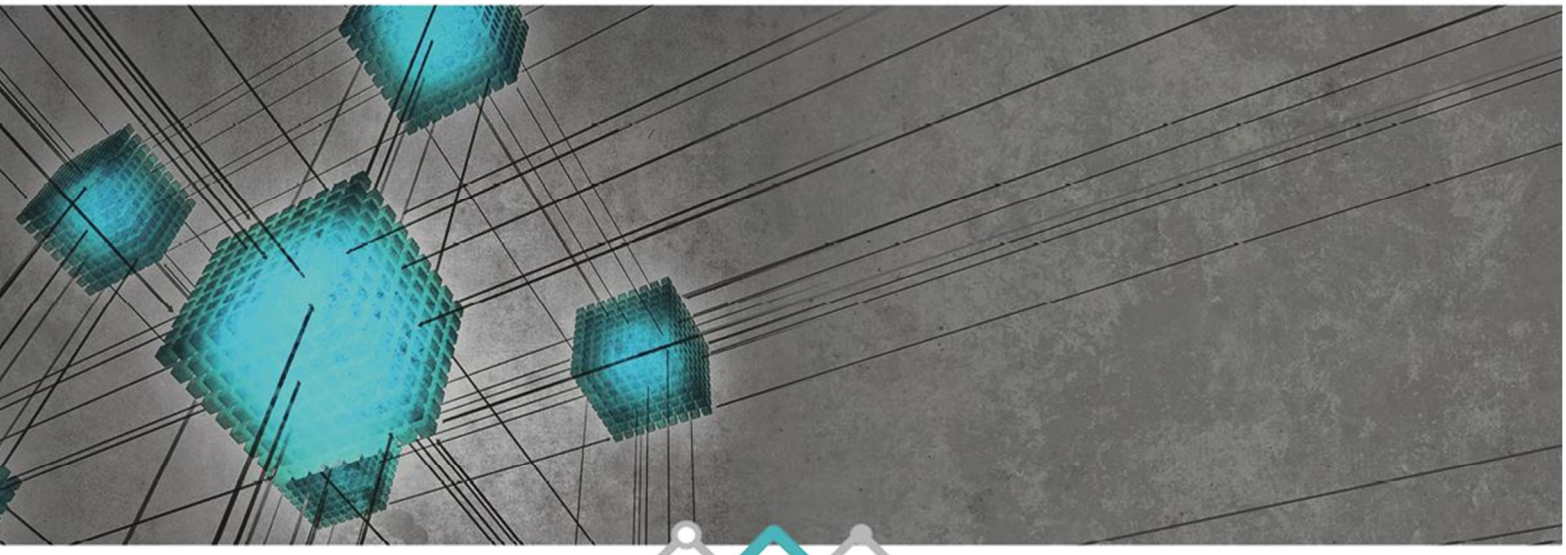

interiot

INTEROPERABILITY
OF HETEREOGENEUS
IOT PLATFORMS.

D6.3 - Appendix

Site Acceptance Test Plan - FAT outcome

Version: 1.0 - Final

February 2019

INTER-IoT

Site Acceptance Test Plan - FAT outcome

Version: 1.0

Security: Confidential

20 February 2019

The INTER-IoT project has been financed by the Horizon 2020 initiative of the European Commission, contract 687283

Disclaimer

Executive Summary

This document provides the overview of the FAT test outcome and is an appendix to the D6.3 deliverable.

List of Authors

Organisation	Authors	Main organisations' contributions
Neways	Dennis Engbers	Document template, structure, Executive summary and edit of the final document
VPF	Pablo Giménez	Provided FAT results
UPV-SABIEN	Álvaro Fides Valero	Provided FAT results
CEA	Jander Nascimento	Provided FAT results
VUB	Kris Steenhaut,	Provided FAT results
VUB	Benjamin Sartori	Provided FAT results
UPF	Toni Adame	Provided FAT results
UPF	Albert Bel	Provided FAT results
VPF	Pablo Giménez	Provided FAT results
Nemergent	Iñigo Ruiz	Provided FAT results
Nemergent	Jose Oscar Fajardo	Provided FAT results
University of Twente	João Moreira	Provided FAT results
Irideon	Bastian Faulhaber	Provided FAT results
INFOLYSiS	Vaios Koumaras	Provided FAT results
INFOLYSiS	Christos Sakkas	Provided FAT results
AUEB	Nikos Fotiou	Provided FAT results
AUEB	George C. Polyzos	Provided FAT results
TU Wien	Hong-Linh Truong	Provided FAT results
TU Wien	Bunjamin Memishi	Provided FAT results
TU Wien	Lingfan Gao	Provided FAT results
TU Wien	Michael Hammerer	Provided FAT results
CNR-ITIA	Gianfranco Modoni	Provided FAT results
CNR-ITIA	Enrico Caldarola	Provided FAT results
E3tcity	Javier Escalera Casillas	Provided FAT results

Change control datasheet

Version	Changes	Chapters	Pages
1.0	Creation, structure and released document	All	306

Contents

Executive Summary 4

List of Authors 5
Change control datasheet 5
Contents 5
List of Figures 7
List of Tables 7
Acronyms 8
1 Factory Acceptance Test Outcome 10
1.1 INTER-LogP SAT 10
1.1.1 Test overview 10
1.1.2 Outcome overview 10
1.2 INTER-Health SAT 11
1.2.1 Test outcome 11
1.2.2 Outcome overview 15
1.3 Open Call SAT's 16
1.3.1 Third Party: SensiNact 16
1.3.2 Third Party: INTER-HARE 19
1.3.3 Third Party: OM2M 30
1.3.4 Third Party: Mission Critical operations based on IoT analytics 32
1.3.5 Third Party: Early Warning System (EWS) 38
1.3.6 Third Party: Senshook 50
1.3.7 Third Party: SOFOS 55
1.3.8 Third Party: ACHILLES 57
1.3.9 Third Party: Inter-HINC 63
1.3.10 Third Party: Semantic Middleware 64
1.3.11 Third Party: SecurIoTy 75
1.3.12 Third Party: E3City 76

List of Figures

Figure 1. Workflow of the scenario 64

List of Tables

Table 1: INTER-Log-P Test outcome overview 10
Table 2: INTER-Health test outcome overview 15
Table 3: SensiNact test outcome overview 18
Table 4: INTER-HARE test outcome overview 29
Table 5: OM2M test outcome overview 31
Table 6: Mission Critical operations test outcome overview 37
Table 7: Early Warning System test outcome overview 49
Table 8: Senshook test outcome overview 54
Table 9: SOFOS test outcome template 56
Table 10: ACHILLES test outcome overview. 62
Table 11: Semantic Middleware test outcome overview 74
Table 12: E3City test outcome overview 77

Acronyms

AIOTI	Alliance for Internet of Things Innovation
API	Application Programming Interface
BDD	Behaviour Driven Development
CCB	Change Control Board
CNR-ITIA	National Research Council - Institute of Industrial Technologies and Automation
CSV	Comma-Separated Values
DMZ	Demilitarized zone
EC	European Commission
ESB	Enterprise service bus
FAT	Factory Acceptance Test
GCP	Google Cloud Platform
GDPR	General Data Protection Regulation
GOIoTP	Generic Ontology for IoT Platforms
ICT	Information and Communication Technology
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
IoT-EPI	IoT European Platform Initiative
IPR	Intellectual property rights
JSON	JavaScript Object Notation
JSONoWS	JSON over WebSockets
LAB setup	Test setup in a controlled environment at the developer site
LPLAN	Low Power Local Area Network
LPWAN	Low Power Wide Area Network
LTL	Less than Truck Load
MC	Mission Critical
MC-IoT	Mission Critical Internet of Things
MCPTT	Mission Critical Push To Talk
MEP	MQTT Event publisher
MiCrOBIoTa	Mission Critical operations based on IoT analytics
MQTT	MQ Telemetry Transport
MS	Microsoft
MSK	Master Secret Key

MSSB	MS Service Broker
MW2MW	Middleware to Middleware
NFV	Network Function Virtualization
oBIX	Open Building Information eXchange
OM2M	open source implementation of oneM2M in Eclipse
OS	Operating System
OSGi	Open Services Gateway initiative
PDR	Packet Delivery Ratio
PIR	Passive Infrared Sensor
PMR	Professional Mobile Radio
PRM	Power Regulation Mechanism
REST	Representational State Transfer
SAT	Site Acceptance Test
SDN	Software Defined Networking
SSBEAS	MS SQL Service Broker External Activator Service
STA	Station
TDD	Test Driven Development
TDMA	Time Division Multiple Access
TED	Transducer Electronic Data Sheet
TRL	Technology Readiness Level
UPF	Universitat Pompeu Fabra
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VPF	Valenciaport Foundation
XML	eXtended Markup Language

1 Factory Acceptance Test Outcome

1.1 INTER-LogP SAT

1.1.1 Test overview

1.1.1.1 IoT access control, traffic and operational assistance

T1.1 Truck triggers information

ID	T1.1
Test	Verify the integration of all the components in the IoT access control, traffic and operational assistance pilot. The main objective in the defined pilot is a service to control access, monitor traffic and assist the operations at the port.
Setup	Deployment, installation and configuration of all the components.
Start	A truck is arriving to the port.
Req.	[27], [28], [166], [194], [195], [198], [245], [246], [268]
Input	Truck data
Output	Exchange of access data between the port and the terminal
Logs	INTER-LogP1.1.log
Outcome	**Pass** / Fail

T1.2 Pilot Dynamic lighting

ID	T1.2
Test	Verify the integration of all the components in the Idynamic lighting pilot. The goal of this pilot is develop a smart illumination (Dynamic Illumination) in the yard of Noatum for the rail yard.
Setup	Deployment, installation and configuration of all the components.
Start	A truck or machinery is accessing to the rail yard area in the terminal.
Req.	[27], [28], [168], [198], [245]
Input	Data from PIR sensors
Output	The light level in the rail yard terminal is adjusted to the operation.
Logs	INTER-LogP1.2.log
Outcome	**Pass** / Fail

1.1.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T1.1	Truck triggers information	**Pass** / Fail
T1.2	Pilot Dynamic lighting	**Pass** / Fail
FAT Outcome		**Pass**

Table 1: INTER-Log-P Test outcome overview

1.2 INTER-Health SAT

1.2.1 Test outcome

1.2.1.1 Scenario 1, 15, 16, 21, 22, 23, 24, 25

1.2.1.1.1 U1 – Creates and operates users/services

T1.1.1 Professional creates user

ID	T1.1.1
Test	The Healthcare Professional enters the system and creates a new patient user
Type	Manual test
Setup	TS_01, TT_01
Start	First time run, empty records, Patient has been given phone and documentation
Req.	[62], [71], [103], [104], [145], [146], [158], [173], [174]
Input	Healthcare Prof. enters PWT, authenticates. Healthcare Prof. enters option to create new patient Healthcare Prof. enters patient data and confirms creation
Output	The new Patient is registered in the system The Patient can now use the system as intended
Logs	C:\PWT\Visor Logs Tailblazer_v0.9.0.536\logs C:\PWT\Daemons C:\inetpub\Web\InterIoT\Logs
Outcome	**Pass** / Fail

T1.1.2 Professional modifies user

ID	T1.1.2
Test	The Healthcare Professional enters the system and updates a patient's data
Type	Manual test
Setup	TS_01, TT_01
Start	Patient was already created in the system, Patient has been given phone and documentation
Req.	[62], [71], [103], [104], [145], [146], [158], [173], [174]
Input	Healthcare Prof. enters PWT, authenticates. Healthcare Prof. enters option to update patient Healthcare Prof. enters new patient data and confirms update
Output	The Patient's new data is registered in the system The Patient can continue to use the system as usual
Logs	C:\PWT\Visor Logs Tailblazer_v0.9.0.536\logs C:\PWT\Daemons C:\inetpub\Web\InterIoT\Logs
Outcome	**Pass** / Fail

T1.1.3 Patient logs with their profile

ID	T1.1.3
Test	Patient enters the system through his/her mobile phone apps to access data
Type	Manual test
Setup	TS_01, TT_01, TP_01, TP_02, TP_03
Start	Patient is in possession of mobile phone with installed app Patient has been registered in the system with proper data
Req.	[62], [71], [103], [104], [172], [176]
Input	Patient enters BC app, authenticates, checks data
Output	Patient successfully accesses app Patient can check up-to-date data
Logs	C:\PWT\Visor Logs Tailblazer_v0.9.0.536\logs C:\PWT\Daemons C:\inetpub\Web\InterIoT\Logs
Outcome	**Pass** / Fail

1.2.1.1.2 U2 – Set patient protocol parameters

T1.2.1 Professional sets protocol

ID	T1.2.1
Test	The Healthcare Professional enters the system and updates a patient's protocol
Type	Manual test
Setup	TS_01, TT_01
Start	Patient was already created in the system, Patient has been given phone and documentation
Req.	[62], [71], [103], [104], [145], [146], [158], [173], [174], [218]
Input	Healthcare Prof. enters PWT, authenticates. Healthcare Prof. enters option to update patient Healthcare Prof. enters new patient protocol and confirms update
Output	The Patient's new data is registered in the system The Patient can continue to use the system as usual, according to new protocol
Logs	C:\PWT\Visor Logs Tailblazer_v0.9.0.536\logs C:\PWT\Daemons C:\inetpub\Web\InterIoT\Logs
Outcome	**Pass** / Fail

1.2.1.1.3 U3 – Perform objective and subjective measures

T1.3.1 Professional collects measures (objective)

ID	T1.3.1
Test	Healthcare Professional takes measures from Patient at centre using devices
Type	Manual test
Setup	TS_01, TT_01, TH_01, TP_01, TP_02, TP_03
Start	Patient has been registered in the system with proper data Healthcare prof. is in possession of mobile phone with installed app Sensor devices are paired to mobile phone
Req.	[62], [71], [101], [102], [103], [104], [107], [127], [157], [164], [173], [177]
Input	Healthcare Prof. enters PWT, authenticates. Healthcare Prof. enters option to update patient measures Healthcare Prof. enters universAAL app, authenticates Patient takes measurement on centre sensor device
Output	Patient measure appears on uAAL app and PWT, allowing Healthcare Prof. to update patient measure data
Logs	C:\uaalserver\data\log
Outcome	**Pass** / Fail

T1.3.2 Patient performs measures (objective)

ID	T1.3.2
Test	Patient takes measures at home using devices
Type	Manual test
Setup	TS_01, TT_01, TH_01, TP_01, TP_02, TP_03
Start	Patient has been registered in the system with proper data Patient is in possession of mobile phone with installed app Sensor devices are paired to mobile phone
Req.	[62], [71], [101], [102], [103], [104], [107], [127], [157], [164], [172], [176], [217]
Input	Patient successfully accesses app Patient takes measurement on home sensor device
Output	The measure is registered in the system at the Healthcare centre and can be checked by Healthcare Prof. in PWT.
Logs	C:\tomcat\apache-tomcat-7.0.82\logs
Outcome	**Pass** / Fail

T1.3.3 Patient performs measures (subjective)

ID	T1.3.3
Test	Patient answers questionnaire about habits
Type	Manual test
Setup	TS_01, TT_01, TP_01, TP_02, TP_03
Start	Patient has been registered in the system with proper data Patient is in possession of mobile phone with installed app Healthcare Prof. has set protocol
Req.	[62], [71], [101], [102], [103], [104], [107], [157], [172], [218]
Input	App notifies Patient about questionnaire Patient successfully accesses app Patient takes questionnaire
Output	The measure is registered in the system at the Healthcare centre and can be checked by Healthcare Prof. in PWT.
Logs	C:\tomcat\apache-tomcat-7.0.82\logs
Outcome	**Pass** / Fail

1.2.1.1.4 U4 – Monitors subjective and objective parameters

T1.4.1 Professional monitors parameters (objective)

ID	T1.4.1
Test	Healthcare Professional accesses Patient data recorded through sensors
Type	Manual test
Setup	TS_01, TT_01, TH_01, TP_01, TP_02
Start	Patient has been registered in the system with proper data Patient has recorded objective measures Healthcare prof. has recorded objective measures of patient
Req.	[61], [71], [103], [104], [157], [173]
Input	Healthcare Prof. enters PWT, authenticates. Healthcare Prof. enters option to observe patient measures
Output	The PWT displays the measures taken with the sensors
Logs	C:\PWT\Visor Logs Tailblazer_v0.9.0.536\logs C:\PWT\Daemons C:\inetpub\Web\InterIoT\Logs
Outcome	**Pass** / Fail

T1.4.2 Professional monitors parameters (subjective)

ID	T1.4.2
Test	Healthcare Professional accesses Patient data recorded through questionnaires
Type	Manual test
Setup	TS_01, TT_01, TH_01, TP_01, TP_02
Start	Patient has been registered in the system with proper data Patient has recorded objective measures Healthcare prof. has recorded objective measures of patient
Req.	[61], [71], [103], [104], [157], [173], [218]
Input	Healthcare Prof. enters PWT, authenticates. Healthcare Prof. enters option to observe patient measures
Output	The PWT displays the measures taken with the questionnaires
Logs	C:\PWT\Visor Logs Tailblazer_v0.9.0.536\logs C:\PWT\Daemons C:\inetpub\Web\InterIoT\Logs
Outcome	**Pass** / Fail

1.2.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T1.1.1	Professional creates user	**Pass** / Fail
T1.1.2	Professional modifies user	**Pass** / Fail
T1.1.3	Patient logs with their profile	**Pass** / Fail
T1.2.1	Professional sets protocol	**Pass** / Fail
T1.3.1	Professional collects measures (objective)	**Pass** / Fail
T1.3.2	Patient performs measures (objective)	**Pass** / Fail
T1.3.3	Patient performs measures (subjective)	**Pass** / Fail
T1.4.1	Professional monitors parameters (objective)	**Pass** / Fail
T1.4.2	Professional monitors parameters (subjective)	**Pass** / Fail
FAT Outcome		**Pass**

Table 2: INTER-Health test outcome overview

1.3 Open Call SAT's

1.3.1 Third Party: SensiNact

1.3.1.1 Test overview

Test output log files… Folder "log", prefix "sensinact-`Tx.y.z.log`"

ID	T1.1
Test	Verify that the application uses only encryption channels to exchange with clients when fetching sensor data
Setup	TT **Error! Reference source not found.**, TT **Error! Reference source not found.**
Start	Activate module rest, simulated devices; start the platform, start Wireshark
Req.	R
Input	Retrieve the value of the current value of simulated button device
Output	Verify on Wireshark if any of the information sent can be seen by the sniffer.
Logs	sensinact-`T1.1.1.log`
Outcome	**Pass** / Fail

ID	T1.2
Test	Verify that the application uses only encryption channels to exchange with clients when activating an actuator
Setup	TT **Error! Reference source not found.**, TT **Error! Reference source not found.**
Start	Activate module rest, simulated devices; start the platform, start Wireshark
Req.	R
Input	Retrieve the value of the current value of simulated button device
Output	Verify on Wireshark if any of the information sent can be seen by the sniffer.
Logs	sensinact-`T1.2.1.log`
Outcome	**Pass** / Fail

ID	T1.3
Test	Verify that the application uses only encryption channels to exchange with clients when receiving a notification from the gateway
Setup	TT **Error! Reference source not found.**, TT **Error! Reference source not found.**, TT **Error! Reference source not found.**
Start	Activate module rest, simulated devices; start the platform, start Wireshark
Req.	R
Input	Retrieve the value of the current value of simulated button device
Output	Verify on Wireshark if any of the information sent can be seen by the sniffer.
Logs	sensinact-`T1.3.1.log`

Outcome	**Pass** / Fail

ID	T2.1
Test	Verify documentation availability on the website
Setup	TT **Error! Reference source not found.**
Start	Point the browser to URL indicated on the setup
Req.	R
Input	-
Output	Verify that the documentation is available
Logs	sensinact-`T2.1.1.log`
Outcome	**Pass** / Fail

ID	T3.1
Test	Verify that the code source is available on the website
Setup	TT **Error! Reference source not found.**
Start	Point the browser to URL indicated on the setup
Req.	R
Input	-
Output	Verify that the documentation is available
Logs	sensinact-`T3.1.1.log`
Outcome	**Pass** / Fail

ID	T4.1
Test	Verify that the REST API is available on the gateway
Setup	TT **Error! Reference source not found.**
Start	Point the browser to URL indicated on the setup
Req.	R
Input	-
Output	Verify that the documentation is available
Logs	sensinact-`T4.1.1.log`
Outcome	**Pass** / Fail

1.3.1.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T1.1	Verify that the application uses only encryption channels to exchange with clients when fetching sensor data	**Pass** / Fail
T1.2	Verify that the application uses only encryption channels to exchange with clients when activating an actuator	**Pass** / Fail
T1.3	Verify that the application uses only encryption channels to exchange with clients when receiving a notification from the gateway	**Pass** / Fail
T2.1	Verify documentation availability on the website	**Pass** / Fail
T3.1	Verify that the code source is available on the website	**Pass** / Fail
T4.1	Verify that the REST API is available on the gateway	**Pass** / Fail
FAT Outcome		**Pass**

Table 3: SensiNact test outcome overview

1.3.2 Third Party: INTER-HARE

1.3.2.1 Test outcome

1.3.2.1.1 S4 – Monitoring reefer containers

The objective of this scenario is to interoperate and use a shipping line's container IoT platform that is currently able to monitor reefer containers along its journey with the IoT platforms of the road hauliers or container terminals. This integration will allow a quick reaction in case of an alarm regarding the functioning of refrigerated goods and it will benefit container terminals and road haulier companies (drivers in this case) to avoid the periodic human inspection required for reefer containers.

Interoperability in this scenario is required to connect the shipping lines, the container terminals and the road hauliers IoT platforms.

The resulting service will be obtained by the integration of:

- Carrier IoT platform who is owner of the container
- Container terminal IoT platform
- Road haulier cloud IoT platform

In the following lines it is described how the port environment (i.e., reefer containers and container terminal) is emulated in an *ad-hoc* testbed located on UPF facilities and the list of considered use cases together with their associated tests. As described in Subsection **Error! Reference source not found.**, while the smallest rooms of our offices are considered as *reefer containers*, the *container terminal* has been located in a larger one.

Use case	Associated test
[19] User interacts with sensors or devices	T4.19.1 User interaction
[46] Device failure detection	T4.46.1 Resilience against failures
[60] Device registry	T4.60.1 Range coverage at 868 MHz
	T4.60.2 Range coverage at 2.4 GHz
	T4.60.3 Interference analysis at 2.4 GHz
	T4.60.4 (Physical) Gateway registration
	T4.60.5 Container registration
	T4.60.6 Sensor registration
	T4.60.7 Multiple sensor registration
[61] Platform Configuration on the Gateway	T4.61.1 Platform setup and simulation
[62] Device (sensor) triggers information	T4.62.1 Event-driven data delivery model test
	T4.62.2 Continuous data delivery model test
	T4.62.3 Hybrid data delivery model test
	T4.62.4 Data aggregation test
	T4.62.5 Relay operation test
[63] Platform requests information from a device (sensor)	T4.63.1 Query-driven data delivery model test (requests)
[64] Platform sends information to device (actuator)	T4.64.1 Query-driven data delivery model test (responses)

1.3.2.1.1.1 U19 – User interacts with sensors or devices

The whole platform is accessed remotely by the user, who can change some configuration settings. The user experience is analyzed.

T4.19.1 User interaction

ID	T4.19.1
Test	User experience analysis when configuring the INTER-HARE platform
Type	E. Integration network
Setup	Need test setup TS_08
Start	All DADs, CHs and the GW are already registered.
Req.	[80], [11], [25], [43], [243], [55], [138], [226], [244]
Input	Test hooks TH_02, TH_03, and TH_06
Output	Check system's ability to configure the parameters selected by the user.
Logs	E. Integration\T4.19.1_ux.txt
Outcome	**Pass** / Fail

1.3.2.1.2 U46 – Device failure detection

The system is able to detect problems in intermediate devices (CHs and DADs) and to act consequently, by reconstructing routing paths and ensuring data transmissions.

T4.46.1 Resilience against failures

ID	T4.46.1
Test	INTER-HARE resilience analysis against failures
Type	D. Resilience
Setup	Need test setup TS_07
Start	All DADs, CHs and the GW are already registered.
Req.	[6], [9], [7], [17], [153], [232], [233], [11], [20], [21], [22], [25], [26], [89], [56], [93], [57], [75], [204], [205], [206], [207], [72], [27], [28], [95], [242]
Input	Test hooks TH_02, TH_03, and TH_05
Output	Check if the system is able to rebuild routing routes after one (or more) DAD (or CH) switches off. Check if the system is able to maintain high reliability levels after one (or more) DAD (or CH) switches off.
Logs	D. Resilience\T4.46.1_resilience.txt
Outcome	**Pass** / Fail

1.3.2.1.3 U60 – Device registry

Devices are able to determine if they are within the range coverage of their immediate parent. If so, they execute the association process to be part of the network by receiving the corresponding network address and listening to the schedule beacons.

T4.60.1 Range coverage at 868 MHz

ID	T4.60.1
Test	Analysis of range coverage at 868 MHz
Type	A. Range coverage
Setup	Need test setup TS_01
Start	GW located in a fixed position. CH initially located close to the GW. CH is moved further from the GW.
Req.	[2], [39], [29]
Input	Test hook TH_01 in different CH positions
Output	CH inside or outside the range coverage of the GW.
Logs	A. Range\T4.60.1_868.txt
Outcome	**Pass** / Fail
	~850 m. (outdoor)

T4.60.2 Range coverage at 2.4 GHz

ID	T4.60.2
Test	Analysis of range coverage at 2.4 GHz
Type	A. Range coverage
Setup	Need test setup TS_02
Start	CH located in a fixed position. DAD initially located close to the CH. DAD is moved further from the CH.
Req.	[2], [29]
Input	Test hook TH_01 in different DAD positions
Output	DAD inside or outside the range coverage of the CH.
Logs	A. Range\T4.60.2_24.txt
Outcome	**Pass** / Fail
	100% of coverage in the indoor scenario

T4.60.3 Interference analysis at 2.4 GHz

ID	T4.60.3
Test	Evaluation of interference between two LPLANs
Type	A. Range coverage
Setup	Need test setup TS_03
Start	2 CHs located in different rooms (‘containers’). DAD initially located close to one CH. DAD is moved to the other CH.
Req.	[18], [19], [29]
Input	Test hook TH_01 in different DAD positions
Output	DAD receives a certain RSSI level from one or both CHs.
Logs	A. Range\T4.60.3_interference.txt
Outcome	**Pass** / Fail
	Roaming test completed

T4.60.4 (Physical) Gateway registration

ID	T4.60.4
Test	GW registration into the INTER-IoT network
Type	B. Association & Registration
Setup	Need test setup TS_04
Start	GW located in a fixed position.
Req.	[14], [15], [39], [45], [138], [244]
Input	Test hook TH_01
Output	Check proper GW registration into the INTER-IoT network.
Logs	B. Association\T4.60.4_gw_reg.txt
Outcome	**Pass** / Fail
	0 seconds

T4.60.5 Container registration

ID	T4.60.5
Test	CH (container) registration into the INTER-IoT network
Type	B. Association & Registration
Setup	Need test setup TS_04
Start	GW & CH located in a fixed position.
Req.	[2], [14], [39], [45], [138]
Input	Test hook TH_01
Output	Check proper CH registration into the INTER-IoT network.
Logs	B. Association\T4.60.5_ch_reg.txt
Outcome	**Pass** / Fail
	9.03 seconds

T4.60.6 Sensor registration

ID	T4.60.6
Test	DAD (sensor) registration into the INTER-IoT network
Type	B. Association & Registration
Setup	Need test setup TS_04
Start	GW, CH & DAD located in a fixed position.
Req.	[2], [14], [39], [45], [11], [22], [23], [16], [138]
Input	Test hook TH_01
Output	Check proper DAD registration into the INTER-IoT network.
Logs	B. Association\T4.60.6_dad_reg.txt
Outcome	**Pass (98.75%)** / Fail
	32.63 seconds

T4.60.7 Multiple sensor registration

ID	T4.60.7
Test	Multiple DAD (sensor) registration into the INTER-IoT network
Type	B. Association & Registration
Setup	Need test setup TS_07
Start	1 GW, 2 CHs & multiple DADs located in a fixed position.
Req.	[2], [6], [9], [14], [17], [45], [233], [11], [22], [23], [16], [138], [207], [242]
Input	Test hook TH_01
Output	Check proper CHs & DADs registration into the INTER-IoT network.
Logs	B. Association\T4.60.7_multiple_reg.txt
Outcome	**Pass (GW – 100%), Pass (CH – 100%), Pass (DAD – 95.75%)** / Fail
	GW – 0 seconds **CH – 12.35 seconds** **DAD – 52.11 seconds**

1.3.2.1.4 U61 – Platform Configuration on the Gateway

The whole platform is accessed remotely by the user. The user configures the system, activates the devices and controls the execution, even applying setup changes and/or sending specific requests to selected DADs.

T4.61.1 Platform setup and simulation

ID	T4.61.1
Test	INTER-HARE setup and full simulation
Type	E. Integration network
Setup	Need test setup TS_08
Start	All devices (1 GW, 2 CHs and multiple DADs) are located in a fixed position but are not associated to the network yet.
Req.	[17], [80], [11], [19], [20], [21], [22], [23], [25], [43], [243], [13], [16], [55], [138], [226], [72], [98], [242], [244]
Input	Test hooks TH_01, TH_02, TH_03, TH_05, and TH_06
Output	Check system's ability to configure the parameters selected by the user. Check correct transmission of packets from/to DADs according to the hybrid data delivery model.
Logs	E. Integration\T4.61.1_platform_conf.txt
Outcome	**Pass** / Fail

1.3.2.1.5 U62 – Device (sensor) triggers information

A device, typically a sensor, triggers an event sending determined information to the gateway in order to be stored in the platform.

T4.62.1 Event-driven data delivery model test

ID	T4.62.1	
Test	Performance analysis of event-driven traffic	
Type	C. Data transmission	
Setup	Need test setup TS_07	
Start	All DADs, CHs and the GW are already registered.	
Req.	[2], [6], [9], [153], [232], [233], [11], [20], [21], [25], [26], [56], [57], [204], [205], [206], [72], [27], [28], [242]	
Input	Test hooks TH_03 and TH_04	
Output	Check performance of different network metrics.	
Logs	C. Transmission\T4.62.1_event.txt	
Outcome		**Pass** / Fail
	PDR (%)	**Alarms 98.15%**
	Delay	**Alarms 11.77 seconds**
	Throughput	**Alarms 1.18 bps**
	PRM	**29 - 27**
	Energy consumption	**CPU (ON): 1.00%** **CPU (LPM): 98.00%** **RADIO (TX): 0.01%** **RADIO (RX): 6.00%** **RADIO (OFF): 93.99%**

T4.62.2 Continuous data delivery model test

ID	T4.62.2	
Test	Performance analysis of continuous traffic	
Type	C. Data transmission	
Setup	Need test setup TS_07	
Start	All DADs, CHs and the GW are already registered.	
Req.	[2], [6], [9], [153], [232], [233], [11], [20], [21], [56], [57], [75], [204], [206], [72], [27], [28], [95], [242]	
Input	Test hooks TH_02 and TH_04	
Output	Check performance of different network metrics.	
Logs	C. Transmission\T4.62.2_continuous.txt	
Outcome		**Pass** / Fail
	PDR (%)	**Alarms 97.71%**
	Delay	**Alarms 56.00 seconds**
	Throughput	**Alarms 5.21 bps**
	PRM	**29 – 21**
	Energy consumption	**CPU (ON): 1.00%** **CPU (LPM): 98.00%** **RADIO (TX): 0.01%** **RADIO (RX): 11.50%** **RADIO (OFF): 88.49%**

T4.62.3 Hybrid data delivery model test

ID	T4.62.3	
Test	Performance analysis of hybrid traffic	
Type	C. Data transmission	
Setup	Need test setup TS_07	
Start	All DADs, CHs and the GW are already registered.	
Req.	[2], [6], [9], [15], [80], [153], [232], [233], [11], [20], [21], [25], [26], [89], [56], [93], [57], [75], [204], [205], [206], [207], [72], [27], [28], [95], [242]	
Input	Test hooks TH_02, TH_03, and TH_04	
Output	Check performance of different network metrics.	
Logs	C. Transmission\T4.62.3_hybrid.txt	
Outcome		**Pass** / Fail
	PDR (%)	**Data – 89.17%** **Responses – 92.08%** **Alarms – 93.52%**
	Delay	**Data – 54.74 s.** **Responses – 20.22 s.** **Alarms – 12.07 s.**
	Throughput	**Data – 1.19 bps** **Responses – 1.23 bps** **Alarms – 1.12 bps**
	PRM	**29 - 27**
	Energy consumption	**CPU (ON): 1.00%** **CPU (LPM): 98.00%** **RADIO (TX): 0.01%** **RADIO (RX): 12.50%** **RADIO (OFF): 87.49%** **[Estimated lifetime = 13.28 days]**

T4.62.4 Data aggregation test

ID	T4.62.4	
Test	Performance analysis of data aggregation mechanism	
Type	C. Data transmission	
Setup	Need test setup TS_06	
Start	All DADs, CHs and the GW are already registered.	
Req.	[2], [6], [9], [153], [232], [233], [11], [20], [21], [25], [26], [89], [56], [93], [57], [75], [204], [205], [206], [207], [72], [27], [28], [95], [242]	
Input	Test hooks TH_02, TH_03, and TH_04	
Output	Check performance of different network metrics.	
Logs	C. Transmission\T4.62.4_aggregation.txt	
Outcome		**Pass** / Fail
	PDR (%)	**99.44%**
	Delay	**68.98 s.**
	Throughput	**1.99 bps**

	PRM	**30 - 26**
	Energy consumption	**CPU (ON): 1.00%** **CPU (LPM): 98.00%** **RADIO (TX): 0.01%** **RADIO (RX): 15.50%** **RADIO (OFF): 84.49%**

T4.62.5 Relay operation test

~~ID~~	~~T4.62.5~~
~~Test~~	~~Performance analysis of relay device & mechanism~~
~~Type~~	~~C. Data transmission~~
~~Setup~~	~~Need test setup TS_05~~
~~Start~~	~~All DADs, CHs and the GW are already registered.~~ ~~The Relay is activated.~~
~~Req.~~	~~[2], [6], [9], [153], [232], [233], [11], [20], [21], [25], [26], [89], [56], [93], [57], [75], [204], [205], [206], [207], [72], [27], [28], [95], [242]~~
~~Input~~	~~Test hooks TH_02, TH_03, and TH_04~~
~~Output~~	~~Check performance of different network metrics.~~
~~Logs~~	~~C. Transmission\T4.62.5_relay.txt~~
~~Outcome~~	~~Pass / Fail~~

1.3.2.1.6 U63 – Platform requests information from a device (sensor)

The user asks for data from a specific DAD.

T4.63.1 Query-driven data delivery model test (requests)

ID	T4.63.1	
Test	Performance analysis of query driven traffic (requests)	
Type	C. Data transmission	
Setup	Need test setup TS_07	
Start	All DADs, CHs and the GW are already registered.	
Req.	[2], [6], [9], [80], [153], [232], [233], [11], [20], [21], [25], [26], [56], [57], [204], [206], [72], [27], [28], [242]	
Input	Test hooks TH_03 and TH_04	
Output	Check performance of different network metrics.	
Logs	C. Transmission\T4.63.1_query_requests.txt	
Outcome		**Pass** / Fail
	PDR (%)	**Responses 94.58%**
	Delay	**Responses 14.52 s.**
	Throughput	**Responses 1.26 bps**
	PRM	**29 - 27**
	Energy consumption	**CPU (ON): 1.00%** **CPU (LPM): 98.00%** **RADIO (TX): 0.01%** **RADIO (RX): 6.00%** **RADIO (OFF): 93.99%**

1.3.2.1.7 U64 – Platform sends information to device (actuator)

Once the DAD receives the data request, it transmits the information through its pre-established path to the GW.

T4.64.1 Query-driven data delivery model test (responses)

ID	T4.64.1	
Test	Performance analysis of query driven traffic (responses)	
Type	C. Data transmission	
Setup	Need test setup TS_07	
Start	All DADs, CHs and the GW are already registered.	
Req.	[2], [6], [9], [80], [153], [232], [233], [11], [20], [21], [25], [26], [56], [57], [204], [206], [72], [27], [28], [242]	
Input	Test hooks TH_03 and TH_04	
Output	Check performance of different network metrics.	
Logs	C. Transmission\T4.64.1_query_responses.txt	
Outcome		**Pass** / Fail
	PDR (%)	**Responses 94.58%**
	Delay	**Responses 14.52 s.**
	Throughput	**Responses 1.26 bps**
	PRM	**29 - 27**
	Energy consumption	**CPU (ON): 1.00%** **CPU (LPM): 98.00%** **RADIO (TX): 0.01%** **RADIO (RX): 6.00%** **RADIO (OFF): 93.99%**

1.3.2.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Concept	Test	Description	Outcome	Value
A. Range coverage	T4.60.1	Range coverage at 868 MHz	Max. distance	~850 m. (outdoor)
	T4.60.2	Range coverage at 2.4 GHz	Max. distance	100% of coverage in the indoor scenario
	T4.60.3	Interference analysis at 2.4 GHz	Interference map	Roaming test completed
B. Association & Registration	T4.60.4	(Physical) Gateway registration	Pass / Fail	Pass (100%)
			Assoc. delay	0 s.
	T4.60.5	Container registration	Pass / Fail	Pass (100%)
			Assoc. delay	9.03 s.
	T4.60.6	Sensor registration	Pass / Fail	Pass (98.73%)
			Assoc. delay	32.63 s.
	T4.60.7	Multiple sensor registration	Pass / Fail	Pass (GW – 100%) (CH – 100%) (DAD – 95.75%)
			Assoc. delay	GW – 0 s. CH – 12.35 s.

				DAD – 52.11 s.
C. Data Transmission	T4.63.1 T4.64.1	Query-driven data delivery model test (requests and responses)	Pass / Fail	Pass
			PDR (%)	Responses 94.58%
			Delay	Responses 14.52 s.
			Throughput	Responses 1.26 bps
			PRM	29 - 27
			Energy consumption	CPU (ON): 1.00% CPU (LPM): 98.00% RADIO (TX): 0.01% RADIO (RX): 6.00% RADIO (OFF): 93.99%
	T4.62.1	Event-driven data delivery model test	Pass / Fail	Pass
			PDR (%)	Alarms 98.15%
			Delay	Alarms 11.77 s.
			Throughput	Alarms 1.18 bps
			PRM	29 - 27
			Energy consumption	CPU (ON): 1.00% CPU (LPM): 98.00% RADIO (TX): 0.01% RADIO (RX): 6.00% RADIO (OFF): 93.99%
	T4.62.2	Continuous data delivery model test	Pass / Fail	Pass
			PDR (%)	Data 97.71%
			Delay	Data 56.00 s.
			Throughput	Data 5.21 bps
			PRM	29 - 21
			Energy consumption	CPU (ON): 1.00% CPU (LPM): 98.00% RADIO (TX): 0.01% RADIO (RX): 11.50% RADIO (OFF): 88.49%
	T4.62.3	Hybrid data delivery model test	Pass / Fail	Pass
			PDR (%)	Data – 89.17% Responses – 92.08% Alarms – 93.52%
			Delay	Data – 54.74 s. Responses – 20.22 s. Alarms – 12.07 s.
			Throughput	Data – 1.19 bps Responses – 1.23 bps Alarms – 1.12 bps
			PRM	29 - 27
			Energy consumption	CPU (ON): 1.00% CPU (LPM): 98.00% RADIO (TX): 0.01% RADIO (RX): 12.50% RADIO (OFF): 87.49% *[Estimated lifetime = 13.28 days]*
	T4.62.4	Data aggregation test	Pass / Fail	Pass
			PDR (%)	Data

				99.44%
			Delay	68.98 s.
			Throughput	1.99 bps
			PRM	30 - 26
			Energy consumption	CPU (ON): 1.00% CPU (LPM): 98.00% RADIO (TX): 0.01% RADIO (RX): 15.50% RADIO (OFF): 84.49%
	T4.62.5	Relay operation test	Pass / Fail	-
			PDR (%)	-
			Delay	-
			Throughput	-
			PRM	-
			Energy consumption	-
D. Resilience	T4.46.1	Resilience against failures	Pass / Fail	Pass
E. Integration network	T4.19.1	User interaction	Pass / Fail	Pass
	T4.61.1	Platform setup and simulation	Pass / Fail	Pass
	FAT Outcome		**Pass / Fail**	**Pass**

Table 4: INTER-HARE test outcome overview

1.3.3 Third Party: OM2M

1.3.3.1 Test outcome

1.3.3.1.1 Scenario 1 - Use case 1

T1.1 Stand-alone bridge, syntactic and messaging/API integration, with Infrastructure Node running locally

ID	T1.1
Test	Data collection from inter-IoT
Type	Network communication
Setup	TS_01 Enabling IPE in OM2M IN Connection between computers running MW2MW and OM2M
Start	Creation request from MW2MW
Req.	[14], [26], [70], [127], [174], [234]
Input	Enable Graphical interface for interaction between end user and stored data Register through MW2MW REST API
Output	Check the HTTP messages exchanged between MW2MW and OM2M Check the creation of the resource in IN-CSE
Logs	Log of the IN-CSE and the OM2M bridge
Outcome	**Pass** / Fail

T1.2: Bridge with syntactic as well as semantic integration

~~ID~~	~~T1..2~~
~~**Test**~~	~~Data collection with semantic alignment from inter-IoT~~
~~**Type**~~	~~Network communication~~
~~**Setup**~~	~~TS_01~~ ~~Enabling IPE in OM2M IN~~ ~~Connection between computers running MW2MW and OM2M~~ ~~IPSM running with alignments between central ontology and OM2M~~
~~**Start**~~	~~Creation request from MW2MW~~
~~**Req.**~~	~~[14], [26], [70], [127], [174], [234]~~
~~**Input**~~	~~Enable Graphical interface for interaction between end user and stored data~~ ~~Register through MW2MW REST API, with alignments between central ontology and OM2M~~
~~**Output**~~	~~Check the HTTP messages exchanged between MW2MW and OM2M~~ ~~Check the creation of the resource in IN-CSE and the output of the alignment~~
~~**Logs**~~	~~Log of the IN-CSE, IPSM and the OM2M bridge~~
~~**Outcome**~~	~~Current work~~

1.3.3.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T1.1	Stand-alone bridge with Infrastructure Node running locally	**Pass** / Fail
~~T1.2~~	~~Bridge with syntactic as well as semantic integration~~	~~Pass / Fail~~
FAT Outcome		**Pass**

Table 5: OM2M test outcome overview

1.3.4 Third Party: Mission Critical operations based on IoT analytics

1.3.4.1 Test outcome

T1.1.1 Boot up and first contact

ID	T1.1.1
Test	Nemergent MC-IoT Module boots up and contacts INTER-IOT Platform
Type	System Testing
Setup	Need test setup TS_01
Start	Nemergent CtrlRoom is not launched
Req.	[47], [122]
Input	Launch the Jenkins job labelled as “T1.1.1”
Output	Check Jenkins job output, whether the job has succeeded or it has failed. Check outputted artifacts for debugging purposes.
Logs	Jenkins Job artifacts section: • backend_log.txt • frontend_log.txt • network_eth0.pcap • network_capture_logs.txt • test_report.html
Outcome	**Pass** / Fail

T1.1.2 INTER-FW Authentication

ID	T1.1.2
Test	Nemergent MC-IoT Module authenticates correctly against INTER-FW gateway.
Type	System Testing
Setup	Need test setup TS_01
Start	Nemergent CtrlRoom is already launched, but module has only made first contact.
Req.	[47], [122]
Input	Launch the Jenkins job labelled as “T1.1.2”
Output	Check Jenkins job output, whether the job has succeeded or it has failed. Check outputted artifacts for debugging purposes.
Logs	Jenkins Job artifacts section: • backend_log.txt • frontend_log.txt • network_eth0.pcap • network_capture_logs.txt • test_report.html
Outcome	**Pass** / Fail

T1.2.1 Obtain a list of trackable entities

ID	T1.2.1
Test	Nemergent MC-IoT Module queries INTER-FW for a list of authorised trackable entities.
Type	Service Discovery
Setup	Need test setup TS_01
Start	Nemergent CtrlRoom is properly launched and authenticated
Req.	[47], [53]
Input	Launch the Jenkins job labelled as “T1.2.1”
Output	Check Jenkins job output, whether the job has succeeded or it has failed. Check outputted artifacts for debugging purposes.
Logs	Jenkins Job artifacts section: • backend_log.txt • frontend_log.txt • network_eth0.pcap • network_capture_logs.txt • test_report.html
Outcome	**Pass** / Fail

T1.2.2 Paint all the trackable entities obtained

ID	T1.2.2
Test	Nemergent MC-IoT Module passes the processed data to the Nemergent CtrlRoom main module and paints the listed entities on a map.
Type	System Testing
Setup	Need test setup TS_01
Start	Nemergent CtrlRoom is properly launched and authenticated
Req.	[47], [53]
Input	Launch the Jenkins job labelled as “T1.2.2”
Output	Check Jenkins job output, whether the job has succeeded or it has failed. Check outputted artifacts for debugging purposes.
Logs	Jenkins Job artifacts section: • backend_log.txt • frontend_log.txt • network_eth0.pcap • network_capture_logs.txt • test_report.html
Outcome	**Pass** / Fail

T1.2.3 Subscribe to the event streams

ID	T1.2.3
Test	Nemergent MC-IoT Module performs subscription for the appropiate entities in order to maintain updated information about them, but without over saturating

	network and system resources.
Type	System Testing
Setup	Need test setup TS_01
Start	Nemergent CtrlRoom is properly launched and authenticated
Req.	[47], [51], [122]
Input	Launch the Jenkins job labelled as “T1.2.3”
Output	Check Jenkins job output, whether the job has succeeded or it has failed. Check outputted artifacts for debugging purposes.
Logs	Jenkins Job artifacts section: • backend_log.txt • frontend_log.txt • network_eth0.pcap • network_capture_logs.txt • test_report.html
Outcome	**Pass** / Fail

T1.2.4 Receive updates from INTER-FW

ID	T1.2.4
Test	Nemergent MC-IoT Module passes the processed data to the Nemergent CtrlRoom main module and repaints the listed entities on a map.
Type	System Testing
Setup	Need test setup TS_01
Start	Nemergent CtrlRoom is properly launched and authenticated
Req.	[47], [51], [122]
Input	Launch the Jenkins job labelled as “T1.2.4”
Output	Check Jenkins job output, whether the job has succeeded or it has failed. Check outputted artifacts for debugging purposes.
Logs	Jenkins Job artifacts section: • backend_log.txt • frontend_log.txt • network_eth0.pcap • network_capture_logs.txt • test_report.html
Outcome	**Pass** / Fail

T1.3.1 Receive detailed sensor data stream from INTER-FW

ID	T1.3.1
Test	Nemergent MC-IoT Module queries INTER-FW gateway for data streams specific to a device or sensor of an entity.
Type	System Testing
Setup	Need test setup TS_01

Start	Nemergent CtrlRoom is properly launched and authenticated
Req.	[47], [51], [122]
Input	Launch the Jenkins job labelled as “T1.3.1”
Output	Check Jenkins job output, whether the job has succeeded or it has failed. Check outputted artifacts for debugging purposes.
Logs	Jenkins Job artifacts section: • backend_log.txt • frontend_log.txt • network_eth0.pcap • network_capture_logs.txt • test_report.html
Outcome	**Pass** / Fail

T1.3.2 Receive detailed sensor updates from INTER-FW

ID	T1.3.2
Test	Nemergent MC-IoT Module passes the processed data to the Nemergent CtrlRoom main module and displays detailed sensor info (ECG, Speed, Temp) in a detailed view.
Type	System Testing
Setup	Need test setup TS_01
Start	Nemergent CtrlRoom is properly launched and authenticated
Req.	[47], [51], [122]
Input	Launch the Jenkins job labelled as “T1.3.2”
Output	Check Jenkins job output, whether the job has succeeded or it has failed. Check outputted artifacts for debugging purposes.
Logs	Jenkins Job artifacts section: • backend_log.txt • frontend_log.txt • network_eth0.pcap • network_capture_logs.txt • test_report.html
Outcome	**Pass** / Fail

T1.3.3 Stop receiving detailed sensor data stream from INTER-FW

ID	T1.3.3
Test	Nemergent MC-IoT Module queries INTER-FW gateway to unsubscribe from sensor data stream.
Type	System Testing
Setup	Need test setup TS_01
Start	Nemergent CtrlRoom is properly launched and authenticated
Req.	[47], [51], [122]

Input	Launch the Jenkins job labelled as “T1.3.3”
Output	Check Jenkins job output, whether the job has succeeded or it has failed. Check outputted artifacts for debugging purposes.
Logs	Jenkins Job artifacts section: • backend_log.txt • frontend_log.txt • network_eth0.pcap • network_capture_logs.txt • test_report.html
Outcome	**Pass** / Fail

1.3.4.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T1.1.1	Boot up and first contact	**Pass** / Fail
T1.1.2	INTER-FW Authentication	**Pass** / Fail
T1.2.1	Obtain a list of trackable entities	**Pass** / Fail
T1.2.2	Paint all of the trackable entities obtained	**Pass** / Fail
T1.2.3	Subscribe to the event streams	**Pass** / Fail
T1.2.4	Receive updates from INTER-FW	**Pass** / Fail
T1.3.1	Receive detailed sensor data stream from INTER-FW	**Pass** / Fail
T1.3.2	Receive detailed sensor updates from INTER-FW	**Pass** / Fail
T1.3.3	Stop receiving detailed sensor data stream from INTER-FW	**Pass** / Fail
FAT Outcome		**Pass**

Table 6: Mission Critical operations test outcome overview

1.3.5 Third Party: Early Warning System (EWS)

1.3.5.1 Test description

1.3.5.1.1 Scenario: accidents at the port area [id.9]

The functional goal of this scenario is to decrease the risk of fatal accidents at the port of Valencia, improving health prevention and enabling quick reaction by reducing time response.

The non-functional goal of this scenario is to exploit how e-Health and e-Care can use IoT platforms dedicated to logistics to prevent the occurrence of accidents and to support evacuation or attention in case of emergency situations [2]: “interoperate the wearable medical devices with IoT platforms (…) to react quickly, thus reducing time responses during accidents and health prevention” [INTER-IoT deliverable 2.4].

Interoperability in this scenario is required to connect the port authority (including emergency systems) and the road hauliers IoT platforms. The haulier solution is composed by two IoT platforms: one representing logistics data and one representing health data.

1.3.5.1.2 UC01: Vehicle collision detection

Monitor the truck’s location and detect possible collisions (impacts). In general, the approaches use an accelerometer within the vehicle to collect time series data about its location, i.e. the device’s acceleration about the corresponding axes (X, Y, Z), allowing the calculation of the G-Force felt in each instant. Then, for each instant, the detection mechanism compares if the G-Force is above a certain threshold, which is usually 3G for devices deployed in the vehicle chassis [16-19]. According to the patent for “vehicle security with accident notification and embedded driver analytics” (US 9491420 B2) [20], “instances of high acceleration/deceleration are due to a large change in velocity over a very short period of time. These speeds are hard to attain if a vehicle is not controlled by a human driver, which simplifies accident detection since we can assume any instance of high acceleration constitutes a collision involving human drivers”. An approach using a smartphone sharing accelerometer data is described in [21]. The Shimmer ECG 3 also provides accelerometer data, thus, it can also provide accelerometer data, providing an opportunity to integrate the health and logistics solutions.

Classification of severity and urgency according to accelerometer data (A) and threshold (B) is described in the table below. In summary, if the cross-axial energy computed is greater than the threshold and less than 20% above the threshold, then it might be a light collision (minor severity). If it is in-between 20% and 40%, then the collision is greater (moderate severity), if it is in-between 40% and 60%, then the collision is severe. Above 60% represents a strong impact, thus, an extreme severity, which probably needs immediate urgency for emergency response.

Range	Severity	Urgency
B < A <= B * 1.2	Minor	Expected
B * 1.2 < A <= B * 1.4	Moderate	Immediate
B * 1.4 < A <= B * 1.6	Severe	Immediate
B * 1.6 < A	Extreme	Immediate

Each test case has an equivalent input and output data file, named TX.Y.json (input and output folders). The type of all test cases here are system testing using scripted data.

This use case involves these requirements: [23], [72], [180], [249], [251].

T1.1 Detected with ECG device accelerometer, computed by

ID	T1.1
Test	Vehicle collision detected with ECG device accelerometer data computed by the smartphone, testing ST_UC01_01.
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [251]
Input	`T1_input/T1.1.json`: during a trip within the port area the accelerometer changes above the threshold, i.e. accelerometer value within an one trip point.
Output	`T1_output/T1.1.json`: EDXL-CAP
Logs	`T1_output/T1.1_logname`
Outcome	**Pass** / Fail

T1.2 Detected with medical wearable accelerometer

ID	T1.2
Test	Vehicle collision detected with smartphone accelerometer data.
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [251]
Input	`T1_input/T1.2.json`: during a trip within the port area the accelerometer changes above the threshold, i.e. accelerometer value within an one trip point.
Output	`T1_output/T1.2.json`: EDXL-CAP
Logs	`T1_output/T1.2_logname`
Outcome	**Pass** / Fail

T1.3 Detected with smartphone and medical wearable accelerometer

ID	T1.3
Test	Vehicle collision detected by using both accelerometer data within a window time.
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T1_input/T1.3_logistics.json`, `T1.3_health.json`: during a trip within the port area the accelerometer changes above the threshold, i.e. accelerometer value within an one trip point.
Output	`T1_output/T1.3.json`: EDXL-CAP
Logs	`T1_output/T1.3_logname`
Outcome	**Pass** / Fail

T1.4 Detected with smartphone or medical wearable accelerometer

ID	T1.4
Test	Vehicle collision detected through the rule that checks the battery consumption of the devices and decides which accelerometer data should be used.
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T1_input/T1.4_logistics.json, T1.4_health.json`: during a trip within the port area the accelerometer changes above the threshold, i.e. accelerometer value within an one trip point.
Output	`T1_output/T1.4.json`: EDXL-CAP
Logs	`T1_output/T1.4_logname`
Outcome	**Pass** / Fail

T1.5 Detected according to T1.1 with 4 classifications of severity + urgency

ID	T1.5
Test	Four executions of T1.1 resulting on the classifications of severity and urgency below (green, yellow, light red, dark red).
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [251]
Input	`T1_input/T1.5_level[1,2,3,4].json`: during a trip within the port area the accelerometer changes above the threshold according to the levels of urgency/severity (table XX).
Output	`T1_output/T1.5_level[1,2,3,4].json`: EDXL-CAP
Logs	`T1_output/T1.5_logname`
Outcome	**Pass** / Fail

T1.6 Detected according to T1.2 with 4 classifications of severity + urgency

ID	T1.6
Test	Four executions of T1.2 resulting on the classifications of severity and urgency above (green, yellow, light red, dark red).
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [251]
Input	`T1_input/T1.6_level[1,2,3,4].json`: during a trip within the port area the accelerometer changes above the threshold according to the levels of urgency/severity (table XX).
Output	`T1_output/T1.6_level[1,2,3,4].json`: EDXL-CAP

Logs	`T1_output/T1.6_logname`
Outcome	**Pass** / Fail

T1.7 Detected according to T1.3 with 4 classifications of severity + urgency

ID	T1.7
Test	Four executions of T1.3 resulting on the classifications of severity and urgency above (green, yellow, light red, dark red).
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T1_input/T1.7_level[1,2,3,4].json`: during a trip within the port area the accelerometer changes above the threshold according to the levels of urgency/severity (table XX).
Output	`T1_output/T1.7_level[1,2,3,4].json`: EDXL-CAP
Logs	`T1_output/T1.7_logname`
Outcome	**Pass** / Fail

T1.8 Detected according to T1.4 with 4 classifications of severity + urgency

ID	T1.8
Test	Four executions of T1.4 resulting on the classifications of severity and urgency above (green, yellow, light red, dark red).
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T1_input/T1.8_level[1,2,3,4].json`: during a trip within the port area the accelerometer changes above the threshold according to the levels of urgency/severity (table XX).
Output	`T1_output/T1.8_level[1,2,3,4].json`: EDXL-CAP
Logs	`T1_output/T1.8_logname`
Outcome	**Pass** / Fail

1.3.5.1.3 UC02: Hazardous health changes

Detect medical issues with the driver by monitoring his/her ECG and derived heart rate, checking possible cardiovascular emergencies. Cardiovascular emergencies are life-threatening disorders that must be recognized as soon as possible to minimize morbidity and mortality. By allowing the EWS to detect cardiovascular emergencies with trucks' drivers, it is possible to reduce the risk of an accident at the port area. The EWS provides messages that include the information of the cardiovascular emergency situation.

This can be achieved, basically, by using the the INTER-Health IoT solution with Shimmer ECG device attached to the driver's chest, wired to electrodes, and an Android-based mobile phone, both part of the Body module of the BodyCloud approach implemented with the SPINE framework. Thresholds used by the detection mechanism should be based on existing classifications to detect health risks. For example, target heart rates used for fitness is a classification of indicators that can be used as a baseline for thresholds. Figure XX illustrates such indicators (red, green, yellow, blue) depending on the person's age. Besides these thresholds, this use case also considers the situations which the driver presents bradycardia and tachycardia, which can be detected with the ECG device (event monitor) [1].

Classification of severity and urgency according to ComputeBPM output (A) and the threshold (B) is described in the table below. In summary, if the BPM calculated is greater than the threshold and less than threshold more 10%, then it might be a light tachycardia (minor severity). If it is in-between 10% and 20%, then the tachycardia is greater (moderate severity), if it is in-between 20% and 30%, then the tachycardia is severe. Greater than 30% represents a strong tachycardia, thus, an extreme severity, which probably needs immediate urgency for emergency response.

Range	Severity	Urgency
B < A <= B * 1.1	Minor	Expected
B * 1.1 < A <= B * 1.2	Moderate	Immediate
B * 1.2 < A <= B * 1.3	Severe	Immediate
B * 1.3 < A	Extreme	Immediate

Each test case has an equivalent input and output data file, named TX.Y.json (input and output folders). The type of all test cases here are system testing using scripted data.

This use case involves these requirements: [23], [72], [180], [249], [251].

T2.1 Bradycardia detected with fixed threshold

ID	T2.1
Test	From ECG data, the heart rate is calculated and compared to a threshold.
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T2_input/T2.1_level[1,2,3,4].json`: during a trip within the port area the heart rate is below the threshold according to the levels of urgency/severity (table XX).
Output	`T2_output/T2.1_level[1,2,3,4].json`: EDXL-CAP
Logs	`T2_output/T2.1_logname`
Outcome	**Pass** / Fail

[1] http://www.mayoclinic.org/diseases-conditions/bradycardia/diagnosis-treatment/diagnosis/dxc-20321665
http://www.mayoclinic.org/diseases-conditions/tachycardia/diagnosis-treatment/diagnosis/dxc-20253919

T2.2 Bradycardia detected with dynamic threshold

ID	T2.2
Test	From ECG data, the heart rate is calculated and compared to a threshold.
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T2_input/T2.2_level[1,2,3,4].json`: during a trip within the port area the heart rate is below the threshold according to the levels of urgency/severity (table XX).
Output	`T2_output/T2.2_level[1,2,3,4].json`: EDXL-CAP
Logs	`T2_output/T2.2_logname`
Outcome	**Pass** / Fail

T2.3 Tachycardia detected with fixed threshold

ID	T2.3
Test	From ECG data, the heart rate is calculated and compared to a threshold.
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T2_input/T2.3_level[1,2,3,4].json`: during a trip within the port area the heart rate is above the threshold according to the levels of urgency/severity (table XX).
Output	`T2_output/T2.3_level[1,2,3,4].json`: EDXL-CAP
Logs	`T2_output/T2.3_logname`
Outcome	**Pass** / Fail

T2.4 Tachycardia detected with dynamic threshold

ID	T2.4
Test	From ECG data, the heart rate is calculated and compared to a threshold.
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T2_input/T2.4_level[1,2,3,4].json`: during a trip within the port area the heart rate is above the threshold according to the levels of urgency/severity (table XX).
Output	`T2_output/T2.4_level[1,2,3,4].json`: EDXL-CAP
Logs	`T2_output/T2.4_logname`

Outcome	**Pass** / Fail

T2.5 Multiple occurrences of bradycardia detected with fixed threshold

ID	T2.5
Test	Several occurrences of T2.1 over a window of time (5 min).
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T2_input/T2.5_level[1,2,3,4].json`: during a trip within the port area the heart rate is below the threshold for a time window, according to the levels of urgency/severity (table XX).
Output	`T2_output/T2.5_level[1,2,3,4].json`: EDXL-CAP
Logs	`T2_output/T2.5_logname`
Outcome	**Pass** / Fail

T2.6 Multiple occurrences of bradycardia detected with dynamic threshold

ID	T2.6
Test	Several occurrences of T2.2 over a window of time (5 min).
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T2_input/T2.6_level[1,2,3,4].json`: during a trip within the port area the heart rate is below the threshold for a time window, according to the levels of urgency/severity (table XX).
Output	`T2_output/T2.6_level[1,2,3,4].json`: EDXL-CAP
Logs	`T2_output/T2.6_logname`
Outcome	**Pass** / Fail

T2.7 Multiple occurrences of tachycardia detected with fixed threshold

ID	T2.7
Test	Several occurrences of T2.3 over a window of time (5 min).
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T2_input/T2.7_level[1,2,3,4].json`: during a trip within the port area the heart rate is above the threshold for a time window, according to the levels of urgency/severity (table XX).

Output	`T2_output/T2.7_level[1,2,3,4].json`: EDXL-CAP
Logs	`T2_output/T2.7_logname`
Outcome	**Pass** / Fail

T2.8 Multiple occurrences of tachycardia detected with dynamic threshold

ID	T2.8
Test	Several occurrences of T2.4 over a window of time (5 min).
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T2_input/T2.8_level[1,2,3,4].json`: during a trip within the port area the heart rate is above the threshold for a time window, according to the levels of urgency/severity (table XX).
Output	`T2_output/T2.8_level[1,2,3,4].json`: EDXL-CAP
Logs	`T2_output/T2.8_logname`
Outcome	**Pass** / Fail

T2.9 Large variation of heart rate

ID	T2.9
Test	Detect whether variations occur. If there is a variation of more than 50% of the heart rates collected during a period of 5 minutes, then this situation is detected.
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]
Input	`T2_input/T2.9.json`: during a trip within the port area the heart rate suffers large variation.
Output	`T2_output/T2.9.json`: EDXL-CAP
Logs	`T2_output/T2.9_logname`
Outcome	**Pass** / Fail

T2.10 Detect high level of stress

ID	T2.10
Test	Based on UNICAL solution with Cardiac Defense Response (CDR).
Type	System testing using scripted data
Setup	Need test setup TS_01
Start	Vehicle is in one of the port gates (entering the port area).
Req.	[180], [249], [251]

Input	`T2_input/T2.10.json`: during a trip within the port area high-level of stress is detected.
Output	`T2_output/T2.10.json`: EDXL-CAP
Logs	`T2_output/T2.10_logname`
Outcome	**Pass** / Fail

1.3.5.1.4 UC03: Temporal relations (UC01 ~ UC02)

This use case exploits the possible temporal relations between UC01 and UC02 for detection of an accidents in the port area. For example, if a truck collision is detected from the accelerometers of the medical and mobile devices (T1.3) and right after (e.g. within 1-2 minutes) detecting large variation of heart rate (T2.9) then there is a high probability that a severe accident occurred, the driver is injured and he/she requires urgent medical help. Notice that the temporal relationship ("right after") is crucial to integrate these use cases.

T3.1 Vehicle collision followed by bradycardia

ID	T3.1
Test	Slow heart rate right after (within 2 minutes) a collision is detected can represent that an accident just occurred and the driver is probably injured.
Type	System testing
Start	
Req.	
Input	
Output	
Logs	
Outcome	**Pass** / Fail

1.3.5.1.5 UC04: Wrong-way driving

This use case exploits the possible temporal relations between UC01 and UC02 for detection of an accidents in the port area. For e

T4.1 Truck on opposite direction of a street within the port

ID	T4.1
Test	From the position data (mobile), the EWS will check the street direction and compare to the truck's position change within 30 seconds.
Type	System testing
Start	
Req.	
Input	
Output	
Logs	
Outcome	**Pass** / Fail

1.3.5.1.6 UC05: Accident involving dangerous goods

This use case will extend the use cases UC01-04 by checking whether dangerous goods are being transported, which will increase the situation urgency and severity and include the dangerous goods classification according to UNECE[2]. Data test will include simulation of trips including the transportation of class 1 (explosives), 3 (flammable liquids), 4 (flamed solids), 6 (toxic and infectious) and 7 (radioactive).

T5.1 UC01 with dangerous goods

ID	T5.1
Test	Tests of UC01 incremented with a check whether dangerous goods are being transported.
Type	System testing
Setup	
Start	
Req.	
Input	
Output	
Logs	
Outcome	**Pass** / Fail

T5.2 UC02 with dangerous goods

ID	T5.2
Test	Tests of UC01 incremented with a check whether dangerous goods are being transported.
Type	System testing
Setup	
Start	
Req.	
Input	
Output	
Logs	
Outcome	**Pass** / Fail

T5.3 UC03 with dangerous goods

ID	T5.3
Test	Tests of UC03 incremented with a check whether dangerous goods are being transported.
Type	System testing

[2] https://www.unece.org/fileadmin/DAM/trans/danger/publi/unrec/rev17/English/Rev17_Volume1.pdf

Setup	
Start	
Req.	
Input	
Output	
Logs	
Outcome	**Pass** / Fail

T5.4 UC04 with dangerous goods

ID	T5.4
Test	Tests of UC04 incremented with a check whether dangerous goods are being transported.
Type	System testing
Setup	
Start	
Req.	
Input	
Output	
Logs	
Outcome	**Pass** / Fail

1.3.5.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T1.1	Vehicle collision detected with smartphone accelerometer	**Pass** / Fail
T1.2	Vehicle collision detected with medical wearable accelerometer	**Pass** / Fail
T1.3	Vehicle collision detected with smartphone and medical wearable accelerometer	**Pass** / Fail
T1.4	Vehicle collision detected with smartphone or medical wearable accelerometer	**Pass** / Fail
T1.5	Vehicle collision detected according to T1.1 with 4 classifications of severity + urgency	**Pass** / Fail
T1.6	Vehicle collision detected according to T1.2 with 4 classifications of severity + urgency	**Pass** / Fail
T1.7	Vehicle collision detected according to T1.3 with 4 classifications of severity + urgency	**Pass** / Fail
T1.8	Vehicle collision detected according to T1.4 with 4 classifications of severity + urgency	**Pass** / Fail
T2.1	Bradycardia detected with fixed threshold	**Pass** / Fail
T2.2	Bradycardia detected with dynamic threshold	**Pass** / Fail

T2.3	Tachycardia detected with fixed threshold	**Pass** / Fail
T2.4	Tachycardia detected with dynamic threshold	**Pass** / Fail
T2.5	Multiple occurrences of bradycardia detected with fixed threshold	**Pass** / Fail
T2.6	Multiple occurrences of bradycardia detected with dynamic threshold	**Pass** / Fail
T2.7	Multiple occurrences of tachycardia detected with fixed threshold	**Pass** / Fail
T2.8	Multiple occurrences of tachycardia detected with dynamic threshold	**Pass** / Fail
T2.9	Large variation of heart rate	**Pass** / Fail
T2.10	Detect high level of stress	**Pass** / Fail
T3.1	Vehicle collision followed by bradycardia	**Pass** / Fail
T4.1	Truck on opposite direction of a street within the port	**Pass** / Fail
T5.1	UC01 with dangerous goods	**Pass** / Fail
T5.2	UC02 with dangerous goods	**Pass** / Fail
T5.3	UC03 with dangerous goods	**Pass** / Fail
T5.4	UC04 with dangerous goods	**Pass** / Fail
FAT Outcome		**Pass**

Table 7: Early Warning System test outcome overview

1.3.6 Third Party: Senshook

1.3.6.1 Test description

1.3.6.1.1 Scenario 15 Surveillance systems for prevention programs

T15.1.1 Device identification over Dispatcher API

ID	T15.1.1
Test	Device Identification over Dispatcher API
Type	System Testing
Setup	TT_01, TT_02, TS_01, TH_01, TP_01
Start	Device is not yet connected.
Req.	[243], [57], [21], [93], [15]
Input	• Connect device • Perform a call to the Dispatcher API method 'discoverTims'
Output	String containing the device ID
Logs	log/T15-1-1.log
Outcome	**Pass** / Fail

T15.1.2 Device identification over REST API

ID	T15.1.2
Test	Device Identification over REST API
Type	System Testing
Setup	TT_03, TS_02, TH_02
Start	Device is not yet connected.
Req.	[21], [26], [243], [93], [226], [14], [15], [39], [244]
Input	• Connect device • Perform a call to the REST API method 'discoverTims'
Output	String containing the device ID
Logs	-
Outcome	**Pass** / Fail

T15.1.3 Device identification over Web Interface

ID	T15.1.3
Test	Device Identification over Web Interface
Type	System Testing
Setup	<Describe the needed setup, tools, hooks and probes needed for this test>
Start	TS_03
Req.	[21], [26], [243], [93], [226], [14], [15], [39], [244]
Input	• Connect device • Click 'TIM Discovery' Button
Output	String containing the device ID

Logs	-
Outcome	**Pass** / Fail

1.3.6.1.2 Obtain information about connected sensors

Information about the connected sensors to the different Senscape devices is retrieved. Senscape implements the IEEE 1451.4 Transducer Electronic Data Sheets (TEDS) standard. Besides the ability to retrieve detailed information about the connected sensors it also provides plug and play functionality for sensors.

Currently there are three interfaces from which it is possible to read the TEDS, a Java/OSGI API, a REST API and a web interface.

T15.2.1 Read TEDs over Dispatcher API

ID	T15.2.1
Test	Read TEDs over Dispatcher API
Type	System Testing
Setup	TT_01, TT_02, TS_01, TH_01, TP_01
Start	Device connected
Req.	[243], [57], [21], [93], [15]
Input	• Perform a call to the Dispatcher API method ‘readTeds
Output	String containing the TEDs
Logs	log/T15-2-1.log
Outcome	**Pass** / Fail

T15.2.2 Read TEDs over REST API

ID	T15.2.2
Test	Read TEDs over REST API
Type	System Testing
Setup	TT_03, TS_02, TH_02
Start	Device connected
Req.	[21], [26], [243], [93], [226], [14], [15], [39], [244]
Input	• Perform a call to the REST API method ‘readTeds’
Output	String containing the TEDs
Logs	-
Outcome	**Pass** / Fail

T15.2.3 Read TEDs over Web Interface

ID	T15.2.3
Test	Read TEDs over Web Interface
Type	System Testing
Setup	TS_03
Start	Device connected
Req.	[21], [26], [243], [93], [226], [14], [15], [39], [244]

Input	• Fill out fields: TIM Id, Channel Id, TEDs type • Click on 'Read TEDs' button.
Output	String containing the TEDs
Logs	-
Outcome	**Pass** / Fail

1.3.6.1.3 Read sensor

The current value of a connected sensor is read.

Currently there are three interfaces from which it is possible to read the sensors, a Java/OSGI API, a REST API and a web interface.

T15.3.1 Read sensor data over Dispatcher API

ID	T15.3.1
Test	Read sensor data over Dispatcher API
Type	System Testing
Setup	TT_01, TT_02, TS_01, TH_01, TP_01
Start	Device connected
Req.	[243], [57], [21], [93], [15]
Input	• Perform a call to the Dispatcher API method 'readData'
Output	String containing the current value of the sensor
Logs	log/15-3-1.log
Outcome	**Pass** / Fail

T15.3.2 Read sensor data over REST API

ID	T15.3.2
Test	Read sensor data over REST API
Type	System Testing
Setup	TT_03, TS_02, TH_02
Start	[21], [26], [243], [93], [226], [14], [15], [39], [244]
Req.	[243], [57], [21], [93], [15]
Input	• Perform a call to the REST API method 'data'String containing the current value of the sensor
Output	String containing the current value of the sensor.
Logs	-
Outcome	**Pass** / Fail

T15.3.3 Read sensor data over Web Interface

ID	T15.3.3
Test	Read sensor data over Web Interface
Type	System Testing
Setup	TS_03
Start	Device connected
Req.	[21], [26], [243], [93], [226], [14], [15], [39], [244]
Input	• Fill out TIM Id and Channel Id field. • Click on 'Read Data' button.

Output	String containing the current value of the sensor.
Logs	-
Outcome	**Pass** / Fail

1.3.6.1.4 Send mosquito flight data to Data Storage

A mosquito entered the trap and the flight is detected and sent to the data storage. This can be done via the Java/OSGI API.

T15.4.1 Send mosquito flight data over Dispatcher API

ID	T15.4.1
Test	TT_01, TT_02, TS_01, TH_01, TP_01
Type	System Testing
Setup	<Describe the needed setup, tools, hooks and probes needed for this test>
Start	• Device connected • Flight data extracted
Req.	[15], [153], [243], [93]
Input	• Perform a call to the Dispatcher API method 'writeDataDb'
Output	-
Logs	log/15-4-1.log
Outcome	**Pass** / Fail

1.3.6.1.5 Retrieve flight data from the data base

The mosquito flight data is retreived from the data base for further use.

This can be done via the Java/OSGI API and the REST API.

T15.5.1 Retrieve flight data from the data base over Dispatcher API

ID	T15.5.1
Test	Retrieve flight data from the data base over Dispatcher API
Type	System Testing
Setup	TT_01, TT_02, TS_01, TH_01, TP_01
Start	Device connected
Req.	[153], [243]
Input	• Perform a call to the Dispatcher API method 'readLastValuesDb'
Output	Flight data results
Logs	Log/15-5-1.log
Outcome	**Pass** / Fail

T15.5.2 Retrieve flight data from the data base over REST API

ID	T15.5.2
Test	Retrieve flight data from the data base over REST API
Type	System Testing
Setup	TT_03, TS_02, TH_02

Start	Device connected
Req.	[14], [39], [153], [226], [244], [39], [244]
Input	• Perform a call to the REST API method 'readLastValuesDb'Flight data results
Output	Flight data results
Logs	-
Outcome	**Pass** / Fail

1.3.6.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T15.1.1	Device identification over Dispatcher API	**Pass** / Fail
T15.1.2	Device identification over REST API	**Pass** / Fail
T15.1.3	Device identification over Web Interface	**Pass** / Fail
T15.2.1	Read TEDs over Dispatcher API	**Pass** / Fail
T15.2.2	Read TEDs over REST API	**Pass** / Fail
T15.2.3	Read TEDs over Web Interface	**Pass** / Fail
T15.3.1	Read sensor data over Dispatcher API	**Pass** / Fail
T15.3.2	Read sensor data over REST API	**Pass** / Fail
T15.3.3	Read sensor data over Web Interface	**Pass** / Fail
T15.4.1	Send mosquito flight data over Dispatcher API	**Pass** / Fail
T15.5.1	Retrieve flight data from the data base over Dispatcher API	**Pass** / Fail
T15.5.2	Retrieve flight data from the data base over REST API	**Pass** / Fail
FAT Outcome		**Pass**

Table 8: Senshook test outcome overview

1.3.7 Third Party: SOFOS

1.3.7.1 Test outcome

S1 – Accident at the port area: Health monitoring system with passengers aboard a ferry

This use case involves these requirements: [15], [21], [70], [78], [227], [231].

T1 - MQTT mapping to UDP-based raw generated data

FAT Test	MQTT mapping to UDP-based raw generated data
Testing Type	VNF
Setup Components	MQTT generator, MQTT mapping VNF, Infolysis IoT GW, OpenVSwitch
Start	MQTT generator produces and sends data (random generation)
Req.	[15],[21],[70],[78],[227],[231]
Input	MQTT sample/raw data (random generation)
Output	UDP-based raw data
Outcome	**Pass** / Fail

T2 - CoAP mapping to UDP-based raw generated data

FAT Test	CoAP mapping to UDP-based raw generated data
Testing Type	VNF
Setup Components	CoAP generator, CoAP mapping VNF, Infolysis IoT GW, OpenVSwitch
Start	CoAP generator produces and sends data (random generation)
Req.	[15],[21],[70],[78],[227],[231]
Input	CoAP sample/raw data (random generation)
Output	UDP-based raw data
Outcome	**Pass** / Fail

T3 - HTTP mapping to UDP-based raw generated data

FAT Test	HTTP mapping to UDP-based raw generated data
Testing Type	VNF
Setup Components	HTTP generator, HTTP mapping VNF, Infolysis IoT GW, OpenVSwitch
Start	HTTP generator produces and sends data (random generation)
Req.	[15],[21],[70],[78],[227],[231]
Input	HTTP sample/raw data (random generation)
Output	UDP-based raw data
Outcome	**Pass** / Fail

1.3.7.2 Outcome overview

Test	Description	Outcome
T1	MQTT mapping to UDP-based raw data	**Pass** / Fail
T2	CoAP mapping to UDP-based raw data	**Pass** / Fail
T3	HTTP mapping to UDP-based raw data	**Pass** / Fail
FAT Outcome		**Pass**

Table 9: SOFOS test outcome template

1.3.8 Third Party: ACHILLES

1.3.8.1 Test description

1.3.8.1.1 S1 - IoT data sharing

The objective of this scenario is to enable a resource owner to share measurement data with other authorized users. In this scenario resource owners have Things they own connected to an Inter-IoT GW. These Things perform various measurements. Measurements are grouped based on the Thing location and can be accessed in real-time using the appropriate CoAP resource URIs (e.g., coap://window.bedroom.user1/temperature). Resource owners define access control policies in the ACP (e.g., “Friends”, “Family”) and define in the GW the access control policy that protects each group of measurements.

1.3.8.1.1.1 U1: New measurement group creation

The resource owner creates a new group of measurements and registers them in the GW, providing at the same time a pointer to the access control policy that protects them.

T1.1.1 New measurement group creation

ID	T1.1.1
Test	Registration of a new group of measurements
Type	System testing
Setup	Needs setup TS_01
Start	Access Table in GW is empty
Req.	[2],[6],[11],[22],[243],[98]
Input	Resource owner invokes the resource registration API call
Output	Access Table is updated
Logs	Folder “`T1_Output`”, prefix “`T1.1.1_achilles`” >
Outcome	**Pass** / Fail

1.3.8.1.1.2 U2- User request

A user is interested in a receiving a measurement protected under a specific access control policy. The user performs an initial request (an unauthorized request) to learn all information required for authorization. Then, it authenticates himself in the appropriate ACP and obtains an authorization token. The latter is used for performing an authorized request.

T1.2.1 Unauthorized request

ID	T1.2.1
Test	Request from an unauthorized user for a protected resource
Type	System Testing
Setup	Needs setup TS_01
Start	Access Table contains some entries
Req.	[27]
Input	A CoAP request from an unauthorized user
Output	• Check if the resource is included in the Access Table • Generate session key • Generate token

	• Respond to the user with the ACP URI and the token
Logs	Folder "`T1_Output`", prefix "`T1.2.1_achilles`" >
Outcome	**Pass** / Fail

T1.2.2 Authorized request

ID	T1.2.2
Test	Request from an authorized user for a protected resource
Type	System Testing
Setup	Needs setup TS_01
Start	Access Table and Token Table contains some entries
Req.	[14],[15],[72],[76]
Input	A CoAP request from an authorized user
Output	• Check if the resource is included in the Access Table • Check if the Token is included in the Token Table and it is still valid • Perform a CoAP request to the appropriate Thing • Encrypt the response and send it back to the user
Logs	Folder "`T1_Output`", prefix "`T1.2.2_achilles`" >
Outcome	**Pass** / Fail

1.3.8.1.2 S2 - B2B services

The objective of this scenario is to enable protected resources for multiple groups of authorized users belonging to diverse administrative domains. In this a scenario, a resource owner owns actuators connected to an Inter-IoT GW. These actuators can accept commands as CoAP PUT requests. Various stakeholders define access control policies in their corresponding ACP (e.g., "Employees", "Managers"). Moreover, the resource owner defines in the GW the access control policies that protect each operation (e.g., "switch1 can be turned on by the "Employees" of the company that has business relationships with ACP A, or the "Employees" of the company that has business relationships with ACP B).

1.3.8.1.2.1 U1: New operation creation and management

The resource owner defines an operation that can be performed on an actuator and provides pointers to the policies that protect this operation. Moreover, later on, the resource owner can modify the list of the pointers to policies by adding or removing a pointer.

T2.1.1 New operation registration

ID	T2.1.1
Test	Registration of a new resource protected by multiple policies
Type	System testing
Setup	Needs setup TS_02
Start	Access Table in GW is empty
Req.	[2],[6],[11],[22],[243],[98]
Input	Resource owner invokes the resource registration API call
Output	Access Table is updated
Logs	Folder "`T1_Output`", prefix "`T2.1.1_achilles`" >
Outcome	**Pass** / Fail

T2.1.2 List of policies modification

ID	T2.1.2
Test	Add or remove a pointer to an access control policy
Type	System testing
Setup	Needs setup TS_02
Start	Access Table in GW has some entries
Req.	[13]
Input	Resource owner invokes the resource registration API call
Output	Access Table is modified
Logs	Folder "`T1_Output`", prefix "`T2.1.2_achilles`" >
Outcome	**Pass** / Fail

1.3.8.1.2.2 U2- User request

A user is interested in triggering an actuator protected by some access control policies. The user performs an initial request (an unauthorized request) to learn all information required for authorization. Then it authenticates himself in the appropriate ACP and obtains an authorization token. The latter is used for performing an authorized request.

T2.2.1 Unauthorized request

ID	T2.2.1
Test	Request from an unauthorized user for a protected actuator
Type	System Testing
Setup	Needs setup TS_02
Start	Access Table contains some entries
Req.	[27]
Input	A CoAP request from an unauthorized user
Output	• Check if the resource is included in the Access Table • Generate session key • Generate token • Respond to the user with the ACP URIs and the token
Logs	Folder "`T1_Output`", prefix "`T2.2.1_achilles`" >
Outcome	**Pass** / Fail

T2.2.2 Authorized request

ID	T2.2.2
Test	Request from an authorized user for a protected resource
Type	System Testing
Setup	Needs setup TS_02
Start	Access Table and Token Table contains some entries
Req.	[14],[15],[72],[76]
Input	A CoAP request from an authorized user
Output	• Check if the resource is included in the Access Table • Check if the Token is included in the Token Table and it is still valid • Perform a CoAP request to the appropriate Thing • Encrypt the response and send it back to the user
Logs	Folder "`T1_Output`", prefix "`T2.2.1_achilles`" >
Outcome	**Pass** / Fail

1.3.8.1.3 S3 - System under attack

The objective of this scenario is to evaluate the security of the integrated platform in the presence of malicious users.

1.3.8.1.3.1 U1-New sessions

An attacker is able to capture and record successful sessions. He then replays the messages in order to gain access to a protected resource.

T3.1.1 Replay attack

ID	T3.1.1
Test	Emulate an attacker that repeats captured sessions
Type	Security Test
Setup	Needs setup TS_01 or TS_02 and Test hook 1
Start	Access Table and Token Table contains some entries
Req.	[27],[28],[95]
Input	A CoAP request that appears to be from an authorized user
Output	• Check if the resource is included in the Access Table • Check if the Token is included in the Token Table and it is still valid • Reply with an error
Logs	Folder "`T1_Output`", prefix "`T3.1.1_achilles`" >>
Outcome	**Pass** / Fail

1.3.8.1.3.2 U2-Tampering with existing sessions

An attacker is able to intercept the communication between an authorized user and a Thing. His goal is to modify the transmitted packets in way that will give him access to protected resources.

T3.2.1 Packet modification attack

ID	T3.2.1
Test	Emulate an attackers that modifies transmitted packets
Type	Security Test
Setup	Needs setup TS_01 or TS_02 and Test hook 1
Start	Access Table and Token Table contains some entries
Req.	[27],[28],[95]
Input	A CoAP request that appears to be from an authorized user
Output	• Check if the resource is included in the Access Table • Check if the Token is included in the Token Table and it is still valid • Reply with an error
Logs	Folder "`T1_Output`", prefix "`T3.2.1_achilles`" >>
Outcome	**Pass** / Fail

T3.2.2 Man-in-the-middle attack

ID	T3.2.2
Test	Emulate an attackers that perform man-in-the-middle attack
Type	Security Test
Setup	Needs setup TS_01 or TS_02 and Test hook 1
Start	Access Table and Token Table contains some entries
Req.	<Define the requirements involved in [x], format>
Input	A CoAP request that appears to be from an authorized user
Output	• Check if the resource is included in the Access Table • Check if the Token is included in the Token Table and it is still valid • Reply with an error
Logs	Folder "`T1_Output`", prefix "`T3.2.2_achilles`" >>
Outcome	**Pass** / Fail

1.3.8.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T1.1.1	New measurement group creation	**Pass** / Fail
T1.2.1	Unauthorized request	**Pass** / Fail
T1.2.2	Authorized request	**Pass** / Fail
T2.1.1	New operation registration	**Pass** / Fail
T2.1.2	List of policies modification	**Pass** / Fail
T2.2.1	Unauthorized request	**Pass** / Fail
T2.2.2	Authorized request	**Pass** / Fail
T3.1.1	Replay attack	**Pass** / Fail

T3.2.1	Packet modification attack	**Pass** / Fail
T3.2.2	Man-in-the-middle attack	**Pass** / Fail
FAT Outcome		**Pass**

Table 10: ACHILLES test outcome overview.

1.3.9 Third Party: Inter-HINC

FAT not executed within the project timeframe.

1.3.10 Third Party: Semantic Middleware

1.3.10.1 Test overview

1.3.10.1.1 S34: Position and Optimization of the pallets

The sensors monitoring the pallet position will play the role of publisher as they will send the information concerning the pallet position through the middleware (Step 1); this information is expressed under the form of a SPARQL UPDATE. Also the working stations will publish their availability status (Step 2). This information will be then consumed by the simulation tool (Optimizer) which has previously subscribed to the changes applied to the pallet position (Step 3) and the availability status of the working stations (using a proper SPARQL query) with the goal to identify the optimized pallet route. In addition, the information concerning the route is then published (Step 4) and in its turn consumed by the IoT actuators which allow to change the route of the pallets along the conveyor belt (Step 5).

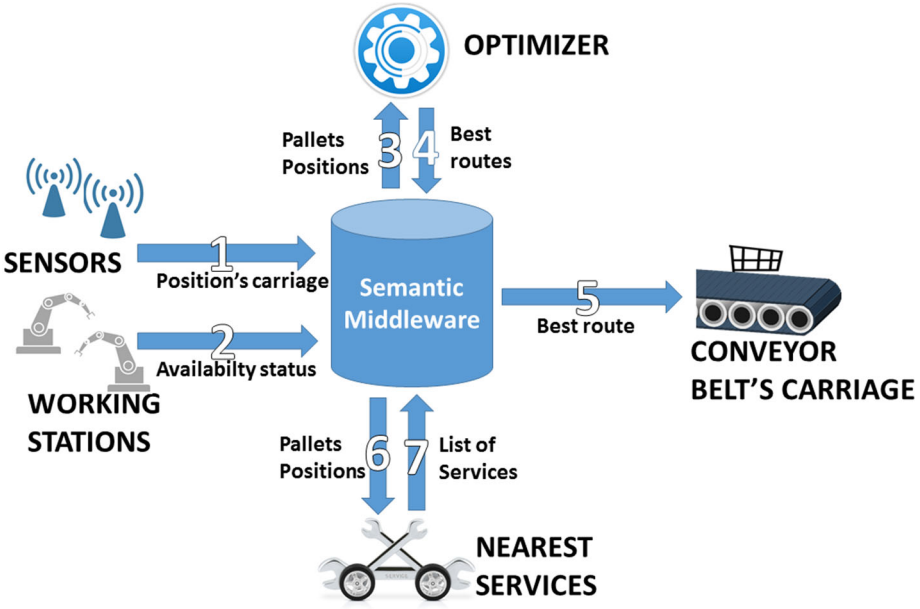

Figure 1. Workflow of the scenario

The use cases reported in the following sections involve various components integrated with the *Semantic Middleware*. The components are the following:

- The Semantic Model. The Semantic model used in these FATs represents knowledge concerning sensors and their corresponding measures, pallets and their real and optimized positions.

- An applications (*Virtual sensor*) that mocks and simulates the behavior of an Ebeacon sensor tracking the position of the pallets. These sensors will publish, through an UPDATE SPARQL (**Query 1**), the position of the pallets P1 and P2 within the knowledge base, according to a proper domain ontology.
  To support the FAT, *Virtual Sensor* provides a GUI that allows to set a position for a specific pallet, thus generating the corresponding UPDATE SPARQL query.

- *Optimizer*, a simulation tool which uses the pallet position and the availability status of the working stations with the goal to identify the optimized pallet route. It consumes data published by the various tracking sensors, while it publishes optimized routes for the pallets.
  To support the FAT, *Optimizer* is virtualized. and provides a GUI that allows to start to listen and consume information concerning position of a specific pallet, thus generating the corresponding SELECT SPARQL query (**Query 2**). In addition, *Virtual Optimizer* allows to set the new best route for the pallet, simulating the behavior of the real application.

- *Virtual carriages* (simulating the real carriages) each one transporting a pallet. They consume data published by the simulator in order to follow a specific route.
  To support the FAT, *Virtual carriage* provides a GUI that allows to start to listen and consume information concerning route for a specific pallet, thus generating the corresponding UPDATE SPARQL query (**Query 3**).

1.3.10.1.1.1 U62 – Device (sensor) triggers information

A device, typically a sensor, triggers an event sending determined information to the gateway in order to be stored on the platform Cloud or server or in order to generate a response for an actuator (being handled by the rules engine).

This use case involves these requirements: [75], [163], [270].

T34.62.1 Information published by Virtual Sensor are persisted

Test	Virtual Sensor publishes information and this information is persisted into the RDF store
Type	System testing
Setup	Need test setup TS_01 SemanticMiddlewareComponents (Query 1, etc.) Need test tool TT_01 RDF store Need test tool TT_02 ActiveMQ Need test hook TH_01 Semantic Model (Query 1, etc.)
Start	Information to be published are not yet persisted
Req.	[75], [163], [270]
Input	Enable the sensor within range of the physical gateway
Output	• The result of a SPARQL query on the RDF store
Outcome	**Pass** / Fail

Test output:

- Access the RDF store and verify through a SPARQL query (SELECT) if the information has been stored

T34.62.2 Information updated by Virtual Sensor are received by the subscribed clients

Test	Information updates are received by the subscribed clients
Type	System testing
Setup	Need test setup TS_01 SemanticMiddlewareComponents Need test tool TT_01 RDF store Need test tool TT_02 ActiveMQ Need test hook TH_01 Semantic Model (Query 1, etc.)
Start	Information to be published are not yet persisted A client (Optimizer) is subscribed to the updated information
Req.	[75], [163], [270]
Input	Information published by Virtual Sensor. It concerns the position of the pallet.
Output	• Check if the subscriber (Optimizer) receives the information updates (postion of the pallet). It has to receive the information updated by the Virtual Sensor. • Check if other subscribers (Virtual Carriage), which are not subscribed to the updated information, does not receive the updated information.
Outcome	**Pass** / Fail

Test output:

- Feedback of the tests will be shown within the GUI of the client applications (TP_01 FeedbackWithinGUI).
- Also a log file can be provided (TP_03 LofFile)

T34.62.3 Information updated by Optimizer are received by Virtual Carriage

Test	Information updates are received by the subscribed clients
Type	System testing
Setup	Need test setup TS_01 SemanticMiddlewareComponents Need test tool TT_01 RDF store Need test tool TT_02 ActiveMQ Need test hook TH_01 Semantic Model (Query 2 and Query 3)
Start	Information to be published are not yet persisted A client (Virtual carriage) is subscribed to the updated information
Req.	[75], [163], [270]
Input	Information published by Optimizer (new route of the pallet).
Output	• Check if the subscriber (Virtual carriage) receives the information updates. The have to receive the updates triggered by the Optimizer changes.
Outcome	**Pass** / Fail

Test output:

- Feedback of the tests will be shown within the GUI of the client applications (TP_01 FeedbackWithinGUI).
- Also a log file can be provided (TP_03 LofFile)

T34.62.4 Updates concerning information on which no client is subscribed

Test	Updates concerning information on which no client is subscribed
Type	System testing
Setup	Need test setup TS_01 SemanticMiddlewareComponents Need test tool TT_01 RDF store Need test tool TT_02 ActiveMQ Need test hook TH_01 Semantic Model (Query 1, etc.)
Start	Information to be published are not yet persisted A couple of clients (Virtual carriage and Optimizer) are subscribed to various information. The latter are not linked with the updated information
Req.	[75], [163], [270]
Input	Information published by Optimizer
Output	• Check if the subscribers (Virtual carriage) receives or not the information updates: They do not have to receive the updates.
Outcome	**Pass** / Fail

Test output:

- Feedback of the tests will be shown within the GUI of the client applications (TP_01 FeedbackWithinGUI).
- Also a log file can be provided (TP_03 LofFile)

1.3.10.2 T34.62.5 (TEST of INTEGRATION with IoT components) The connection with the Semantic Middleware Bridge

Test	Establishing a connection between a mocked client application and a test platform and test the following intermw rest api requests: register client, register platform, register thing and subscribe to thing.
Type	TEST of INTEGRATION with IoT components
Setup	Need TS_04 Inter-IoT middleware, the test bridge and the emulator platform. Need TP_02 Mockup client application
Start	An intermw service is up and running on a server machine reachable over a Local Area Network. The test bridge is packed in a Java Archive File (.jar) and copied to the intermw web application /lib folder, along with the bridge configuration file (.properties).
Req.	[237], [282]
Input	The following input requests are sent to the intermw via the *curl* command through a linux shell on the same host where the intermw is running:

	1. Register client: Command: `curl -X POST --header 'Content-Type: application/json' -d ' {"pullMessagesLimit": 5} ' http://localhost:9080/mw.api.rest/api/intermw/client/myclient` 2. Register platform: Command: `curl -X POST --header 'Content-Type: application/json' -d '{"id":{"id":"http://inter-iot.eu/<tested-platform>"},"type":{"typeId":"TestedPlatform"},"capabilities":[],"baseURL":"http://<Platform Host address>:4568","name":"Tested platform"}' http://localhost:9080/mw.api.rest/api/intermw/platform/myclient`  3. Register thing: Command:  `curl -X POST --header 'Content-Type: application/json' -d '{"attributes":[],"platformId":{"id":"http://inter-iot.eu/<tested-platform>"},"thingId":{"id":"http://www.example.com/sensor1"}}' http://localhost:9080/mw.api.rest/api/intermw/thing/myclient`  4. Subscribe to registered thing: Command: `curl -X POST --header 'Content-Type: application/json' -d '{"attributes":[],"platformId":{"id":"http://inter-iot.eu/<tested-platform>"},"thingId":{"id":"http://www.example.com/sensor1"}}' http://localhost:9080/mw.api.rest/api/intermw/subscribe/myclient`
Output	Check if the curl command succeeds and the request is correctly sent to the tested platform through the tested bridge.
Outcome	**Pass** / Fail

Test output:

- Feedback of the curl command are shown through the shell where the curl command is executed. All the requests succeeds returning the following response:

  `>> {"success":"true"}`

Otherwise an error message appears. For example, in the case the register Client failed (due to an existing client) the message is:

```
>> {"type":"error","message":"An unexpected error occurred: Halt! No registering of an already registered client is allowed!"}
```

- Feedback about the integration between components are shown through the debug messages printed out by the tested platform. For example, if a thing subscribe request is correctly sent to the tested platform, the following debug message will be printed:

```
12:46:14.661 [main] INFO  eu.interiot.intermw.bridge.<Tested Platform>.start(<classname>.java:81) Received /things/subscribe request.
```

T34.62.6 (INTEGRATION with IoT comp) The ontology alignment test through IPSM

Test	**The ontology alignment test between our Application Ontology (AO) and the GOIoTP ontology will be performed according to the general structure of the INTER-IoT alignment format (also called IPSM alignment format) [ref1]. The alignment element describes a uni-directional set of translation rules comprised of independent mapping cells, each of which has an "input" and "output" entity descriptions.**
Type	TEST of INTEGRATION with IoT components
Setup	Need test setup TS_02 IPSM Need test hook TH_01 Semantic Model (a subset of this model must be aligned)
Start	Invoking the proper function of IPMS Aligner (dashboard : http://grieg.ibspan.waw.pl:3000/translation) and passing it the mapping in the form presented above
Req.	[178], [179], [180]
Input	The following message: { "@graph": [{ "@graph": [{ "@id" : "http://itia.cnr.it/SemanticMiddleware#testSensor", "@type" : "http://itia.cnr.it/SemanticMiddleware#Sensor" }, { "@id" : "http://itia.cnr.it/SemanticMiddleware#testMachining", "@type" : "http://itia.cnr.it/SemanticMiddleware#Machining" }, { "@id" : "http://inter-iot.eu/GOIoTP#testPlatformComponent",

```
  "@type" : "http://inter-iot.eu/GOIoTP#PlatformComponent"
},

{
  "@id" : "http://itia.cnr.it/SemanticMiddleware#testProductionResource",
  "@type" : "http://itia.cnr.it/SemanticMiddleware#ProductionResource"
},


{
  "@id" : "http://itia.cnr.it/SemanticMiddleware#testResourceComponent",
  "@type" : "http://itia.cnr.it/SemanticMiddleware#ResourceComponent"
},


        {
          "@id": "http://itia.cnr.it/SemanticMiddleware#testProductionResource",
          "@type": [
            "http://itia.cnr.it/SemanticMiddleware#ProductionResource"
          ],
          "http://itia.cnr.it/SemanticMiddleware#hasResourceComponent": {
              "@id": "http://itia.cnr.it/SemanticMiddleware#testResourceComponent"
          }
        },



{
  "@id" : "http://itia.cnr.it/Sensor",
  "@type" : [ "http://www.w3.org/2002/07/owl#Class" ]
},
{
  "@id" : "http://itia.cnr.it/SemanticMiddleware#Machining",
  "@type" : [ "http://www.w3.org/2002/07/owl#Class" ]
},
{
  "@id" : "http://inter-iot.eu/GOIoTP#PlatformComponent",
  "@type" : [ "http://www.w3.org/2002/07/owl#Class" ]
}
],
      "@id": "InterIoTMsg:payload"
    }
  ],
  "@context": {
    "ns": "http://ontology.universaal.org/PhThing.owl#",
    "owl": "http://www.w3.org/2002/07/owl#",
    "InterIoTMsg": "http://inter-iot.eu/message/",
```

	"InterIoTInst": "http://inter-iot.eu/inst/", "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#", "xsd": "http://www.w3.org/2001/XMLSchema#", "rdfs": "http://www.w3.org/2000/01/rdf-schema#", "InterIoT": "http://inter-iot.eu/", "ns2": "http://ontology.universaal.org/Measurement.owl#", "ns1": "http://ontology.universAAL.org/Context.owl#", "ns4": "http://ontology.universAAL.org/Device.owl#", "ns3": "http://ontology.universaal.org/HealthMeasurement.owl#" } }
Output	We expect to obtain the following message in order to accept the alignment test: <**map**><**Cell**> <**entity1** rdf:resource="http://www.opengis.net/gml/Point"/> <**entity2** rdf:resource="http://www.w3.org/2003/01/geo/wgs84_pos#Point"/> <measure rdf:datatype="http://www.w3.org/2001/XMLSchema#float">0.99</measure> <**relation**>=</**relation**> </**Cell**></**map**> The correct response returns the URI of the entities that have been aligned and two fundamental information: the logical relation existing between them (<relation>) and the relative confidence of such relation (<measure>).
Outcome	**Pass** / Fail

Test output:

- The result is reported in the GUI of the dashboard:

```
{
  "@graph" : [ {
    "@id" : "InterIoT:GOIoTP#PlatformComponent",
    "@type" : "owl:Class"
  }, {
    "@id" : "InterIoT:GOIoTP#testPlatformComponent",
    "@type" : "InterIoT:GOIoTP#PlatformComponent"
  }, {
```

```
    "@id" : "http://itia.cnr.it/SemanticMiddleware#Machining",
    "@type" : "owl:Class"
  }, {
    "@id" : "http://itia.cnr.it/SemanticMiddleware#testMachining",
    "@type" : "InterIoT:GOIoTP#Service"
  }, {
    "@id" : "http://itia.cnr.it/SemanticMiddleware#testProductionResource",
    "@type" : [ "InterIoT:GOIoTP#IoTDevice", "InterIoT:GOIoTP#SoftwarePlatform" ],
    "InterIoT:GOIoTP#hasComponent" : {
      "@id" : "http://itia.cnr.it/SemanticMiddleware#testResourceComponent"
    }
  }, {
    "@id" : "http://itia.cnr.it/SemanticMiddleware#testResourceComponent",
    "@type" : "InterIoT:GOIoTP#PlatformComponent"
  }, {
    "@id" : "http://itia.cnr.it/SemanticMiddleware#testSensor",
    "@type" : "InterIoT:GOIoTP#IoTDevice"
  }, {
    "@id" : "http://itia.cnr.it/Sensor",
    "@type" : "owl:Class"
  } ],
  "@id" : "InterIoTMsg:payload",
  "@context" : {
    "ns" : "http://ontology.universaal.org/PhThing.owl#",
    "owl" : "http://www.w3.org/2002/07/owl#",
    "InterIoTMsg" : "http://inter-iot.eu/message/",
    "InterIoTInst" : "http://inter-iot.eu/inst/",
    "rdf" : "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "xsd" : "http://www.w3.org/2001/XMLSchema#",
    "rdfs" : "http://www.w3.org/2000/01/rdf-schema#",
    "InterIoT" : "http://inter-iot.eu/",
    "ns2" : "http://ontology.universaal.org/Measurement.owl#",
    "ns1" : "http://ontology.universAAL.org/Context.owl#",
    "ns4" : "http://ontology.universAAL.org/Device.owl#",
    "ns3" : "http://ontology.universaal.org/HealthMeasurement.owl#"
  }
```

```
}
```

Reference

[ref1] Maria Ganzha et al., Alignment-based semantic translation of geospatial data,

T34.62.7 (INTEGRATION with IoT comp) Application Ontology imports GOIoTP

Test	GOIoTP ontology are imported within the Application Ontology used to represent knowledge for the scenario S34.
Type	TEST of INTEGRATION with IoT components
Setup	Need test setup TS_03 GOIoTP Need test tool TT_01 RDF store Need test hook TH_01 Semantic Model
Start	The following subsequent steps will be carried out: 1. Add the import directive in the AO ontology directed to the GOIoTP ontology, so that all the statements of the latter are imported in the former ontology; 2. After the concept alignment has been executed between AO and GoIoTP, the alignment results are used in order to create logical relation axioms within the integrated ontology (e.g., equivalentClass axioms, subClassOf, etc.).
Req.	[42], [96]
Input	Instantiate an ontological individual as an instance of a specific class of the AO ontology, which is equivalent to a class imported from the GoIoTP ontology.
Output	Test if ontological individual inherits the features of the equivalent class.
Outcome	**Pass** / Fail

Test output:

1.3.10.3 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T31.62.1	Information published by Virtual Sensor are persisted	**Pass** / Fail
T31.62.2	Information updated by Virtual Sensor are received by the subscribed clients	**Pass** / Fail
T31.62.3	Information updated by Optimizer are received by Virtual Carriage	**Pass** / Fail
T31.62.4	Updates concerning information on which no client is subscribed	**Pass** / Fail
T31.62.5	(TEST of INTEGRATION with IoT components) Connection with the Bridge	**Pass** / Fail
T31.62.6	(TEST of INTEGRATION with IoT components) The ontology alignment test through IPSM	**Pass** / Fail
T31.62.7	(TEST of INTEGRATION with IoT components) Ontology import Text	**Pass** / Fail
FAT Outcome		**Pass**

Table 11: Semantic Middleware test outcome overview

1.3.11 Third Party: SecurIoTy

FAT not executed within the project timeframe.

1.3.12 Third Party: E3City

1.3.12.1 Test description

1.3.12.1.1 Scenario: Testing power-on, power-off and reception of measurements of a luminaire.

In the following, we detail a scenario where we will check the operation of a system that must control the lighting of luminaire and receive measures from it.

T1.1.1 Test electrical

ID	T1.1.1
Test	This test is to check the correct manufacture of the equipment
Type	Physical
Setup	TS_01
Start	Connected to a battery
Req.	Battery and tools
Input	
Output	
Logs	Our database
Outcome	**Pass** / Fail

T1.1.2 Test connectivity

ID	T1.1.2
Test	Probe the connection with de cloud
Type	Cloud
Setup	TS_02
Start	Connected to the platform
Req.	
Input	Commands from the platform
Output	Error report
Logs	Our database
Outcome	**Pass** / Fail

T1.1.3 Test correct interpretation of commands

ID	T1.1.3
Test	Probe the response of the device
Type	Cloud
Setup	TS_03
Start	Connected to the platform
Req.	
Input	Commands from the platform
Output	Correct response to commands
Logs	Our database
Outcome	**Pass** / Fail

1.3.12.2 Outcome overview

The following table will provide an overview of the test result of all the performed tests in this FAT.

Test	Description	Outcome
T.01	Test electrical	**Pass** / Fail
T.02	Test connectivity.	**Pass** / Fail
T.03	Test correct interpretation of commands	**Pass** / Fail
FAT Outcome		**Pass**

Table 12: E3City test outcome overview