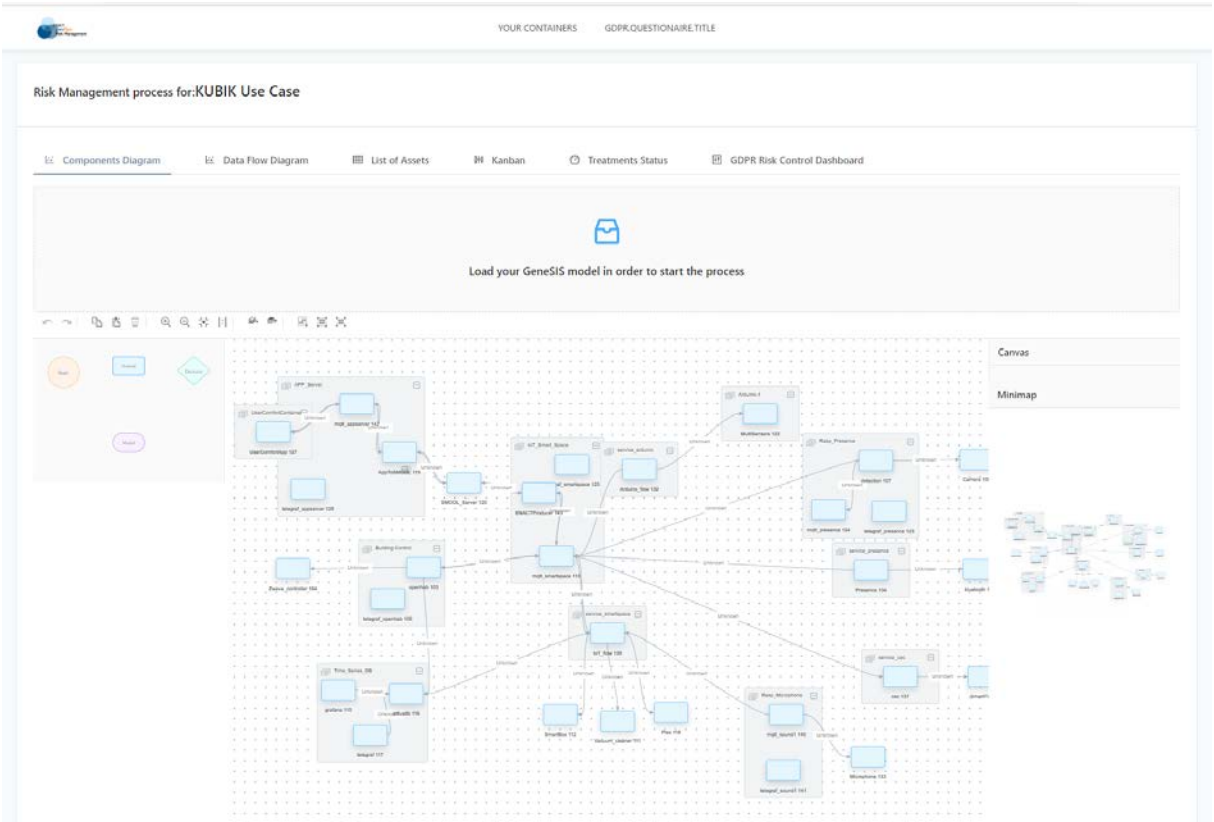


Figure 36. Architectural view for the KUBIK use case in the Risk Management tool

The DFD architectures can be modelled into different ways based in one the way the data flow's within the described architecture, this is something required because these diagrams represent the "logical" decisions of the different connected systems to understand how they are working and what type of different problems could appear. Figure 37 shows one of the DFD diagrams which shows the modulization of a DND to understand the flow of the intelligent shutters installed on KUBIK building.

[4] https://en.wikipedia.org/wiki/Data-flow_diagram

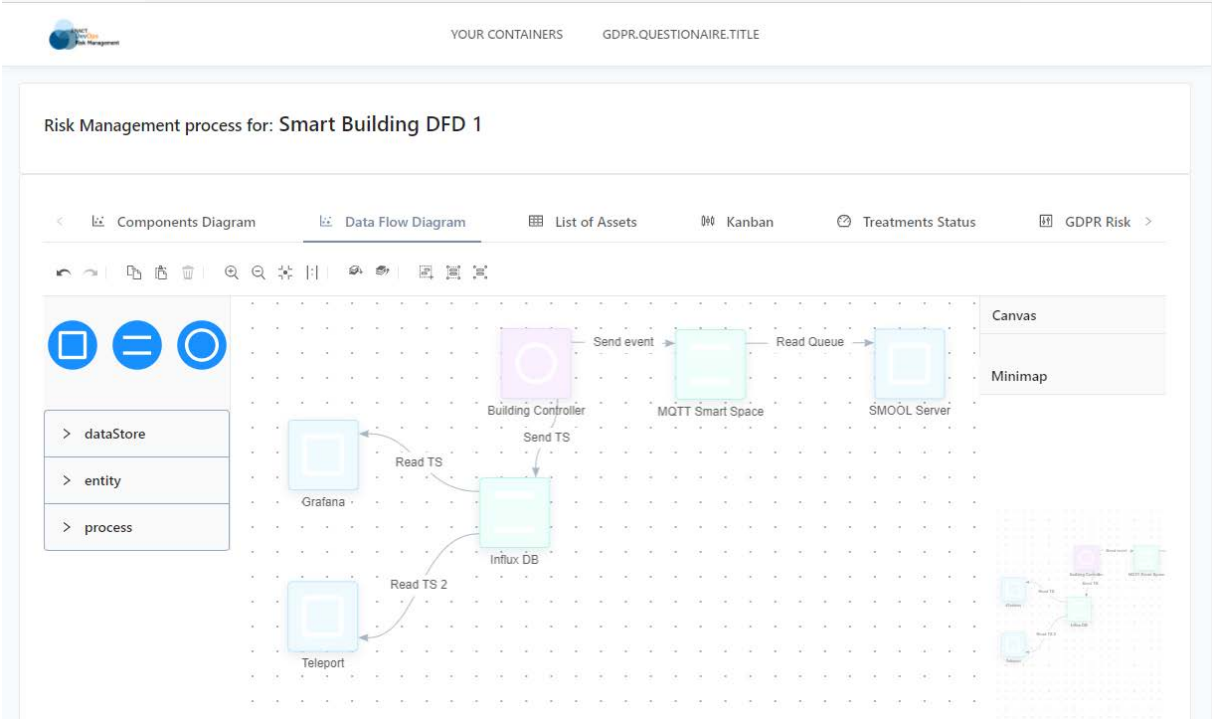


Figure 37. Example of the DFD diagram illustrating a scenario for storing Time Series information.

3.3.3.1.2 Detecting, assessing and implementing mitigation actions

After understanding the different requirements to detect the possible threats inside the Dependency Network Diagram (DND), partners proposed to address the CWE-664[5] and the CVE-1999[6] threats because they are common in any software development process. These two threats are critical to detect any intrusion on the process of managing the installed intelligent blinds and provide the needed counter measurements to avoid any type of smurfing or attack to the installed system.

Figure 38 provides an example of the proposal of the mitigation actions are shown taking consideration based on the detected threats. As it is possible to see, the main idea behind this actuation is to avoid the takeover of the system or the modifications of each one by providing a simple disconnect/reconnect trigger and a warning message to the project administrators to inform that the system was compromised. This type of measurements ensures the quality of the installed solutions in the Smart Building and allows to protect the occupant's privacy.

[5] https://cwe.mitre.org/data/definitions/664.html
[6] https://cwe.mitre.org/data/definitions/664.html

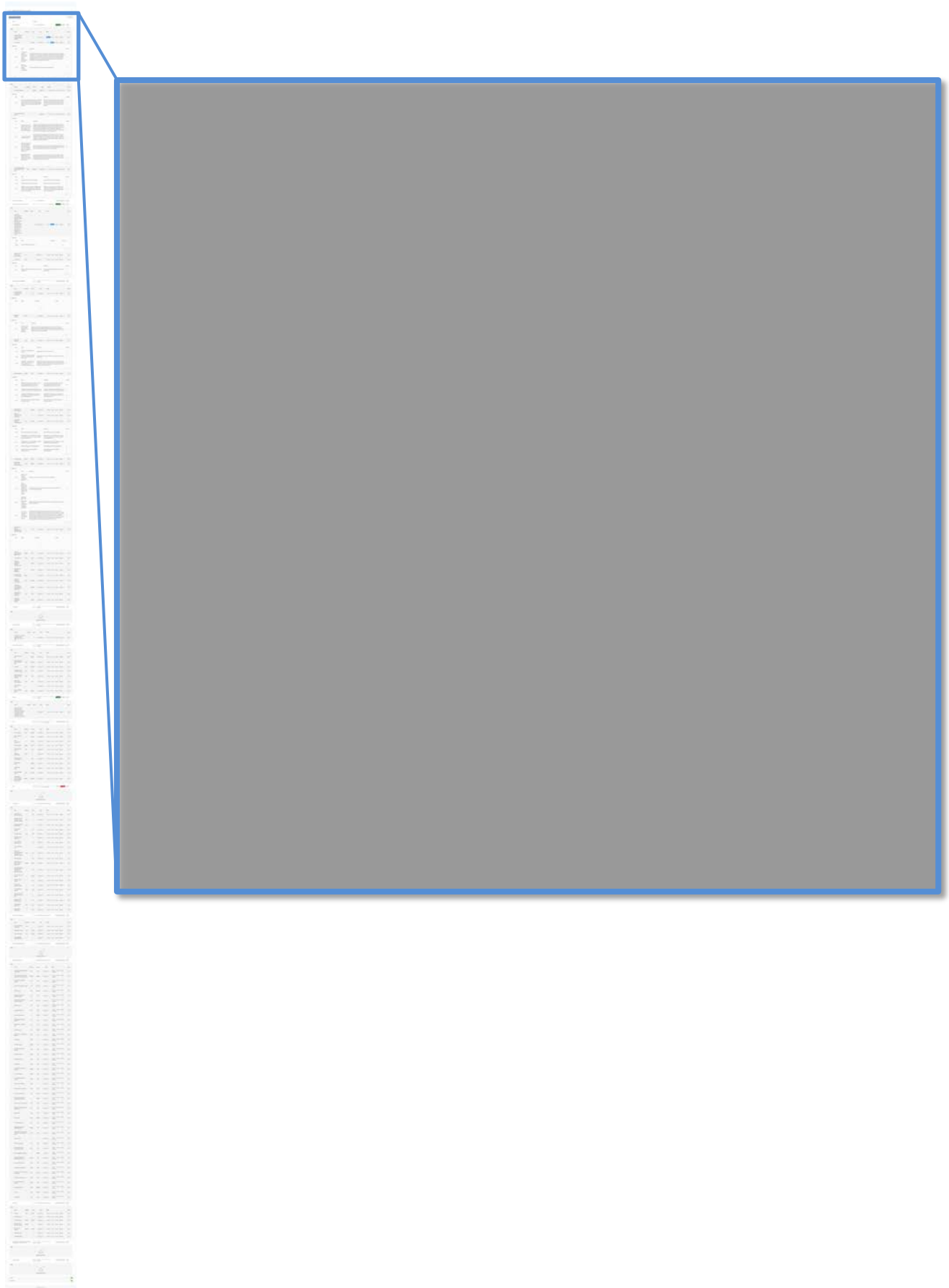


Figure 38. Example of proposed mitigation actions against a list of them on a single asset (in this case "Building controller")

Based on the run scenarios, and considering that the automatic detection engine of the Risk Management tool proposes on average:

- from 10-40 vulnerabilities per asset
- from 10-80 risks per asset
- from 30-300 mitigation actions per asset

depending on the range of answers in the DFD Questionnaire, we found that around 70% of the proposed vulnerabilities and risks are adequate and proposal of mitigation actions is considered sufficient.

3.3.3.2 Orchestration and Continuous Deployment, Actuation Conflict Management and Behavioural Drift Analysis Enablers

GeneSIS and the Actuation Conflict Management enabler were already successfully applied at mid-term of the project and details about their application in the use case can be found in D1.2. In the second period, the exploitation of these two enablers in the smart building scenario was further investigated to better understand the benefits provided by the two enablers, with a strong focus on their integration, in particular as part of a DevOps process and pipeline. It is worth noting that, when restrictions to access KUBIK appeared in relation to COVID-19, hindering our ability to quickly set up KUBIK (configure access to devices for partners, installation of new edge devices, sensors and actuators), as a preventive measure, the partners agreed to set up a complementary laboratory in CNRS premises. This laboratory includes a set of devices complementary to the one offered in KUBIK (approximatively 57 IoT devices providing up to 319 parameters including 249 sensor/actuators). The devices in this smart home laboratory are interacting with the same instance of SMOOL server used in KUBIK thus offering developers with a single entry point for accessing all the devices.

In the following we first shortly recall the experiments performed in the first period, which already fulfilled the smart home use case KPIs for these two enablers. Second, we detail the new experiments conducted in the second period.

3.3.3.2.1 User comfort and energy efficiency applications in KUBIK

The deployment of applications in KUBIK and the management of actuation conflict was already demonstrated in D1.2. For instance, these two enablers were used in a scenario consisting of two IoT applications (see Figure 39) fed with temperature values from two different sensors and both trying to change the state of an actuator in the PLC depending on the moment when those temperature values reach a threshold mimicking the operation of a thermostat. The temperature sensors were located at different spots in the room and therefore different temperature values are potentially obtained possibly leading to conflicting actuation orders. The conflicts were managed using the Actuation Conflict Management enabler and the applications were deployed using GeneSIS.

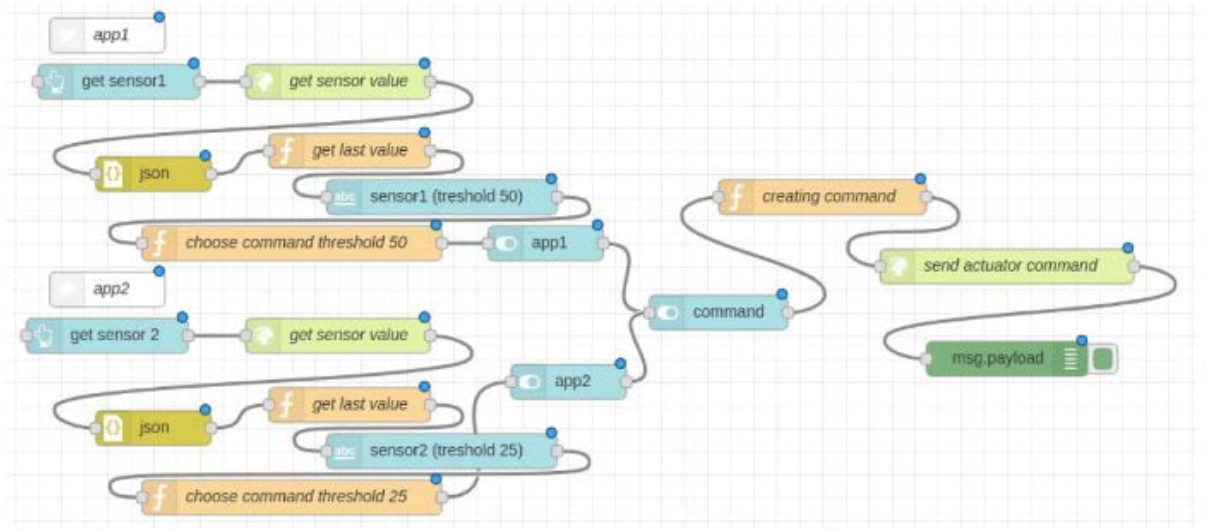


Figure 39. Applications conflicting when managing the temperature in KUBIK.

The integration of the Behavioural Drift Analysis (BDA) enabler is done in KUBIK building. The covered case is the actuation of heating a room towards different connected elements in the building, for example, an electric heater connected using Z-Wave (System 1) or a fan coil unit in "heating mode" connected via PLC (System 2). The actuation in cooling can be done also through access to the fan coil in "cooling mode", but only using the PLC device (System 2).

One of the problems encountered in this implementation is that it is not possible to modify the ventilation of the fan coil unit because KUBIK building has a centralised air handling unit (AHU) which ventilates the entire building making not possible to actuate individually in the air control of a room or floor.

In addition of the heating/cooling control modes, the actuation of motorized windows and doors has been taken into account to provide an additional ventilation control in the building and have a better adjustment if, for some reason, there is not a direct way of providing cooling in the room when needed (for example, because it is not possible to control directly the airflow rates of the fan coil unit).

Figure 40 depicts the cartography of devices installed in the KUBIK building. As it is possible to see, the numbers 7 and 8 contain the living room and kitchen blinds, the numbers 9 to 12 contain the sensors related to light, occupancy and humidity and numbers from 15 to 17 are related to lights in living room, kitchen, and bedroom.

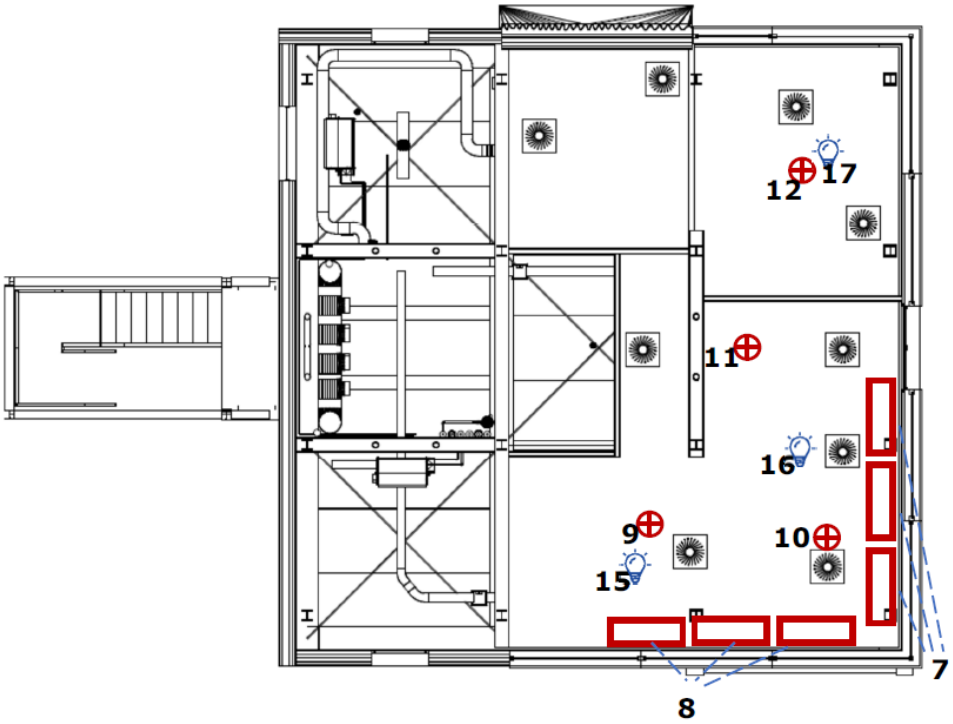


Figure 40: Considered sensors/actuators to implement the BDA engine in the KUBIK's ground floor.

Table 1 contains the description of considered sensors to perform the BDA inside the KUBIK building. The first column explains what system is used to manage the information (System 1 or 2). The second and third column identify if the current element is a sensor or an actuator. The next column shows the real name inside the KUBIK building system. The Notes column provides a better understanding of the aim of the sensor. The last column provides the id of the sensor according to the Figure 32 showed above.

Groups	Sensor	Actuator	Topics	Notes	Ground floor ID
Zwave	1	0	domotics_zw/F0M1_MS/pir	Occupancy	9
Zwave	1	0	domotics_zw/F0M1_MS/temperature	Indoor temperature	9
Zwave	1	0	domotics_zw/F0M1_MS/luminiscence	Illuminance	9
Zwave	1	0	domotics_zw/F0M1_MS/humidity	Relative humidity	9
Zwave	1	0	domotics_zw/F0S1_MS/pir	Occupancy	10
Zwave	1	0	domotics_zw/F0S1_MS/temperature	Indoor temperature	10
Zwave	1	0	domotics_zw/F0S1_MS/luminiscence	Illuminance	10
Zwave	1	0	domotics_zw/F0S1_MS/humidity	Relative humidity	10
Zwave	1	0	domotics_zw/F0S2_MS/pir	Occupancy	11
Zwave	1	0	domotics_zw/F0S2_MS/temperature	Indoor temperature	11
Zwave	1	0	domotics_zw/F0S2_MS/luminiscence	Illuminance	11
Zwave	1	0	domotics_zw/F0S2_MS/humidity	Relative humidity	11
Zwave	1	0	domotics_zw/F0S3_MS/pir	Occupancy	12
Zwave	1	0	domotics_zw/F0S3_MS/temperature	Indoor temperature	12
Zwave	1	0	domotics_zw/F0S3_MS/luminiscence	Illuminance	12
Zwave	1	0	domotics_zw/F0S3_MS/humidity	Relative humidity	12
PLC	0	1	DOM/cmdGoUpSalon	Raise the living room blinds	7
PLC	0	1	DOM/cmdGoDownSalon	Lower the living room blinds	7

PLC	0	1	DOM/cmdGoUpCocina	Raise the kitchen room blinds	8
PLC	0	1	DOM/cmdGoDownCocina	Lower the kitchen room blinds	8
PLC	0	1	DOM/cmdTurnOnCocina	Turn on the kitchen lights	15
PLC	0	1	DOM/cmdTurnOffCocina	Turn off the kitchen lights	15
PLC	0	1	DOM/cmdTurnOnSalon	Turn on the living room lights	16
PLC	0	1	DOM/cmdTurnOffSalon	Turn off the living room lights	16
PLC	0	1	DOM/cmdTurnOnDorm1	Turn on the bedroom lights	17
PLC	0	1	DOM/cmdTurnOffDorm1	Turn off the bedroom lights	17
PLC	1	0	DOM/swLightDorm1		
PLC	1	0	DOM/swLightSalon1		
PLC	1	0	DOM/swLightSalon2		
PLC	1	0	DOM/swLightCocina		
PLC	1	0	DOM/swBlindUp		
PLC	1	0	DOM/swBlindDown		
PLC	1	0	ww/temperatureOutdoor	Outdoor temperature	
PLC	1	0	ww/northSolarRadiation	North solar radiation heats the living room	
PLC	1	0	ww/eastSolarRadiation	East solar radiation heats the kitchen	
PLC	1	0	ww/setpointFCM1	Read temperature setpoint value for the living room/kitchen	
PLC	0	1	ww/write.setpoint	Write temperature setpoint value for the living room/kitchen	

Table 1: Technical description of the considered elements.

The actuation on different elements has been done using MQTT messages. For example, to actuate on a fan coil unit, a MQTT message is sent with the attached temperature setpoint (using its internal PID), or the actuation on a Z-Wave remove socket is done by a simple ON/OFF actuation message.

3.3.3.2.2 Advanced DevOps scenario

In this section we detail the advanced DevOps scenario experimented during the second period of the project. A detailed description of the SIS architecture as well as technical details about the application of each enabler in this scenario can be found in D2.3.

In this scenario, the infrastructure exploited mainly consisted in consumer electronics devices (e.g., Netatmo devices, Neato vacuum cleaner, LG TV, etc.), complemented with custom IoT devices based on Arduino Uno/Nano and Raspberry Pi equipped with sensors and actuators targeting specific purposes.

A first application (UserComfortApps) was dedicated to give sensor data access to decision making algorithms (App_LumandApp_RC_TV) that control respectively (1) the luminosity level by acting on the roller shutters and the light bulbs and (2) the sound by acting on the TV remote controller and the Amazon echo smart speaker. A second application (i.e., CommunicationCenterApps) was designed,

creating a home-working environment where the focus is put on controlling sound sources so as to prevent home workers to be disturbed during phone calls or video conferences (e.g., App_Phone_TVmutes TV while a phone call is in progress). All applications are implemented through Node-RED flows.

The deployment model depicted in Figure 41 was first defined. It is worth noting the specified deployment included Arduino devices with no direct access to Internet and the deployment of security policy for SMOOL to provide the applications with access to the sensors and actuators. The corresponding deployment model is available at: https://gitlab.com/enact/smarthome-demo/-/blob/master/genesis/deployment-model-template-step1.json

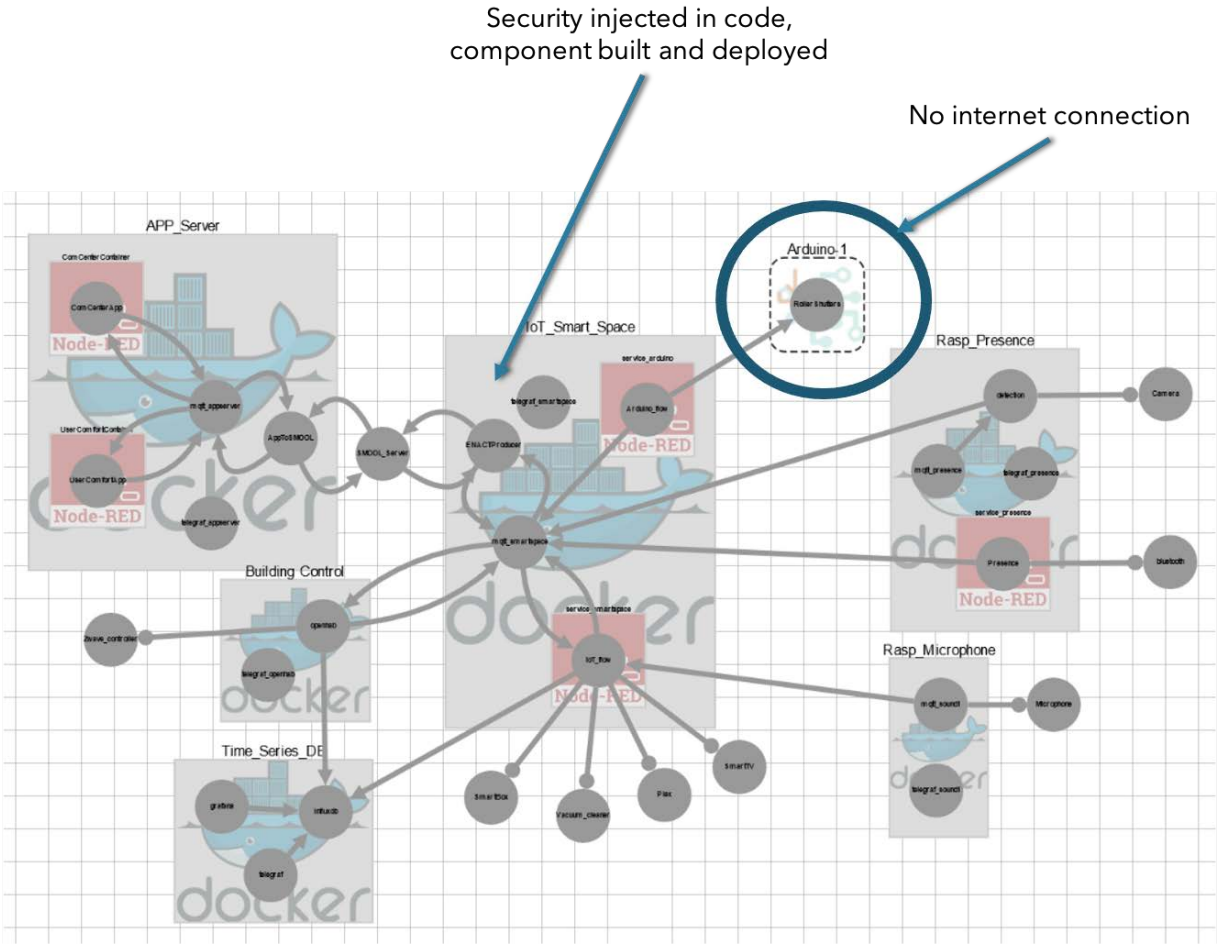


Figure 41. GeneSIS deployment model for the Smart Home SIS

Before deploying the whole system, the Actuation Conflict Manager was used to identify and solve actuation conflicts, relying on the WIMAC model depicted in Figure 42 and built from the GeneSIS deployment model together with the application and physical environment models. At this point, a potential direct actuation conflict was identified related to the concurrent access to the TV by the two applications (UserComfortApp for remote control of the TV and the CommunicationCenterApp to stop the TV when a phone call occurs). The management of this direct actuation conflict was straightforward as the applications merely send a Boolean value to a shared actuator. This conflict was solved by selecting among the available off-the-shelf actuation conflicts managers, in our case a component implementing a OR logic.

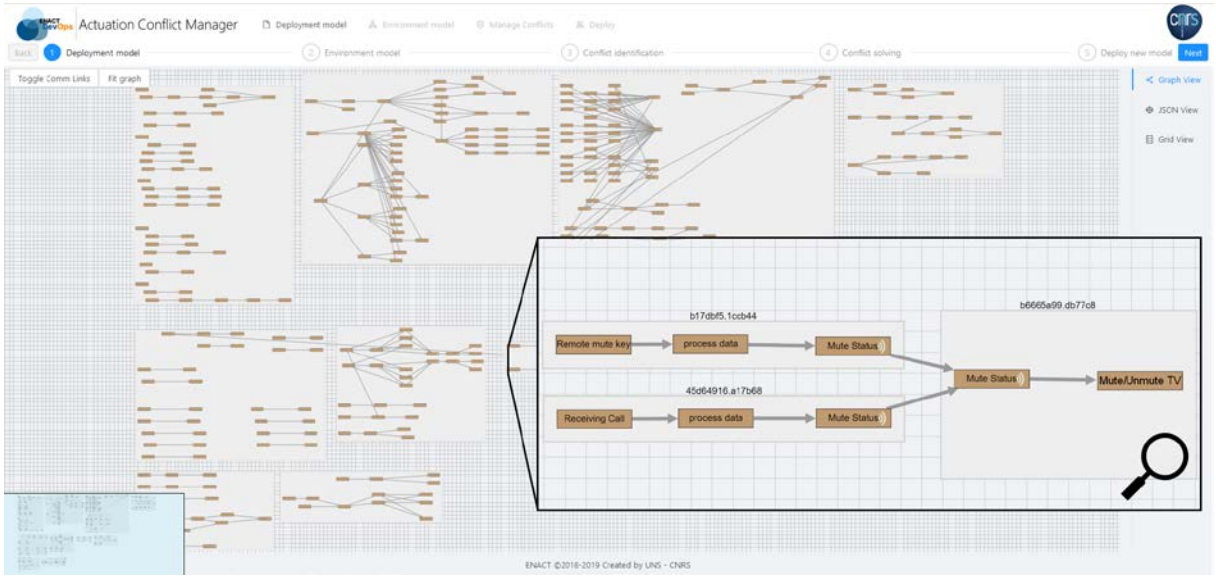


Figure 42. WIMAC model

Once this conflict solved and the application automatically adapted by the Actuation Conflict Management enabler, the deployment was performed using GeneSIS. It is worth noting here the complementarity of the two enablers, not only GeneSIS provides the basis for the Actuation Conflict Management enabler to build a WIMAC model but also the continuous deployment offered by GeneSIS supports its integration as part of a DevOps process. In addition, GeneSIS provides ACM with a single way to seamlessly deploy actuation conflict managers on infrastructure ranging from Cloud to IoT devices.

Together with the SIS we deployed the Behavioral Drift Analysis enabler. The later was configured using the model provided by CNRS, and depicted in Figure 43, with the objective to assess the effectiveness of the SIS in managing the TV. In particular, looking at the operating status of the TV and at the communication status (i.e., OFF/ON means TV:ON,COM:OFF). The configuration ON/ON is not legitimate as the direct ACM instantiated is supposed to mute the TV while a communication is in progress.

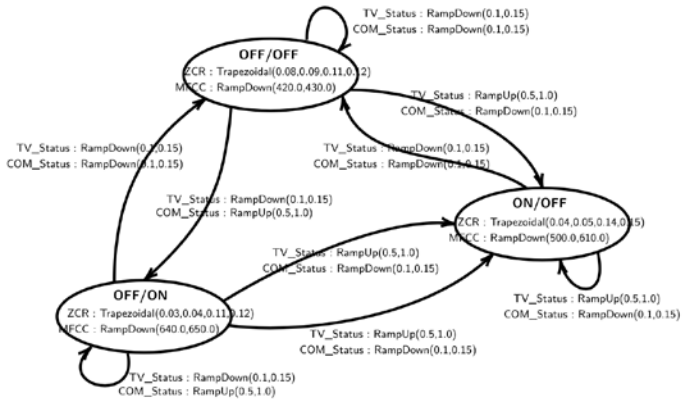


Figure 43. Model of the expected behaviour of the SIS when managing the TV and the communications.

The BDA enabler allowed us to observe the following unexpected behavior. While the SIS was under operation, an autonomous smart vacuum cleaner moving around in the house, bursts into the living-room. By operating in the living-room, this device produces unexpected sounds resulting in behavioural drifts. The reported behavioural drift suggested that the model of the physical environment was incomplete. Indeed, the indirect conflict on the sound physical property should have been identified during the first phase of development. A new development cycle was therefore necessary to correct the

model of the physical environment. A new WIMAC model was produced from the deployment, application, and environment models of the SIS under operation (benefiting from the ability of GeneSIS to maintain up-to-date a model of the SIS deployed). The environment model was updated, specifying the impact of the vacuum cleaner on the audio of the living room and an indirect actuation conflict could be detected, as depicted in Figure 44. A specific Actuation Conflict Manager was designed to solve this conflict and the new SIS was deployed.

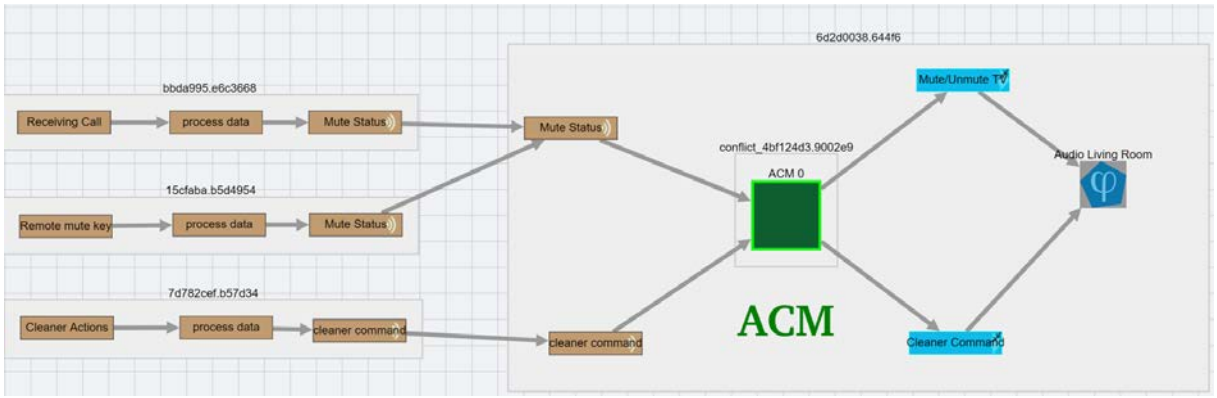


Figure 44. Excerpt of indirect conflict detection and management.

From this scenario we could validate the use and complementarity of the GeneSIS, ACM, and BDA enablers (i) in a single scenario (ii) involving several DevOps iterations. This combination of tools provided the following benefits:

- GeneSIS saved time during the deployment providing a single entry point for specifying how to deploy the whole system including (i) the small Arduino devices and (ii) the security policy enabling the integration of the application with SMOOL.
- Actuation Conflict Management help us saving time when integrating the different applications? In particular, the two application could be developed without worrying about conflicts that may potentially occur when the applications would be integrated.
- BDA clearly helping us understanding and detecting an unexpected abnormal behaviour during operation and its feedbacks drove the subsequent development iteration.

3.3.3.3 Security and Privacy Monitoring and Control Enabler

The full description of the security monitoring and control mechanisms implemented as part of the S&P Mon&Control Enabler can be found in D4.3 section 4.1, thus these descriptions are not repeated here. The Enabler offers network monitoring and SIS situational awareness capabilities validated in this use case, as well as other with security measures developed as SMOOL IoT platform enhancements which are fully exploited in the building to control the security of the communications to and from smart things deployed indoor.

The integration of this Enabler with the KUBIK building systems allows to ensure the security of the communications and transmitted data in the following ways:

- Integration of security monitoring agents in form of distributed sensors to continuously monitor the communications of top value assets of the building such as the main control SCADA.
- Integration of signature-based intrusion detection system and AI-based detection of security anomalies in the KUBIK communications to prevent confidentiality, integrity and availability issues in the building, which may prevent either system 1 or system 2 working properly.
- Integration and customization of notifications and situational awareness dashboard fully tailored to the needs of the KUBIK building security surveillance.
- Integration of SMOOL IoT platform to control communications security between the Mosquitto MQTT broker and the Node-REST exposed API endpoints to ensure that the data are secured and that actuation orders are not tampered with by external third parties (more information about the used technical tools to acquire and send actuations in the use case is provided in the Annex).

The complete process of the authentication of actuation orders through the use of JWT tokens verified by SMOOL is described in the deliverable D4.3.

3.3.3.4 Online Learning Enabler

In collaboration with UDE, Tecnalia helped in the validation of the developed AI algorithms used in the development of the Online Learning Enabler (OLE) by using the KUBIK building data as a ground truth.

To accomplish this task, a model of the KUBIK'S building Ground Floor (see Figure 32) was specified to train a Reinforcement Learning (RL) based agent to make a direct control of a HVAC system installed in the building (fan coils). Figure 45 shows in red, the considered area inside KUBIK to perform the experiments, which consist of the living room / kitchen area. The aim of this enabler was to automatically perform a direct control of the HVAC elements to ensure the comfort levels while the energy consumption is reduced to save costs. The HVAC decisions was conducted by an Artificial Intelligence (AI) based on Reinforcement Learning (RL) algorithms.

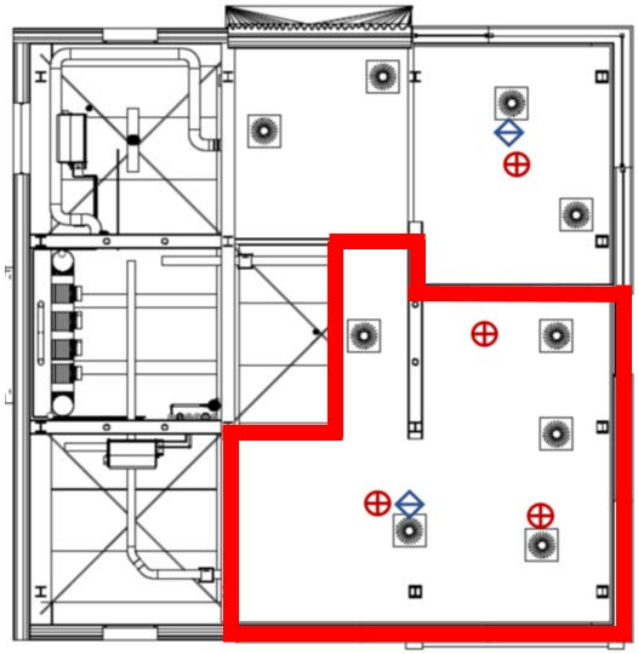


Figure 45: Considered experimental area selected from the building. Source TECNALIA

The living room/kitchen area contains the following materials: 1) two big windows; 2) 6 walls internal and external; It also has the following elements 3) 2 fan coils linked with the heat and cool plant of the building to control the temperature of the space, each fan coil has two exhausts; 4) temperature sensors. Figure 46 depicts the considered space inside the KUBIK building.

Figure 46: Living room area of KUBIK. Source TECNALIA

Figure 47 shows the total space of the KUBIK's ground floor, as well as the total considered space to create the experimental platform and run the AI algorithms. The KUBIK's ground floor has a height of 2.388 m, the total size of the considered area (in red) has a surface of 38.7m^2. Values from A to H shows the distance between the edges of each wall.

To build the thermal model of the space, the properties of KUBIK have been considered. First the thermal exchange with the **external wall** of the building were considered. The outdoor wall of the area is covered by windows almost from floor to ceiling. Taking this into account, the total external surface of the model has been considered to be window.

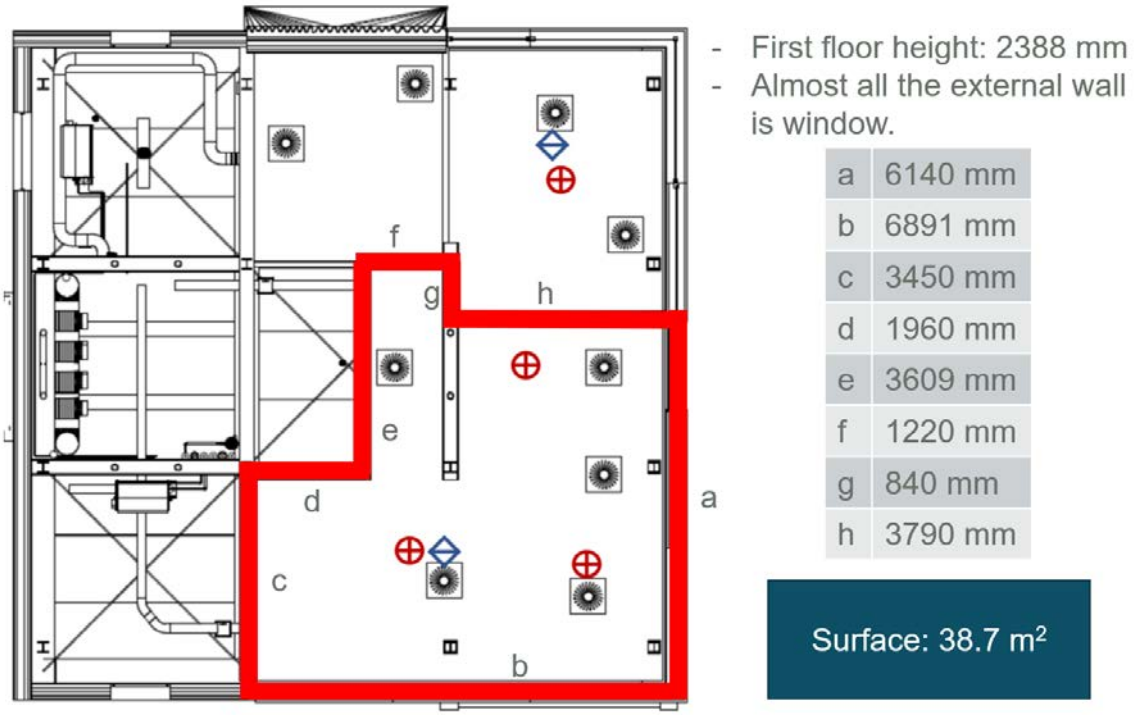


Figure 47: Surface information of the considered area. Source TECNALIA

The considered thermal properties of KUBIK's windows are close to the configuration 6/12/6 found in the standards where the thickness is 24 mm and the U-value is 1.7 W/(m^2 * K).[7]

Because the air diffusion in the room was not considered, the total air flow of both **fan coil units** (unit a and b) was considered. These fan coil units can operate in two different modes: 1) heating mode, in which the unit gives heat to the room; 2) cooling mode, in which the unit gives cool to cool the room. The given air flow is delimited in three operation modes levels: 1) minimum with gives an airflow of 260m^3/h; 2) medium which gives an airflow of 350m^3/h; 3) maximum which gives an airflow of 450 m^3/h. When the fan coil unit is in heating mode, the maximum air flow is 450m3/h and when is in cooling mode the maximum air flow is 350m3/h.

One of the problems encountered when the model was proposed, is that it is not possible to directly know the air temperature that the unit delivers to the room because the system does not measure it explicitly. For that reason, Tecnalia uses the fan coil datasheet and measure data to adjust the values and create a model which represents the reality. According to the given values, in cooling mode the fan coil circuit gives 12ºC, and in heating mode 50ºC. Based on this, Table 2 depicts the calculated value for each operational speed.

Air flow [m3/h]	Heating [ºC]	Cooling [ºC]
260	38	17
350	39	16.5
450	40	16

Table 2: FCU operation

[7] Thermal conductivity = U-value * thickness and R = 1 / (U-Value * Area)

To calculate the energy cost, we have considered a general assumption in which the different operation modes give different energy cost ratios. To calculate the energy cost of the fan coil unit operations, the following formula has been considered:

$$cooling\ cost = 1.1 * heating\ cost$$

Where heating cost are values given by the different operation modes: 1) minimum equals to 1/2; 2) medium equals to 3/4; 3) maximum equals to 1.

Finally, the dead band of the thermostat is assumed to be 1.8ºC.

Based on the assumption and values discussed above, the heat balance of the KUBIK building was modelled using following equations:

Equation 1

$$\frac{dQ_{FCU}}{dt} = (T_{FCU} - T_{room})\, M_{dot}\, c$$

Where:

- $\frac{dQ_{FCU}}{dt}$ is the heat/cool flow provided by the FCU to the room.
- c is the heat capacity of air at constant pressure.
- M_{dot} is the air mass flow rate through the fan coil unit.
- T_{FCU} is the temperature of the air getting out of the fan coil.
- T_{room} is the room air temperature.

Equation 2

$$\frac{dQ_{losses}}{dt} = A\ \frac{T_{room} - T_{out}}{R_{eq}}$$

Where:

- $\frac{dQ_{losses}}{dt}$ is the heat loss (or gain in cooling mode) through the external walls.
- $A\ is\ the\ surface\ of\ the\ external\ wall$
- T_{out} is the outdoor temperature.
- R_{eq} is the equivalent thermal resistance of the house.

Equation 1 represents the heat exchange between the fan coil unit and the room, when Equation 2 represents the heat loss/gain to/from the outdoor. The variation of the room temperature over time, can then be modelled by Equation 3

Equation 3

$$\frac{dT_{room}}{dt} = \frac{1}{M_{air}\, c}\left(\frac{dQ_{FCU}}{dt} - \frac{dQ_{losses}}{dt}\right)$$

Where:

- M_{air} is the mass of air in the room.

This model has been used to test and validate the Online Learning Enabler as described in D3.3. The following lines contain a summary of the given results by UDE in the experiments exposed in D3.3 where the above formulas were used to create the experimental workbench to train their Reinforcement Learning (RL) algorithms, and the KUBIK's data (e.g. outdoor temperature) was used to check if the given results are real enough.

Figure 48 shows the averaged result of five experiments where a policy-based RL algorithm called "Proximal Policy Optimization" (PPO) is applied. It can be seen that the algorithm is capable of learning

an adequate control policy to control the indoor temperature. During the first 250.000 timesteps the algorithm learns that it needs to keep the indoor temperature within the setpoint boundaries when a person is present in the room in order to maximize its cumulative reward and to avoid heating or cooling actions if no person is present in the room. This can be interpreted from the increasing reward curve. After around 250.000 timesteps the reward converges which results from the algorithm now being able to properly perform the heating and cooling actions, what can be seen from the periodic spike in the indoor temperature. These spikes result from the algorithm only taking the NOP (No OPeration) action (0) if the room is not occupied. This leads to an increase or decrease of the indoor temperature depending on the current outdoor temperature (during the winter period starting from timestep 400.000 these spikes become more visible in the plot).

Furthermore, the agent is able to proactively perform heating or cooling actions in such a way that the indoor temperature is already within the setpoint boundaries of +/- 1°C when a person is in the room, based on the predicted occupancy variable (non-occupancy phases 3, 4 & 5). This can be seen in Figure 49, which is an excerpt of the experiment showing the behaviour throughout one single week, after learning has converged (and without averaging the single variables). However, as the isolation of the KUBIK building is pretty efficient, the temperature is not falling below 18°C (in the excerpt). This makes no extensive preheating necessary, as only one minute of heating already increases the indoor temperature by approximately 0.5°C. It is important to note that the excerpt is based on the data of a single experiment.

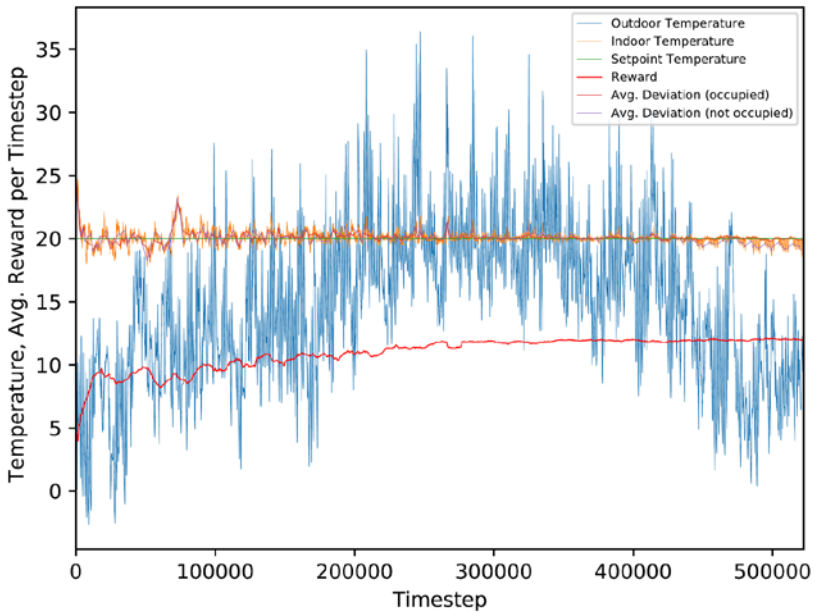


Figure 48: Evolution of temperature & reward averaged over five experiments

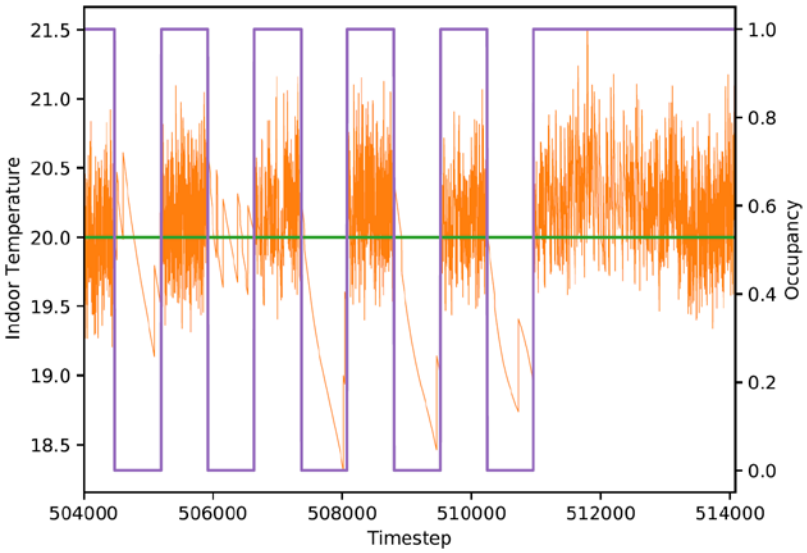


Figure 49: Excerpt of one week showing the proactive heating/cooling of a single experiment (seed: 463276947)

3.3.4 Software

The software development in the Smart Building use case for the evaluation of the ENACT enablers makes use of the SMOOL middleware. The SMOOL middleware has been adapted to the Smart Building domain by developing specific clients or KPs that represent the different sensor and actuator devices which contain distinct physical variables, e.g. temperature, humidity, luminosity…

The source code of the SMOOL middleware server and the SDK are located in a website that contains a lot of guidelines, presentation and example videos that ease the implementation of the middleware for the common user. Although this repository existed before the project, the SMOOL server and the code have been adapted to the new ENACT components and KPs specific of the Smart Building domain:
https://bitbucket.org/jasonjxm/smool/

The clients or KPs specifically developed for the ENACT project with the most common sensors and actuators for the Smart Building environment and the integration with ThingML are in the ENACT repository with Java examples for publishing data and subscribing to events for retrieving data. The data is created in the KUBIK smart building (temperature, humidity, luminosity, etc...). The current repository contains 4 Java projects:

- Producer: Publish data (temp, CO2, smoke...)
- Consumer: Subscribe to data and process when arriving
- Security: Control clients when sending data and actions.
- Monitor: Adding of lighting control to SMOOL.

https://gitlab.com/enact/smool_enact

The implementation of secure actuation in KUBIK building using the SMOOL middleware and the flows of Node-RED code to adapt IoT messages from devices, fan coils and meteo data to SMOOL middleware are available here:
https://gitlab.com/enact/kubik_enact

In the former repository, the code for a secure actuation integrated in SMOOL and tested in KUBIK can be found. This code is necessary to implement the GeneSIS and the Security and Privacy Monitoring and Control Enablers. Two KPs, consumer and producer of SMOOL, have been developed for this purpose using a secure token. Also, in the repository the Node-RED code that serves as bridge between sensor/actuator data and SMOOL is accessible:

- **SMOOL_KP_EnactKubik**: EnactKubik KP, adaptation of the SMOOL producer to make secure actuation.
- **SMOOL_KP_EnactKubikConsumer**: EnactKubikConsumer KP, adaptation of the SMOOL consumer to make secure actuation
- **Node-RED_code**: Code to adapt IoT messages from Z-Wave devices, fan coils and meteo data to the SMOOL middleware

The applications developed in the context of the complementary CNRS experimental lab (based on a network with devices from both KUBIK and the lab via SMOOL) are available in the following repository:

- https://gitlab.com/enact/smarthome-demo

3.3.5 Outlook and further work

3.3.5.1 GeneSIS, Actuation Conflict Management and Behavioural Drift Analysis enablers

The Actuation Conflict Management Enabler is very important for the Smart Building use case and the KUBIK in particular as a laboratory for testing the behaviour of complete IoT systems interacting with each other. Nowadays, the ACM is an Enabler used in the design phase and an evolution could be made to adapt it to use it for the operation phase as well. That could be used to fix the most complex actuation conflicts that occur when acting over a physical parameter of the building, e.g. when controlling the indoor temperature. For example, the effect on temperature control when opening a window is not the same in winter than in summer, its dependency on the current outdoor temperature, wind speed, airstreams… makes its effect quite different. In the design phase, the risk of a conflict actuation can be detected when simultaneously running an application that controls an electric heater and another application that ventilates the room by opening the window, but its real effect can only be quantified in operation time.

For the Behavioural Drift Analysis enabler, we envision as future work monitor more context information in more applications related to the Smart Building domain. This will provide more confidence on the IoT applications running correctly and that the introduction of new applications does not hinder the operation of the previous applications. In our opinion, we foresee a convergence between ACM and BDA Enablers in a single tool that works both in design and operating time. These two tools benefits from the integration with GeneSIS as a mean to integrate them as part of a DevOps process and to facilitate the deployment of monitoring probes for BDA and of actuation conflict managers for ACM. Finally, as a suggestion, specifying the behavioural model is something that would be nice to simplify in a future step to facilitate its implementation.

3.3.5.2 Security and Privacy Monitoring and Adaptation Enabler

All the security architecture implemented for the use case is expected to benefit not only the system 1 and 2 deployed in the building, but also other IoT systems and applications that in the future will share the KUBIK infrastructure and that will be monitored and controlled by ENACT S&P Mon&Control tools.

Together with the future exploitation of the KUBIK to support the research and implementation of an increasing number of smart building and home automation services (by both TECNALIA and their customers), it is expected that the features offered by the S&P Mon&Control Enabler will be enhanced so as the tool capabilities are enriched. This way, monitoring sensors tailored to new protocols data capturing and storage may be developed and deployed together with the existing ones and the necessary smart detection capabilities will be added on top of the current ones. This will enable the holistic monitoring of all the co-existing applications in the building and prevent attacks and anomalies occurring over the different systems' assets.

This will be possible thanks to the modular and highly scalable design of the ENACT S&P Mon&Control Enabler, that is based on the distributed architecture of the sensors deployment and the combination of their data to perform attacks and issues identification based on the application of AI-based detection algorithms. As more and more monitoring agents are deployed and more data are captured, the accuracy of the results obtained by the machine learning detection models of the Enabler will increase. This will have a direct impact of increasing the protection efficiency of the Enabler applied to all the IoT systems in the building.

With regards to the security control through SMOOL IoT platform used, the new design of SMOOL KPs (in the smart things) and the SMOOL Server enables that different security policies and types of security checks and controls are applied over the IoT applications co-existing in the building and connected to SMOOL, allowing different security configurations be adopted according to the application needs. This will perfectly fit with the different options that may be required by Tecnalia customers sharing the KUBIK facilities.

3.3.5.3 Online Learning Enabler

The application of the OLE inside the KUBIK building as a use case demonstrates how AI algorithms are capable of performing automatic decisions through AI techniques. The creation of an intelligent environment through a simulation engine to train and test the developed AI algorithms opens new key challenges in which an AI is capable of controlling the different comfort elements of a building. In fact, one of the best applications to improve this work, is the adaption of this algorithm sin legacy systems, which needs to be digitalized in order to create intelligent environments in old buildings.

However, unless there are several attempts to automate the comfort elements of a building to save energy and improve the comfort of the occupants, several things need to be addressed to improve the implementation of AI techniques. First of all, the used thermal model of the KUBIK building needs to be improved. The given formulas are accurate, but they are based in different assumptions such as the consideration of the area is a square or the different walls have the same properties. An improved modelisation of each element inside the building needs to be done in order to obtain an accurate thermal behaviour of the building elements.

Secondly, the used formulas need to be improved in order to obtain better results of the model itself. The addition of the fan coil unit operational equations will enhance the model and will allow to understand how the thermal variations inside the affected room are affected.

4 Conclusion

In the final stage of the case studies implementation, the use case provider partners focused on developing trustworthy and ready-to-use scenarios to test the DevOps Enablers and the final validation of the integrated ENACT DevOps framework. An initial group of DevOps Enablers have been tested and the works for testing and validating the Enablers under development have been initiated.

From the ITS use case perspective, the implementation part is considered enough to start a further implementation with the Enabler tools. This can be probed as first approaches are taken with three ENACT tools with no compatibility issues. It must be highlighted that the system is designed to provide the maximum possible flexibility into a highly safe and secure environment. Further improvements into the functionalities are performed to enable the integration with the rest of the planned tools for the ITS use case.

In the eHealth use case a main effort has been on implementing the new remote patient monitoring system and to investigate and learn how to be able to apply DevOps principles for its operation and evolution based on ENACT concepts and state of the art tools. This has resulted in a reference

implementation where trustworthy DevOps is partly supported based on ENACT concepts and tooling (e.g., ThingML, Risk Management concepts, Context-aware access control etc) as well as some state of the art DevOps tools such as Ansible, Jenkins, Jira, PagerDuty, ZenDesk, Grafana and Prometheus. This reference implementation will be the baseline when integrating and validating further the ENACT DevOps enablers in the next phase of the project.

In the Smart Building use case, the implementation done in KUBIK building comprises two different systems that interoperate using the Interoperability SMOOL IoT middleware and the Building Management System or BMS. These two differentiated systems are the IoT Smart Space (system 1) composed by new wireless IoT devices that communicate mainly using the Z-Wave protocol and the Building Control Space (system 2) composed by the legacy building control systems of the building that communicate using building automation protocols. This architecture enables the operation of complex Smart Energy Efficiency and Smart User Comfort applications to test the Enablers developed in the project and validate the ENACT DevOps framework.