This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement N° 732679

ACTIVAGE PROJECT

ACTivating InnoVative IoT smart living environments for AGEing well

Interoperability Report

Deliverable No.	D3.5	**Due Date**	30-06-2018
Type	Report	**Dissemination Level**	Confidential
Version	1.0	**Status**	Release
Description	Interoperability report [M18, M30] - Deployment ACTIVAGE Interoperability Framework; developer and user		
Work Package	WP3		

Authors

Name	Partner	e-mail
Jorge Posada	01 MDT	jorge.posada@medtronic.com
Mario Diaznava	02 STM	mario.diaznava@st.com
Álvaro Martínez	04 MYS	amartinez@mysphera.com
Alejandro Medrano	05 UPM	amedrano@lst.tfo.upm.es
Helmi Ben Hmida	06 Fh-IGD	Helmi.Ben.Hmida@igd.fraunhofer.de
Stéphane Bergeon	07 CEA	Stephane.bergeon@cea.fr
Mathieu Gallisot	07 CEA	Mathieu.Gallisot@cea.fr
Konstantinos Votis	08 CERTH	kvotis@iti.gr
Nikolaos Kaklanis	08 CERTH	nkak@iti.gr
Stefanos Stavrotheodoros	08 CERTH	stavrotheodoros@iti.gr
Dimitrios Tzovaras	08 CERTH	dimitrios.tzovaras@iti.gr
Philippe Dallemagne	09 CSEM	philippe.dallemagne@csem.ch
Regel G. Usach	11 UPV	regonus@upv.es
Matilde Julian	11 UPV	majuse@teleco.upv.es
Carlos E. Palau	11 UPV	cpalau@dcom.upv.es
Clara Valero	11 UPV	clavalpe@upv.es
Germán Molina	12 HOPU	german@hopu.eu
Felipe Roca	12 HOPU	felipe@hopu.eu
Tuan Tran	13 NUIG	tuan.trannhat@insight-centre.org
Aqeel Kazmi	13 NUIG	aqeel.kazmi@insight-centre.org
Martin Serrano	13 NUIG	martin.serrano@insight-centre.org
Dario Russo	23 CNR	dario.russo@isti.cnr.it
Andrea Carboni	23 CNR	andrea.carboni@isti.cnr.it
Korina Papadopoulou	40 GNO	k.papadopoulou@gnomon.com.gr
Liverios Stavropoulos	40 GNO	l.stavropoulos@gnomon.com.gr

History

Date	Version	Change
17-May-2018	0.1	Structure of the document and task assignments
28-May-2018	0.2	Structure finalized & drafts of sections 2, 5 & 7.3 finished
6-Jun-2018	0.3	Integration of drafts of sections 3, 5.1, 8
14-Jun-2018	0.4	Integration of consolidated drafts of section 3, 5.1, 5.4, 7.2. Adjustments to align section 3 with other sections.
26-Jun-2018	0.5	Integration of section 6 and changes on section 4.
27-Jun-2018	0.6	Changes on the order of sections. Integration of missing sections.
18-Jul-2018	0.7	Changes recommended after internal revision
19-Jul-2018	1.0	Official release

Key data

Keywords	Interoperability, IoT platforms, semantic & syntactic interoperability, data model
Lead Editor	Regel G. Usach, 11 UPV
Internal Reviewer(s)	Juan Montalva, 05 UPM Pilar Sala, 04 MYS

Abstract

This deliverable is the continuation of the work presented in deliverable 3.2 "Interoperability layer architecture" and extends further information in this regard presented in deliverable 5.1 "Integration Plan and Operational Framework". This document is closely related with the outcome of tasks T3.3 and T3.4. D3.5 provides further information regarding the definition and implementation of components, tools and techniques employed within ACTIVAGE project to facilitate interoperability with the DS IoT platforms. In particular, main initiatives for the enablement of interoperability are the Semantic Interoperability Layer, that provides both syntactic and semantic interoperability, and the creation of a common data model. Also, some security and privacy considerations are drawn.

Statement of originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Table of contents

List of tables

List of figures

1 About This Document

This document is the deliverable D3.5 “Interoperability Report” and represents a continuation of the content provided on deliverable D3.2 “Interoperability Layer Architecture” and on deliverable D5.1 “Integration Plan and Operational Framework” in regard of the Semantic Interoperability Layer (SIL). D3.5 is closely related with the outcomes of the activities T3.3 “Building bridges to platforms and protocols” and T3.4 “Implementing the Semantic Interoperability Layer”, from Work Package WP3 “ACTIVAGE Secure Interoperability Layer”. Moreover, from a broader scope, this document has also some relation with the outcomes of other two tasks of WP3: T3.1 “Specification of the open cross-pilot ACTIVAGE architecture” and T3.2 “ACTIVAGE solution for security and privacy”.

The overall scope of this deliverable is to present the elements that enable interoperability in ACTIVAGE: the Semantic Interoperability Layer (SIL), and the use of a common data model. Another main objective is to provide a set user guides to enable the deployment, development and use of this interoperability framework. The SIL has been addressed in previous deliverables D3.2 and D5.1. This document extends the information provided in those deliverables; in particular, the role of platform bridges in the Interoperability, and the use of semantic alignments for achieving semantic translations with the IPSM (Inter Platform Semantic Mediator). Moreover, precise information for further development of bridges is given in user guides within the document, and a technical guide for SIL deployment. Also, this deliverable presents new refined use cases of interoperability in ACTIVAGE.

The content of this deliverable can be summarized as follows:

Section 1 describes the nature and the reason why this deliverable has a particular relevance for the ACTIVAGE project. Moreover, in this section are also defined the structure, organisation and objectives of the document.

Section 2 is an introductory section that explains the objectives and proposals of the deliverable in detail, beside the challenges that must be confronted to reach the deliverable goal.

Section 3 gives a general overview about IoT interoperability and the needs regarding interoperability in the ACTIVAGE project. It contains a brief state-of-art and main concepts of IoT interoperability detailed in a clear and descriptive way, and explains the particular needs of interoperability of the ACTIVAGE system.

Section 4 explains the potential interoperability use cases that can be achieved in ACTIVAGE with the use of the current interoperability framework.

Section 5 explains the current framework for interoperability within the ACTIVAGE project. It is devoted to the Semantic Interoperability Layer that enables interoperability among IoT Platforms from Deployment Sites. This layer has already been described in deliverables D3.2 and D5.1 giving this deliverable further information and details about this set of components for interoperability. Moreover, new interoperability use cases are described in addition to those already explained in deliverable D3.2.

Section 6 describes one of the elements in ACTIVAGE that enables semantic interoperability with the data shared from deployment sites, conjointly with the SIL IPSM: a common Data Model.

Section 7 addresses security and privacy concerns that must be taken into account regarding the interoperation and data sharing with DS and gives information about the role of the security and privacy module of task T3.2 on interoperability for ensuring secure data exchanges without compromising privacy.

Section 8 provides detailed technical guides for the deployment and development of the SIL. In this way SIL can be implemented and also extended, thus improving the ACTIVAGE interoperability (i.e. more platforms can be included in DS). Moreover, other future guides in progress in the project related with the enablement of interoperability are commented.

Finally, the document ends with some conclusions and an outlook of the future work.

1.1 Deliverable context

Project item	Relationship
Objectives	*Semantic interoperability layer to allow integration and interoperability of heterogeneous platforms;* *Framework and API to allow the connection of new services and interact transparently with IoT platforms.*
Exploitable results	*The ACTIVAGE architecture is one of the necessary outputs so as to create the ACTIVAGE IoT Ecosystem Suite in WP5.*
Work plan	*This document summarizes the activity of T3.4 Implementation of the interoperability layer, and T3.3 Building bridges to the IoT platforms and protocols.* *Moreover, this deliverable is linked with T3.1 Specification of the open cross-pilot ACTIVAGE architecture and has some transversal relation with T3.2 ACTIVAGE solution for security and privacy.* *The outputs of WP3 will be integrated in WP5 in the AIoTES suite, along with other component and tools developed in WP4.*
Milestones	*MS1 - BUILD - All DS ready and solution integrated. D9.1 Completed.*
Deliverables	D3.5 Interoperability report [M18, M30] - Deployment ACTIVAGE Interoperability Framework; developer and user guide
Risks	*This deliverable contributes to the clearance of Risk 16, 'Difficult to achieve technical integration/interoperability between existing platforms'*

2 Introduction

The ACTIVAGE project aims to guarantee interoperability among platforms and applications. Platforms in IoT are key elements of the AHA Deployment Sites, but unfortunately, they are not interoperable among them, and also not directly to the ACTIVAGE Core. Data from an IoT platform has specific standards, data format, structure and semantics, and thus, it cannot be directly understood by other platforms or systems. Therefore, these platforms and the ACTIVAGE system are unable to directly communicate among them.

Thus, interoperability is key in ACTIVAGE to enable data sharing with IoT platforms, deployment sites, and applications that feed from these data sources. Syntactic and semantic interoperability are necessary to enable the understanding at different levels of the information shared from different IoT platforms. With the aim of providing interoperability among the different Deployment Sites, and to facilitate the creation of common applications on top of ACTIVAGE, several elements had been implemented in the ACTIVAGE architecture within the Semantic Interoperability Layer (SIL). The role of the SIL is to provide an abstraction layer that enables interoperability among the Deployment Sites and the ACTIVAGE system. Additionally, a common data model is being designed for enabling semantic interoperability among the different data sources of ACTIVAGE.

SIL is a component of ACTIVAGE IoT Ecosystem Suite (AIoTES). This suite consists of a set of techniques, tools and methodologies for interoperability between heterogeneous IoT Platforms and an open framework for providing semantic interoperability of IoT Platforms for AHA, while addressing trustworthiness, privacy, data protection and security. The SIL is responsible of the data interoperability with other IoT platforms, while other elements integrated in AioTES provide additional functionality (e.g. security and privacy). SIL provides interoperability among platforms, and allows other elements of ACTIVAGE to communicate with any platform through a common API.

This document is very related with several WP3 tasks. In special, it is closely linked with T3.3 ‘Building bridges to the IoT protocols and platforms’ and T3.4.’Implementation of the interoperability layer’. It is a continuation of the work of T3.1 ‘Specification of the open cross-pilot ACTIVAGE architecture’. Also task T3.2 ‘ACTIVAGE solution for security and privacy’ has a transversal importance in this interoperability report.

This document aims to provide a report regarding the interoperability development (components, tools, techniques) employed in ACTIVAGE with the aim to facilitate interoperability, extending the information about the SIL already provided in D3.2 and D5.1, and explaining for the first time the design of AIoTES common data model. Also, D3.5 aims to provide a guide for the deployment, use and even development and extension of the aforementioned components that enable interoperability. Moreover, the uses cases of this interoperability are detailed. And furthermore, some security and privacy considerations are commented, as these aspects are of utmost importance regarding any data sharing and interoperation with the deployment sites.

3 IoT interoperability

3.1 Importance of IoT interoperability

Today, the IoT market is growing and offering increasingly attractive applications but the lack of industry consensus on the use of open standards and protocols is posing a major barrier to their diffusion. In addition, most notable about current software development practice is the continued preponderance of ad-hoc approaches driven by industry needs, commercial interests, and market pressures, rather than scientific principles. It can be stated that "every IoT domain and every IoT vendor produces its own IoT platform." As a matter of fact, different "vendor groups" can be found in different domains, while not a single vendor can be seen as having an "upper hand" in being positioned across all IoT domains. Unfortunately, without the definition of a common mechanism through which devices and applications can exchange information, regardless of their technological standard, brand or manufacturer, the IoT will never reach its full potential[1].For these reasons, research IoT challenges like standards, scalability, heterogeneity, common service description language, domain specific service discovery, integration with existing IT systems etc. have to be faced. Those topics are strongly related with the concept of interoperability.

The International Organization for Standardization (ISO) has defined *interoperability* [2] as the ability of two or more systems to understand, to use each other's functionalities and to give access to their respective resources. The *Healthcare Information and Management Systems Society* (*HIMSS*) has also given its own definition of *interoperability* [3]. *HIMSS* has described it as "the extent to which systems and devices can exchange data, and interpret that shared data. For two systems to be interoperable, they must be able to exchange data and subsequently present that data such that it can be understood by a user".

On a technical level, interoperability helps to reduce the time it takes to have useful exchange of information between providers. Some advantages that highlight the importance of interoperability are:

- *Improved Efficiency*: interoperability is designed to boost efficiency. When data is presented on a consistent basis no matter what the source, it's easier for users to quickly get to the information they are seeking.
- *Safer Transitions of Care*: continuity of care is crucial for patients, whether for chronic conditions or taking care of an acute situation with multiple health service providers. Interoperability enables safer transitions of care, which leads to better patient outcomes.
- *Can Help Lower Costs*: Interoperability means that more useful information can be shared in a timely manner. So, the data from a patient who had a blood test last week at his doctor's office can be used today during a trip to the emergency room, saving the time and cost of doing more (and unneeded tests) at the hospital.

This means that interoperability will help the final users to work in mixed environments without facing the complexity of managing different technologies inside their organization's infrastructure, reducing the cost of buildings. In a worldwide view, users can make available on the network devices belonging to multiple vendors' technologies. This permits to implement scalability in *IoT* increasing the workflow efficiency in any environment connecting objects from anywhere to anywhere using different technologies.

Interoperability is surely the main challenge to be faced to realize IoT, due to the current inexistency of a widely accepted global standard for IoT, and the vast heterogeneity of IoT systems and elements, at all levels.

During last years, the EU co-financed numerous research project to create innovative IoT platforms [4]. Each solution has provided a specific vision of architecture based on the requirements, scope and domain of the project, focusing on different aspects or a sub-domain of *IoT*. Due to a large heterogeneity of application domains, requirements and approaches to the architecture, the proposed solutions are only interoperable between themselves, and not with other platforms or solutions.

Moreover, to face interoperability between IoT platforms, the EU has funded seven research projects that started in January 2016. One of them is the *INTER-IoT* project [5], which provides the background for the *Semantic Interoperability Layer* of *ACTIVAGE*. The *INTER-IoT* project presents a comprehensive approach to IoT platform interoperability, offering its tools and services across the communication/software stack.

3.2 Interoperability in IoT systems

IoT systems and architectures, today, consists mostly on heterogeneous standards and proprietary interfaces. The traditional architecture is made by different devices, sensors and applications that require different and customized middleware. This type of architecture is known as “vertical architecture” and typically does not allow data sharing between different applications and results in global inefficiency. Additionally, it leads to barriers, especially for small enterprises, as one solution provides a separate interface for each different IoT platform. Horizontal platform architectures, instead, are based on a common middleware layer, allowing data sharing across different platforms and grants a better use of the devices.
IoT interoperability is important at all layers of the hardware/software stack:

- the **device layer**, to seamlessly integrate new devices into the existing IoT ecosystem;
- the **networking layer**, to handle object mobility and information routing;
- the **middleware layer**, to facilitate seamless service discovery and management of smart objects;
- the **application service layer**, to enable the combination of heterogeneous application services from different IoT platforms (inter-platform service composition);
- the **data and semantics layer**, to introduce common understanding of data and information among the two IoT systems or entities that share this data.

From a healthcare point of view the Healthcare Information and Management Systems Society (HiMSS[1]) identifies three levels of health information technology interoperability:

- *Foundational Interoperability*: gives the ability to one information system to exchange data with another information system. The data will be available for use.
- *Structural (Syntactic) Interoperability*: defines the format of the data exchange. This has to do with standards that govern the format of messages that are exchanged, so that the operational or clinical purpose of the information is evident and passes through without alteration.
- *Semantic Interoperability*: is the highest level of connection. Two or more different systems or parts of systems can exchange and use information. The structure of the exchange of data and how the data itself is codified lets medical providers share patient data even when using completely different software solutions from different vendors.

The foundation of interoperability between heterogeneous platforms are given by a correct definition and implementation of the components related to the syntax and semantic domains. The concept of foundational interoperability is very related with the capability of a system of

[1] http://www.himss.org

establishing a connection to the network, as a first step necessary to enable the other two types of interoperability. Foundational interoperability is assumed in systems capable of connecting to the Internet (and thus capable of send/receive information bytes). On the other hand, syntactic and semantic interoperability require a more extended description to understand in depth the concepts and also the challenges to face in ACTIVAGE for its enablement.

3.2.1 Syntactic Interoperability

The syntactic interoperability refers to the agreement on rules that specify format, structure and encoding of the information that has to be exchanged between transaction parties. It is like the natural language syntax, documents, paragraphs and sentences that contain words that follow rules and structures for mental decomposing by the reader. Proper syntax enables decomposition of content (ability to read the content); it does not mean that the content makes sense or its meaning is understood by the receiver entity.

Syntactic interoperability provides the following functions:

- Translation of character data from one format to another, such as EBCDIC to ASCII
- Message content structure, such as SOAP encoding
- Message exchange patterns, such as Synchronous Request/Response or synchronous Publish/Subscribe.

Examples of common Syntactic Interoperability standards include:

- HTML - Hypertext Markup Language
- XML - Extensible Markup Language
- ASN.1 - Abstract Syntax Notation One
- SOAP - Simple Object Access Protocol
- SNMP - Simple Network Management Protocol
- JSON - JavaScript Object Notation

Some popular approaches for syntactic interoperability are:

- service-oriented computing (SOC)-based architecture [6],
- web services [7],
- RESTful web services [8],
- open standard protocols,
- closed protocols,
- CORBA (Common Object Request Broker Architecture) [9]

3.2.2 Semantic Interoperability

The semantic interoperability is focused on the meaning of the data that are shared between communicating parties, to achieve a common understanding of this meaning on both entities (e.g. IoT platforms). This means that the data sent from a device x to another device y is interpretable by the last one exactly as the first one originally means, and the meaning responding from the device y to device x is interpretable by the device x exactly as the device y originally means. To make this work, it is required, first, syntactic interoperability (ability to understand the data formats), and second, a semantic consistency that permits a correct meaning and interpretation of device messages among devices, software programs, IoT platforms and humans. Let's consider the example of the switching of a light: semantic interoperability will allow that the turn on and turn off meanings can be understood by a lamp with the respective commands open and close, and at the same time, by another different lamp, with a different vendor and different specifications with the respective commands on and off. But generally two devices or two IoT systems are not semantically interoperable. To

face semantic interoperability problem in IoT, ontologies are used. An ontology for IoT is a way of storing knowledge and is composed by set of objects and relationships among them. It can contain information about both concrete data items about entities (individuals) and structural information about data, usually in a given area of interest. One formal way of expressing semantics is the OWL, the W3C proposed semantic web language designed to represent a complex knowledge from a domain in a human and machine understandable and readable.

To implement ontology interoperability, there are three different main techniques [10]:

- ontology alignment: is a set of correspondences between two or more ontologies. Correspondences may be simple (between atomic entities) or complex (between groups of entities and sub-structures), but always relate entities from different ontologies. Alignments contain predicates about similarity, called a matching (e.g. equivalence and subsumption axioms), or a logical axiom—a mapping. Ontology alignment tools often state a degree of confidence for every correspondence in the mapping. Note, that the terms “mapping” and “matching” are often not distinguished in the terminology used by the alignment tools and are used interchangeably.
- ontology merging: is a process of combining two, or more, ontologies into one so that the resulting ontology stores knowledge from all merged ones. A simplistic case of a merge is an ontology that is a result of a simple sum of sets of axioms from combined ontologies. More sophisticated merges include additional axioms that state how the axioms from combined ontologies relate to each other. Such additional axioms are often the result of an alignment.
- ontology translation: is a process of changing the underlying semantics of a piece of knowledge. Given some information described semantically, in terms of a source ontology, it is transformed into information described in terms of a target ontology. Resulting information contains information interpretable (understandable) in the scope of target semantics. Semantic translation is considered good if the meaning of original knowledge is preserved. Ideally, no information is lost as a result of translation process. A perfect translation is also reversible, and original information can be ideally reconstructed by reversing the translation.

3.3 Interoperability needs in ACTIVAGE

ACTIVAGE’s DS network will ensure efficient management of the deployments at National level and strong coordination at transnational level with common generic Use Case definition, shared user involvement methodology, common usage and acceptance evaluation process, cross implementation of solutions thanks to interoperability of the platforms.
Pilots are encouraged to exploit previous work where applicable with the objective of further demonstrating the generic applicability and interoperability of already available architectures, platforms and standards, and to identify where standards are missing or should evolve, as well as needed pre-normative activities.
The ACTIVAGE Large Scale IoT Pilot for Ageing Well consists of 9 independent Deployment Sites (DS) where the DS reuse and scale up open and proprietary IoT platforms, technologies and standards in order to create the first European IoT ecosystem for AHA. It is important to note that each DS of the project uses different IoT platforms and the establishment of an AHA ecosystem is facing the problem of the lack of interoperability across IoT platforms and things. Each DS typically employs a single IoT platform, and currently, seven different IoT platforms are employed in the DSs of ACTIVAGE (FIWARE, UniversAAL, OpenIoT, SensiNACT, SOFIA2, SENIORSOME). Hence, it emerges the creation of the AIoTES framework, whose aim is to resolve this lack of interoperability by providing an interoperable framework that any of the IoT platforms of the DS can connect to. Also, applications on top can extract information

from DS through the AIoTES API. The aim is to provide a stable, consistent and safe framework for both vendors and consumers to integrate without redesigning connectivity. Hence, developers can succeed in overcoming the interoperability challenge of developing IoT platform independent applications and services.
The ACTIVAGE project consists of several heterogeneous platforms and deployment sites which reuse technologies in order to create the first European IoT ecosystem for AHA. The different standards, data formats and semantics between the IoT platforms make them unable to interoperate among them, thus are causing a lack of interoperability. Platforms are unable to communicate and share data (in an understandable way for them) among themselves nor among the ACTIVAGE system. This also happens with the platforms and the applications built on top of ACTIVAGE. Applications cannot employ data from a DS if these applications have not been developed specifically for the DS particular IoT platform. Interoperability cases on are described in depth in this deliverable, on section 4.
Therefore, the ACTIVAGE proposes the use of the AIoTES framework whose architecture resolves the interoperability issue.
The deployment sites need interoperable IoT-enabled Active & Healthy Ageing solutions, while they also have the ability to enhance those systems with new features and services. The need for interoperability here is clear. AIoTES can integrate remote healthcare services and wearable devices which provide important remote medical information such as physical detection and medical measurements. Also, the deployment site will have the opportunity to communicate with others in order to exchange features and tools.
Thus, ACTIVAGE requires interoperability at three levels:

- Intra-deployment site interoperability: services provided at each deployment site must be interoperable to each other.
- Inter-deployment site interoperability: enables to new services to be automatically incorporated into the ecosystem of the deployment sites (DS).
- Interoperable external adopted solutions: the DS's will call for solutions that by default need to be interoperable according to the ACTIVAGE interoperability framework and according to their needs as demand sites, this process will be run during the Open Call process in the project.

3.4 ACTIVAGE interoperability approach

Key elements for ACTIVAGE's interoperability are already addressed and described in deliverable D3.2 "Interoperability Layer Architecture". These aspects are:

- The alignment with other IoT architectures
- The support of known communication protocols
- The support of machine-to-machine (M2M) communication
- The support of the known middleware platforms
- Extensibility
- Semantic and syntactic interoperability

The ACTIVAGE architecture is based on the IoT-A model, which its goal is to have standardisation in the Internet of Things. This standardisation can enable interoperability between different IoT systems, if they are compliant with it. IoT-A developed an Architectural Reference Model (ARM) for generating reference architectures based on domain specific requirements in the Internet of Things domain, together with the definition for the technical design of its protocols, interfaces and algorithms to achieve this goal.

The ACTIVAGE system architecture is aligned with the reference models of other IoT projects, especially IoT-A. This alignment with other architectures is useful in order to avoid reinventing

new architectural models from scratch and to be compatible with those projects (thus more easily interoperable).

The system architecture of ACTIVAGE follows the machine-to-machine approach with minimal human user involvement. The communication between nodes are M2M which allows different machines to interact with back-end systems and with other machines to provide real time information. M2M interoperability is able to help technology providers to maximize use of their existing infrastructure. An M2M approach also enable autonomy in heterogeneous systems thus increasing interoperability.

In addition, the ACTIVAGE system architecture has transparent access to integrated platforms from DS and is able to receive data from different sensor types. The system is ready to support extensions, upgrades and addition of new modules.

The achievement of this transparent access to IoT platforms has the barrier of the lack of interoperability among IoT platforms and systems due to the use of different standards, data formats, semantics and way of functioning. Inter-platform interoperability is one of the most complex challenges to solve in IoT. Typically, IoT platforms remain isolated islands of information (vertical silos), unable to interoperate or communicate with other platforms or systems. To solve this barrier, it is necessary an abstraction layer between the IoT platforms from DS and the ACTIVAGE system that provides both syntactic and semantic interoperability among the IoT platforms and the ACTIVAGE system. This layer has already been described in deliverables D3.2 and D5.1, whereas this document, D3.5, aims to extend the information provided in them by explaining how to deploy the SIL, what the alignments and bridges are (key elements for syntactic and semantic interoperability among a specific platform), and how it is possible to develop new ones, thus extending the SIL by integrating new platforms. This approach is not merely informative, and it aims to be as well a practical guide on how to deploy and use the interoperability framework and how to extend the SIL functionality and integrate new platforms by developing new interoperability enablers.

The SIL is the core element, responsible of enabling interoperability with platforms, of ACTIVAGE IoT Ecosystem suite called AIoTES. AIoTES is a set of techniques, features and tools that provide semantic interoperability between all the heterogenous IoT platforms and an Open Framework, and additionally ensures security, privacy and secure data. This Ecosystem suite has been detailed in depth in deliverable D5.1 of WP5.

User-demand driven interoperable IoT-enabled Active & Healthy Ageing solutions will be deployed on top of the AIoTES in every DS, enhancing and scaling up existing services, for the promotion of independent living. The AIoTES architecture also has some components that establish interoperability:

- Interoperability Layer (part of SIL) [11]: Interoperability solution at middleware level that enables syntactic interoperability among IoT platforms and AIoTES. This component provides an abstraction level at the middleware. It is provided by the H2020 INTER-IoT project.
- IPSM (part of SIL) [10]: Interoperability solution at middleware level that enables semantic interoperability among IoT platforms and AIoTES. This component provides an abstraction level at the middleware. It is provided by the H2020 INTER-IoT project.

Also, AIoTES integrates other components from WP4 that employ the SIL for acquiring data from platforms. They support management and analysis of the data shared by platforms:

- Data Lake: Storage repository that offers a common point of access for the data gathered.
- Data Analytics: The component that analyses the data collected from different IoT platforms.
- Visual Analytics: Visual means for the user to understand and interpret the data.

- AIoTES management: A web application that provides monitoring features.

In the context of ACTIVAGE system architecture, both the data management system and the information system must be compliant with the different data structures at the deployment sites. Also, there is the need for user device detection capability by using communication protocols.

In order to build a general holistic approach and to extract the main features of the IoT domain, the need of semantic models is important to face in a universal way the interoperability issue. A common data model for AIoTES is being developed and represents another key element for semantic interoperability, in addition to the SIL semantic translator (IPSM). This data model is also presented and described in this deliverable.

4 Interoperability Use Cases

This section makes an update of the interoperability use cases: from the first version of the interoperability use cases described in the previous architectural and integration deliverables (D3.2 and D5.1) and reminded in section 5.1.1, we will precise the actual use cases that are today expressed at different levels (section 5.1.2):

- Needs for interoperability at deployment site level, at application level for application developers and deployment level for installers
- Needs for interoperability at European level: gather data (performance indicators) for overall evaluation of all deployments

The section 5.1.3 is a discussion about the five presented "elementary" use cases that aims to highlight priorities for implementation among them.

See A.1. annex for details about the convention used for the description of the different interoperability use cases in this section.

4.1 Preliminary ACTIVAGE interoperability use cases

The following diagrams are used in D3.2 and D5.1 to depict the interoperability use cases.

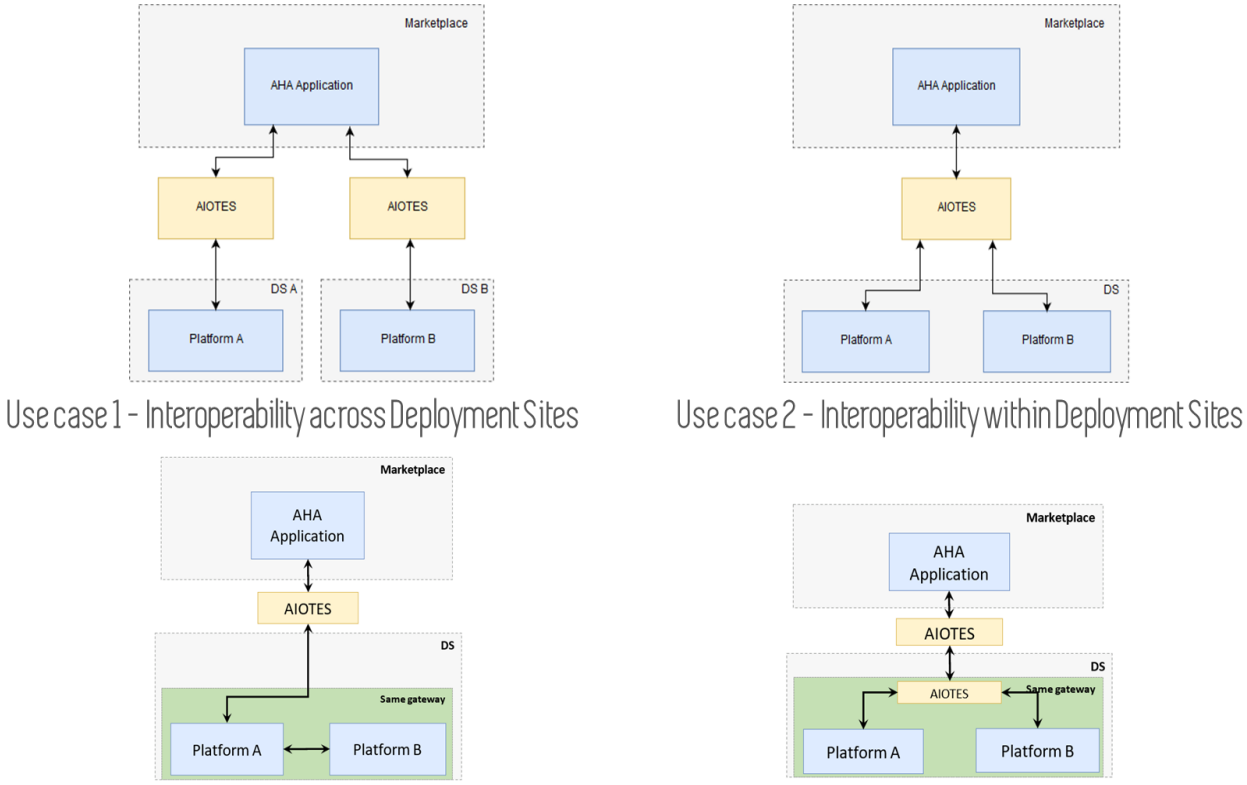

Figure 1: Preliminary ACTIVATE interoperability use case models

The first use case demonstrates how AIoTES can be used in order to provide interoperability between different Deployment Sites (DSs). Each DS may have its own computing infrastructure and deployed IoT platforms with particular data models. An Active and Healthy Ageing (AHA) application is built over these DSs and the AIoTES, which is between the AHA

Application and the DSs, provides technology adapters in the form of IoT platform bridges and wrapper components for the data of the different IoT platforms, thus enabling the abstract communication between them.

The second use case demonstrates how AIoTES can be used in order to provide interoperability between different IoT platforms within the same DS. Specifically, different stakeholders within the same DS may work on different IoT platforms without the need to migrate. An AHA application can be built over the DS that can combine the two platforms with the use of AIoTES.

The third use case demonstrates a similar scenario with the second use case. The main difference is that instead of using different gateway for each IoT platform, the two platforms are hosted in the same gateway. This case is mainly for connectivity reasons at device layer and it is supported in two ways. In the first approach, one of the two platforms (A) is connected to AIoTES. The second platform (B) is connected the first through a dedicated bridge. Through this bridge, the platform A gathers all sensor data or sends all actuator commands to platform B. In the second approach, an AIoTES local instance is deployed in the gateway and provides the interoperability bridges between platform1-AIoTES and platform2-AIoTES. Moreover, the local instance is responsible to dispatch data gathering and command sending to the right platform.

The aforementioned use cases provide a generic description of how the AIoTES can be utilized for the resolution of interoperability issues. By following a more stakeholder-centric approach and by exploiting their interoperability needs, these use cases are further extended in 5 new use cases that are explained in the following section.

4.2 The 5 ACTIVAGE interoperability use cases

From the analysis of deliverables from WP2 (requirements) and WP9 (DS descriptions in D9.1) and WP7 (needs by DS for third parties in D7.1), we extracted the 5 interoperability elementary use cases described below. The use cases are distributed among the different ACTIVAGE stakeholders.

- Stakeholders who deploy/integrate devices
    - **IUC1**: **Inside a single DS** deployed with platform A, make it possible to **use third-party HW devices** by this platform A (but supported by another platform B), **keeping unchanged native applications** running on top of this platform A.
        - EX IEUC1 (see D3.4 and D9.1): DS6 ISE installs the DS7 WOQ bed sensor in place of its current bed sensor solution in one home
- • Stakeholders who develop applications
    - **IUC2**: **Inside a single DS** deployed with a platform (A), make it possible to **use third-party applications** running on top of a platform B API
    - **IUC3: Inside a single DS** deployed with a platform (A), make it possible to **enhance existing applications** running on top of this platform A **with new functions** provided by the API of another platform B
    - **IUC4**: **Inside a single DS** deployed initially with a platform (A), make it possible to **implement new multi-platform applications** using functions provided by platform A API and functions provided by platform B API
        - EX IEUC4: an AHA application for remote health monitoring is built on top of AIoTES, and it is feed using API functions from two different

platforms. One of them collects data of a patient on a medical center, and the other collects data of the patient's monitoring at home in a DS.

- Stakeholders at European Commission who want to monitor the status of the overall ACTIVAGE project
    - **IUC5**: **At European level**, make it possible to **implement a common application** gathering all technical KPI values coming from all 7 platforms from all the 9 ACTIVAGE DSs.
        - EX IEUC5: See in D5.1: common technical KPI shared among ACTIVAGE DSs and Management (monitoring) tool

4.2.1 Interoperability Use Case IUC1

4.2.1.1 IUC1 description

Inside a single DS deployed with platform (A), it makes it possible to **use third-party HW devices** by this platform A (but supported by another platform (B)), **keeping unchanged native applications** running on top of this platform A.

4.2.1.2 IUC1 AIoTES interoperability solution

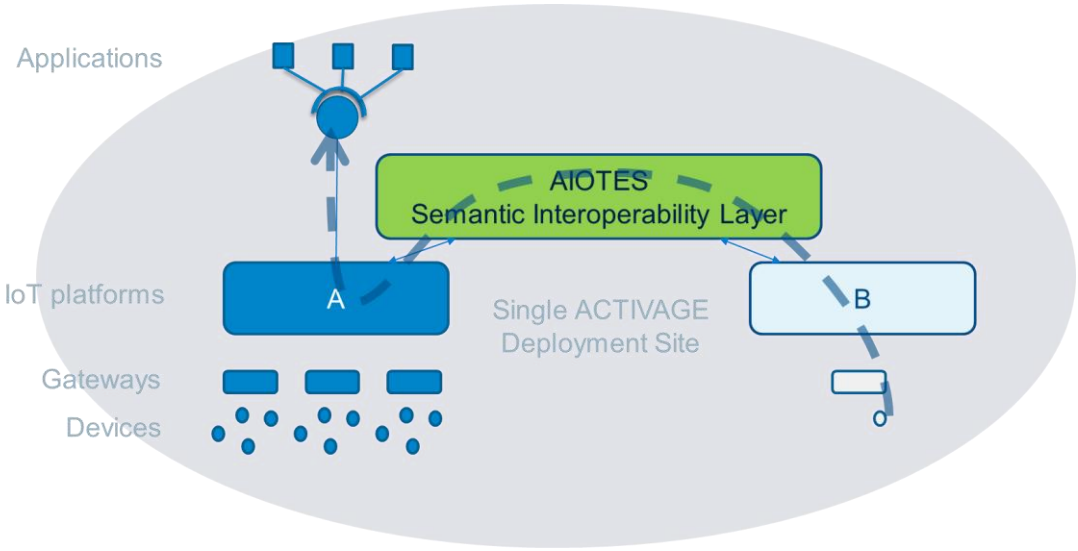

Figure 2: ACTIVAGE solution for Interoperability Use Case

1. install platform B in the DS
2. install AIoTES SIL in the DS with A-SIL bridge and B-SIL bridge
3. All platform A native IoT applications remain unchanged, data goes through the SIL

4.2.1.3 IUC1 Deployment example

This example is extracted from D9.1 and D3.4 deliverables.

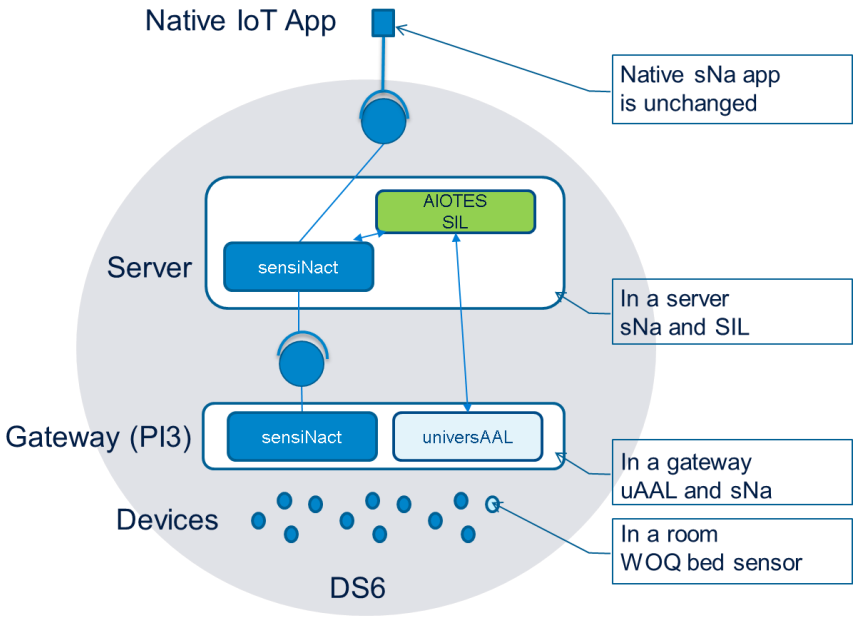

Figure 3: IUC1 Deployment example

4.2.2 IUC2 description

Inside a single DS deployed with a platform (A), make it possible to **use third-party applications** running on top of a platform B API

4.2.2.1 IUC2 AIoTES interoperability solution

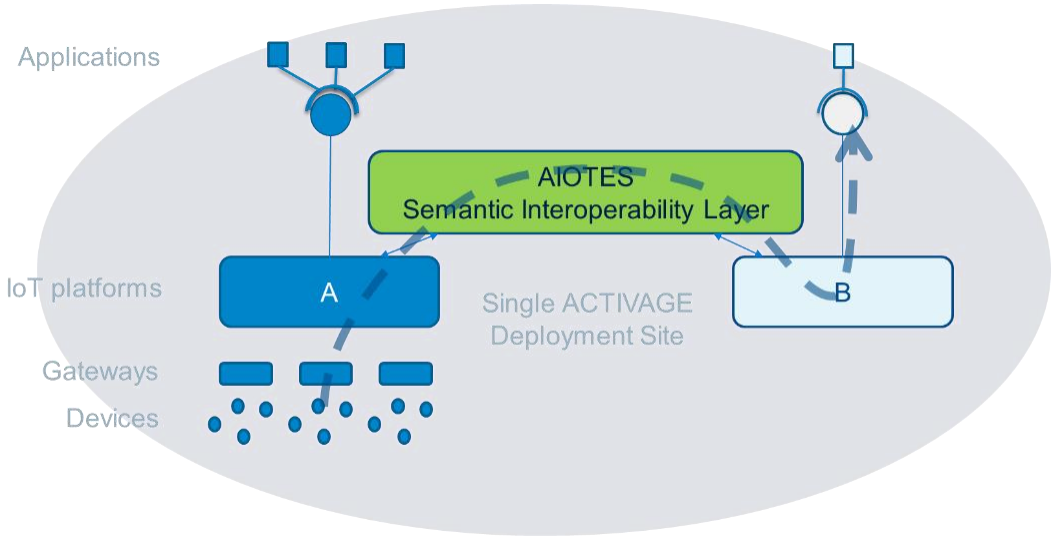

Figure 4: ACTIVAGE solution for Interoperability Use Case

1. install platform B in the DS
2. install AIoTES SIL in the DS with A-SIL bridge and B-SIL bridge
3. a platform B native IoT application remains unchanged and can run, and data goes through the SIL

4.2.3 Interoperability Use Case IUC3

4.2.3.1 IUC3 description

Inside a single DS deployed with a platform (A), make it possible to **enhance existing applications** running on top of this platform A **with new functions** provided by the API of another platform B

4.2.3.2 IUC3 AIoTES interoperability solutions

AIoTES can bring three different solutions to fulfill this Interoperability Use Case 3:

4.2.3.2.1 IUC3 AIoTES interoperability solution #1

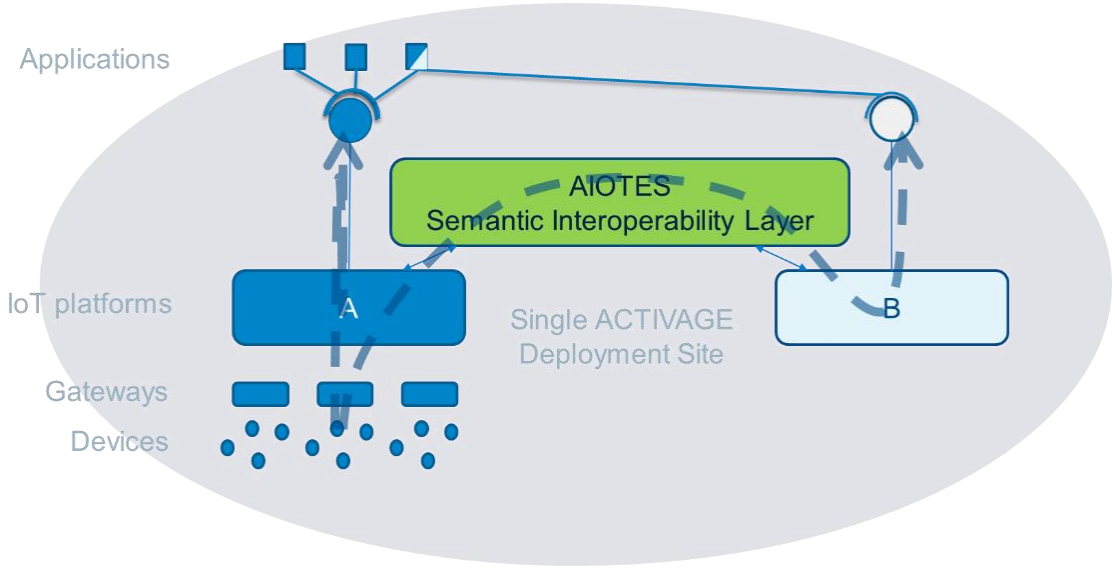

Figure 5: ACTIVAGE solution #1 for Interoperability Use Case 3

1. install platform B in the DS
2. install AIoTES SIL in the DS with A-SIL bridge and B-SIL bridge
3. modify the application using the platform B API, and data goes through the SIL

4.2.3.2.2 IUC3 AIoTES interoperability solution #2

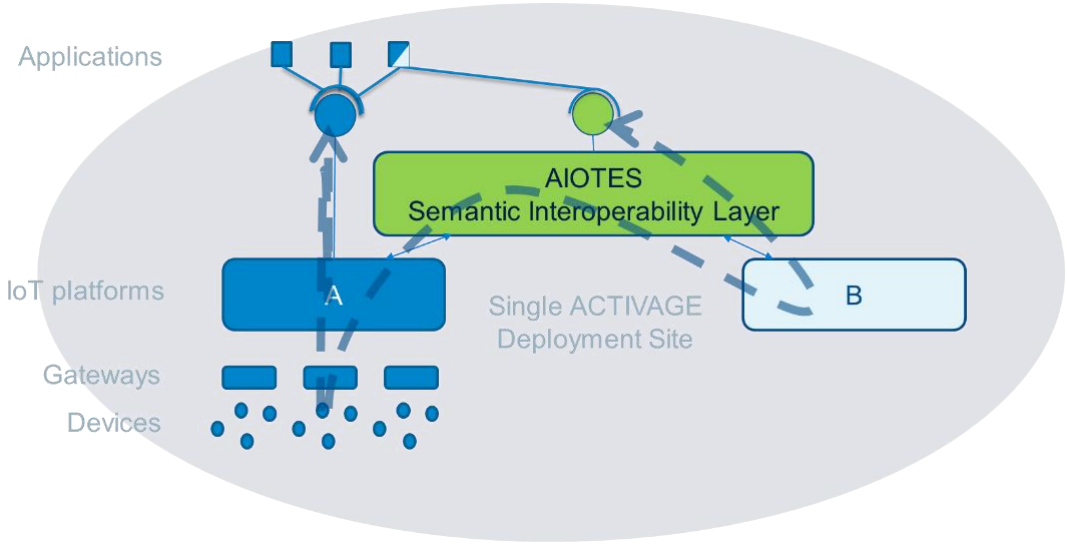

Figure 6: ACTIVAGE solution #2 for Interoperability Use Case 3

1. install platform B in the DS
2. install AIoTES SIL in the DS with A-SIL bridge and B-SIL bridge
3. modify the application using the AIoTES API, and data goes through the SIL (AIoTES API must support all platform B API functions with any translation)

4.2.3.2.3 IUC3 AIoTES interoperability solution #3

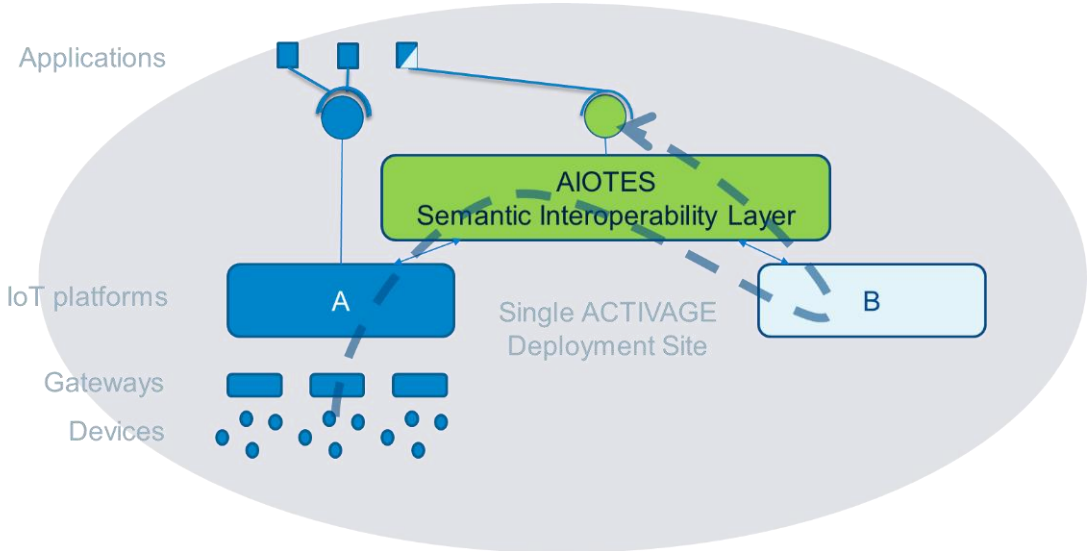

Figure 7: ACTIVAGE solution #3 for Interoperability Use Case 3

1. install platform B in the DS
2. install AIoTES SIL in the DS with A-SIL bridge and B-SIL bridge
3. refactor the application using only the AIoTES API, data goes through the SIL (AIoTES API must supports all platform A and B API functions, with any translation)

4.2.4 Interoperability Use Case IUC4

4.2.4.1 IUC4 description

Inside a single DS deployed with a platform (A), it makes it possible to **implement new multi-platform applications** using functions provided by platform A API and functions provided by platform B API

4.2.4.2 IUC4 Deployment example

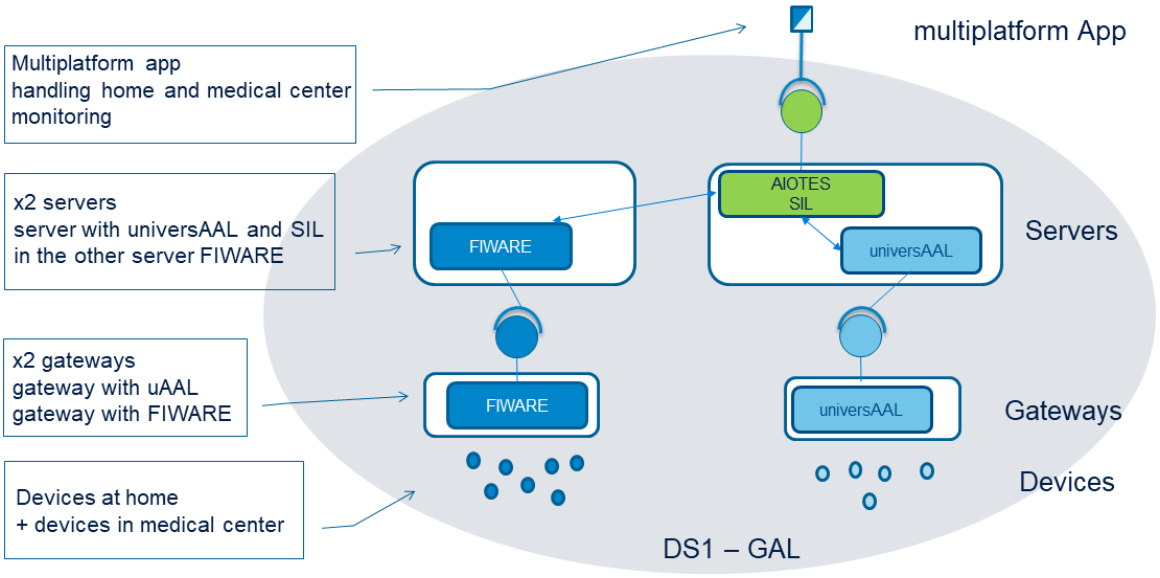

Figure 8: IUC4 #2 Deployment example

4.2.4.3 IUC4 AIoTES interoperability solutions

AIoTES can bring two different solutions to fulfill this Interoperability Use Case 4:

4.2.4.3.1 IUC4 AIoTES interoperability solution #1

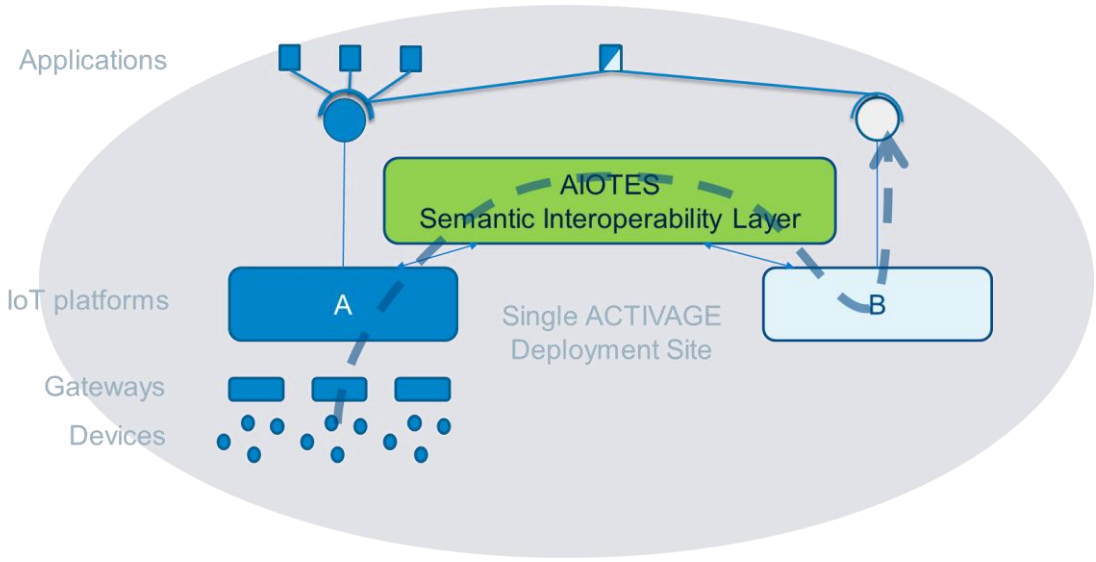

Figure 9: ACTIVAGE solution #1 for Interoperability Use Case 4

1. install platform B in the DS
2. install AIoTES SIL in the DS with A-SIL bridge and B-SIL bridge
3. implement the application using both platform B and platform A APIs, data goes through the SIL

4.2.4.3.2 IUC4 AIoTES interoperability solution #2

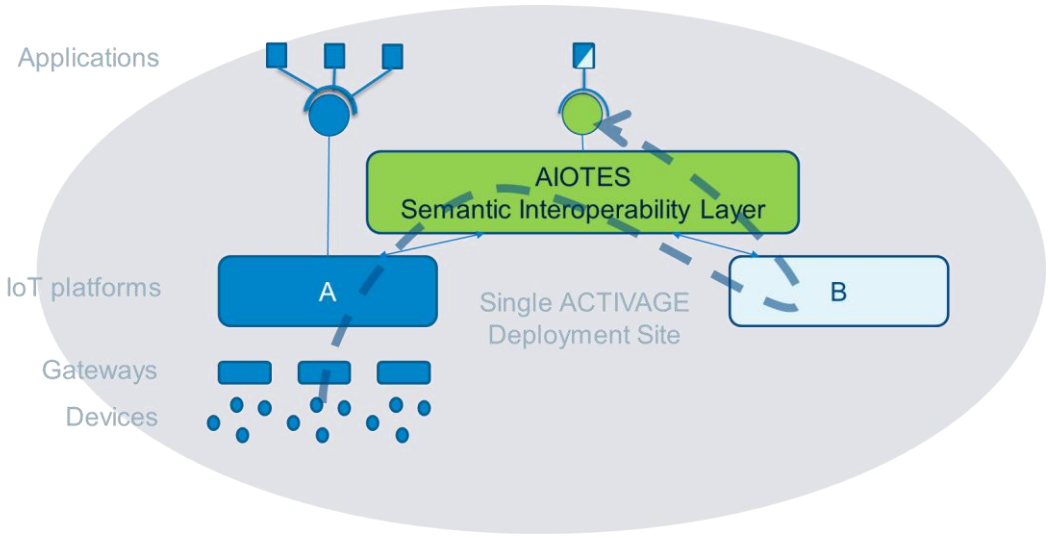

Figure 10: ACTIVAGE solution #2 for Interoperability Use Case 4

1. install platform B in the DS
2. install AIoTES SIL in the DS with A-SIL bridge and B-SIL bridge
3. implement the application using the AIoTES API, data goes through the SIL

4.2.5 Interoperability Use Case IUC5

4.2.5.1 IUC5 description

At European level, IUC5 refers to make it possible to **implement a common application** gathering all KPI values coming from all platforms from all the 9 ACTIVAGE DSs

4.2.5.2 IUC5 AIoTES interoperability solutions

This interoperability use case is extracted from D5.1, AIoTES management tool description section.

AIoTES can bring two different solutions to fulfill this Interoperability Use Case 5:

4.2.5.2.1 IUC5 AIoTES interoperability solution #1

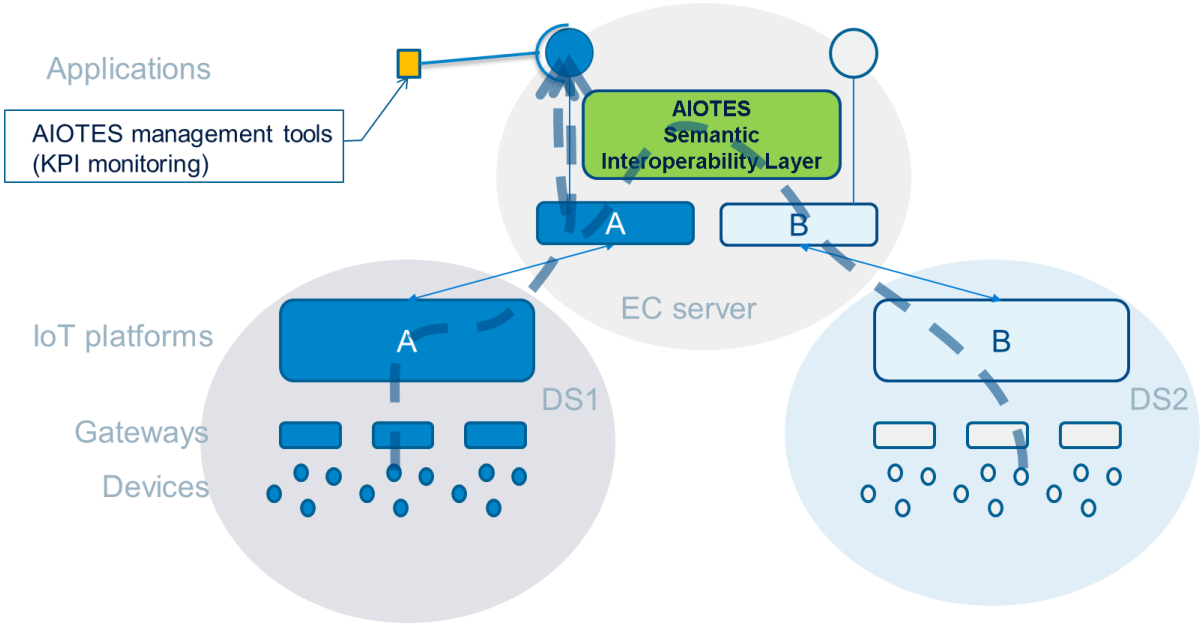

Figure 11: ACTIVAGE solution #1 for Interoperability Use Case 5

1. install the 7 ACTIVAGE platforms in the European server
2. install AIoTES SIL in the European server with all the 7 platform-SIL bridges
3. implement the application on the top of the chosen platform API, KPI data goes through the SIL

4.2.5.2.2 IUC5 AIoTES interoperability solution #2

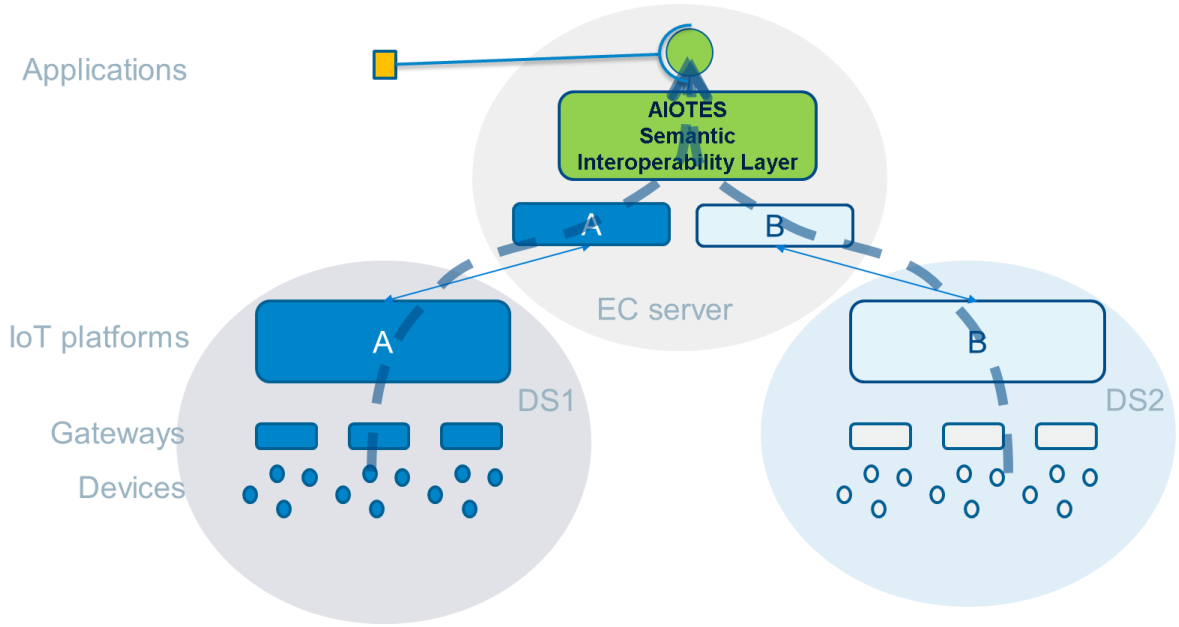

Figure 12: ACTIVAGE solution #2 for Interoperability Use Case 5

1. install the 7 ACTIVAGE platforms in the European server
2. install AIoTES SIL in the European server with all the 7 platform-SIL bridges
3. implement the application on the top of the AIoTES API, KPI data goes through the SIL

4.2.6 The 3 focused ACTIVAGE Interoperability Use Cases

It can be noticed that the preliminary interoperability use cases described in section 4.1. evolved into the 5 interoperability use cases described above

Among these 5 interoperability use cases, three have actually expected outcomes in some of the deployment sites, and at European level:

- IUC1: Give solution to handle devices from another deployment site.
- IUC4: Give solution to implement new applications running on top of several IoT platforms in a same deployment site
- IUC5: Give solution to gather in a single application all technical key performance indicators from the 9 European deployment sites and through the 7 kinds of IoT platforms

These three focused interoperability use cases are the ones that are addressed during the second year of the project, and are expected to be demonstrated in the next review.

The two remaining use cases are not cancelled, but won't be treated in the second year of the project. The actuality of them will be evaluated after the second open call.

5 Semantic Interoperability Layer

5.1 Interoperability Layer & Bridges

The Semantic Interoperability layer is composed for two elements, namely, the Interoperability layer [11] and the IPSM [10]. In this section the Interoperability layer, which has been previously introduced in D3.2 and D5.1, is described in more detail.

The Interoperability Layer enables syntactic interoperability among platforms and is provided by INTER-IoT framework (from H2020 INTER-IoT project). As was explained in D5.1, the Inter Middleware component of INTER-IoT will be used as the Interoperability Layer in ACTIVAGE.

Data is sent through the Interoperability Layer using messages expressed in the common INTER-IoT ontology (GOIoTP) [12]. Each message consists of two RDF graphs (metadata and payload). The metadata graph contains information for routing and processing the message, which includes the identifier of the message, the identifier of the sender/receiver platform, the type of message and the conversation identifier, which identifies a group of messages and allows matching a response with its corresponding request. The payload graph contains the actual data that is being sent.


```
{
  "@graph" : [ {
    "@graph" : [ {
      "@id" : "InterIoTMsg:meta/8b0d0611-06cf-4360-865b-08a2ecf90bbb",
      "@type" : [ "InterIoTMsg:Platform_register", "InterIoTMsg:meta" ],
      "InterIoTMsg:ReceiverPlatformId" : {
        "@id" : "InterIoT:universaal"
      },
      "InterIoTMsg:conversationID" : "conv42ca151c-c06f-46e3-98ab-d1bcc37a97ce",
      "InterIoTMsg:dateTimeStamp" : "2018-02-20T11:38:51.649Z",
      "InterIoTMsg:messageID" : "msg804b7c1a-6c73-455a-99df-811f6b763c5b",
      "InterIoTMsg:status" : "test"
    } ],
    "@id" : "InterIoTMsg:metadata"
  }, {
    "@graph" : [ {
      "@id" : "_:b0",
      "@type" : [ "InterIoT:GOIoTPex#UniversAAL", "InterIoT:GOIoTP#Middleware" ],
      "InterIoT:GOIoTP#hasBaseEndpoint" : "http://localhost:9000/uaal/"
    }, {
      "@id" : "InterIoT:universaal",
      "@type" : [ "InterIoT:GOIoTP#SoftwarePlatform",
"http://www.w3.org/ns/sosa/Platform" ],
      "InterIoT:GOIoTP#hasMiddleware" : {
        "@id" : "_:b0"
      },
      "InterIoT:GOIoTP#hasName" : "UniversAAL Platform"
    } ],
    "@id" : "InterIoTMsg:payload"
  } ],
  "@context" : {
    "InterIoTMsg" : "http://inter-iot.eu/message/",
    "InterIoT" : "http://inter-iot.eu/"
  }
}
```


Figure 13: Example of Message

The bridges are the components of the Interoperability Layer that connect to each individual platform. The functions of a bridge are the communication with a platform through its API and

the two-way syntactic conversion of the data between the Interoperability Layer JSON-LD and the platform's specific format.

Messages directed to the IoT platforms are processed by the corresponding bridges. If necessary, the bridge translates the information contained in the message to the data format used by the platform and generates the proper call to the platform API. A new bridge is created when an instance of the platform is registered. The bridges offer the following functions:

- Register/unregister platform
- Register/update/unregister device
- Subscribe/unsubscribe
- Query
- Send data to devices
- Handle error messages

When the platform sends data to the SIL, the bridge translates this data to the common JSON-LD format, generates a message and sends it upstream. The semantics of the translated data must be made explicit making use of an ontology or an RFD vocabulary. A bridge defines a base URI (e.g. http://inter-iot.eu/onto/syntax/FIWAREv2#) for the RDF entities that it produces when it translates data coming from the platform. In this way, the interoperability layer provides syntactic interoperability.

5.2 IPSM & Semantics

5.2.1 IPSM

The IPSM (Inter Platform Semantic Mediator) [10] provides semantic interoperability to the AIoTES Framework. The IPSM was created within the open INTER-IoT framework, and this component from INTER-IoT is included in the SIL. IPSM is able to perform ontology-to-ontology data translation among two different platforms that employ different ontologies or data models. As a result, one platform can receive data from another platform understanding the semantic meaning of this data.

Those platforms must previously register their semantic alignments, services and used ontology into the IPSM. Also, the broker connection channels with the IPSM for each specific platform-to-platform translation require previous configuration.

A more extended description of IPSM structure and functionality is provided in deliverable D5.1.

5.2.2 Universal Semantic Translation

Currently, the INTER-IoT approach for platform-to-platform interoperability is the only existing approach that provides universal semantic translation. This concept refers to the ability that any pair of platforms are able to exchange information among them –and understand the meaning of the exchanged data as the semantic model of one platform is 'translated' to the semantic model of the other platform-.

Other approaches require that any extra platform registers an alignment per each other platform in the semantic translator, thus the number of alignments grows exponentially. Thus, the number of platforms needs to be low, and the translator can only cover no more than 3 or 4 platforms at most. But the INTER-IoT approach, as it is based on translations to and from a central ontology only requires an alignment and method for each new platform, not limiting the number and combination of platforms, making universal semantic interoperability feasible.

This is achieved through a novel technique for ontology-to-ontology translation, and the use of a common modular Central Ontology, called GOIoTP. INTER-IoT semantic interoperability tools are fully configurable, flexible and work with any ontology and any devices.

5.2.3 Ontology & Semantics

IPSM employs as its central ontology for translations the ontology GOIoTP. Data from platforms is translated semantically into this ontology, GOIoTP.

Thus, data inside SIL employs the semantic model of the GOIoTP ontology.

5.2.4 IPSM API

The IPSM API is integrated into the SIL API. Its functionality is described in next section, and allows the IPSM configuration of alignments and channels.

5.2.5 CHANNELS

In order to enable the transmission of data to/from this semantic translator, it is required the establishment and configuration of broker communication channels using the IPSM.

5.2.6 ALIGNMENTS

The semantic alignments contain the information required by the IPSM to provide a translation among one platform ontology and GOIoTP. These alignments have to be previously configured within the IPSM to enable a semantic translation. This configuration is performed through the IPSM API, integrated into the SIL API.

5.3 SIL API

The SIL provides a REST API that can be accessed by other components of the AIoTES framework. This API provides a set of libraries and tools to work with the SIL by exposing all their functionalities in a unified way. This API has been defined making use of the Swagger (OpenAPI) REST API definition language and the Swagger annotation library for Java.

Each one of the SIL component (IPSM and Interoperability Layer) has its own API and together they compose the SIL API. The operations that the SIL API provides are grouped in this two groups.

5.3.1 Interoperability Layer API operations

Below are the operations that the API includes for the Interoperability Layer. These operations allow client and platform management operations, as well as device registration and data acquisition (through subscriptions).

Table 1: Client operations

Operation	Description
POST /mw2mw/clients	Register a new client
PUT mw2mw/clients/{clientId}	Update specified client
DELETE /mw2mw/clients/{clientId}	Remove specified client

Table 2: Platform operations

Operation	Description
GET /mw2mw/platforms	List all platforms registered in the SIL
POST /mw2mw/platforms	Register a new platform instance
DELETE /mw2mw/platforms/{platformId}	Remove specified platform instance

Table 3: Message operations

Operation	Description
POST /mw2mw/responses	Retrieve response messages concerning the client
POST /mw2mw/request	Send given JSON-LD message downstream towards the platform

Table 4: Devices operations

Operation	Description
GET /mw2mw/devices	List all devices registered in the SIL according to the specified filter
POST /mw2mw/devices	Register (start managing) devices
POST /mw2mw/subscriptions	Subscribe to specified devices
DELETE /mw2mw/subscriptions/{conversationId}	Unsubscribe from specified conversation

5.3.2 IPSM API operations

The following API operations allow to configure the IPSM for performing specific semantic translations among two artifacts, such as a DS IoT platform and AIoTES.

To enable a semantic translation, it is necessary to previously configure the connection channels that are going to be used, as well as the necessary alignments, which must be uploaded.

Table 5: Channels operations

Operation	Description
GET /channels	List active IPSM channels
POST /channels	Create new channel
DELETE /channels/{channelId}	Delete channel based on the ID

Table 6: Alignments operations

Operation	Description
GET /alignments	List alignments
POST /alignments	Upload new alignment
DELETE /alignments/{name}/{version}	Delete alignment identified by name+version
GET /alignments/{name}/{version}	Get an alignment identified by name+version

Table 7: Translation operations

Operation	Description
POST /translation	Translate JSON-LD message via sequence of alignments

Table 8: Logging operations

Operation	Description
GET /logging	Get the current IPSM logging level
POST /logging/{level}	Set logging level

6 AIoTES Data Model

Another element for interoperability in the ACTIVAGE project is the use of a common data model in AIoTES and all DS platforms focused on the AHA domain. The design and development of this data model is currently an ongoing task. As a starting point, the SIL and AIoTES are employing the INTER-IoT data model, which will be extended and modified until being transformed on this common data model.

Semantic interoperability can be described as the way to facilitate exchange and virtual connectivity between multiple computing or communications systems using the meta-data capacity of information systems to simplify and facilitate the process. In the context of the Internet of things systems Semantic Interoperability reside on the connectivity of IoT platforms and their way to collect the information.

The creation of a common data model is a technique for the enablement of semantic interoperability among diverse systems that employ different vocabularies and ontologies, such as the DS platforms and AIoTES. Then, the information sent by platforms will be semantically understandable in AIoTES, as far as the meaning will be already included in the common data model employed.

In this sense the common data model will be used conjointly with the IPSM translation to provide semantic interoperability. Currently, IPSM [10] is a very powerful platform-to-platform semantic translator, that solves the scalability problems (i.e. adding new platforms and ontologies) of the use of ontology-to-ontology translators. It also represents a solution for providing semantic interoperability among diverse entities by translating semantically the messages among them, in such a way that the messages would be understandable for the receiver system or platform. In ACTIVAGE it was decided to employ conjointly both approaches, the creation of a common data model and the IPSM translation, to get advantage of the benefit that each of them provide. Thus, the IPSM will and require only one type of semantic alignment, and the data model employed in AIoTES will be this common data model.

This section presents the ACTIVAGE approach for the design and development of a common data model, background information on semantics and domain modelling, and future work on this direction, in order to finalize the development of the AIoTES data model and the creation of an AHA ontology.

6.1 Domain Modelling

In this section we analyse existing data models, data formats and ontologies that can be used as the basis of the ACTIVAGE data models and extend them as needed. In ACTIVAGE we subdivide the work of data modelling into four domain areas:

- IoT platforms/systems and services where sensors and sensor measurements are the focus, Active and Healthy Ageing (AHA) services where end-user applications are the focus,
- Data Security and Privacy systems and services where the IoT platforms/systems device protection and access control and data protection and privacy preservation are the focus aspects and last but not least
- Healthcare Information Systems Support.

For each of these areas we discuss which ontologies are used for modelling the required data (and why), present necessary additional data items and how to model them. The presented data models are not final and/or static. Instead, in the time until the second version of this

deliverable we will fine-tune the data models, incorporate practical experiences and new requirements, and integrate new data items that we identified during the project lifetime.

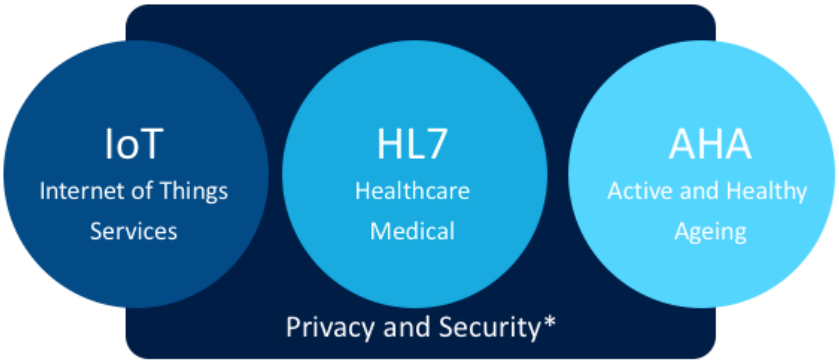

Figure 14: ACTIVAGE Domain Areas for Data Modelling

This document describes/specifies the data models, formats and ontologies used in the ACTIVAGE project. Following the main interoperability requirements in ACTIVAGE these models, formats and ontologies are intended to be used in four different ways.

- First, they are intended to be used by AHA applications and information services to communicate meta-data between them and the ACTIVAGE AIoTES framework, e.g. querying registered data resources or also available data types.
- Second, they are used by the so-called AIoTES framework interface (API), which specifies an interface between AHA applications and AIoTES information services/systems that should be integrated into AIoTES Architecture, e.g. a description of the services offered by the IoT system at a deployment site.
- Third, they are used inside the AIoTES framework to communicate data between AIoTES framework services, e.g. security level information, management, data storage services, etc.
- Fourth they are used by AHA IoT-enabled systems external to AIoTES framework and that mostly resided in Deployment Sites. This third use is intended to enhance interoperability capacity of a deployment site technology to offer enhanced services and increase ecosystem capability to interoperate with other technologies.

Due to the diversity and provided advantages about the use of a data model for the different stakeholders (set of users) and as result from the multiple use cases, ACTIVAGE has to cover a wide application of the data modelling requirements, that are shortlisted as follow:

- ACTIVAGE data model(s) has to specify metadata for the AIoTES framework and its components, e.g. to model security, management and monitoring information.
- ACTIVAGE data model(s) has to identify data models for various AHA domain specific applications, e.g. for home monitoring, emergency situations, social isolations, etc. and
- ACTIVAGE data model(s) has to specify from how basic IoT sensors and sensor measurements are modelled, e.g. referring and using the Semantic Sensor Network (SSN) ontology, Semantic Observations and Semantic Actuation (SOSA). ACTIVAGE has to define how to describe IoT systems and IoT services, e.g. using the Minimal Service Model (MSM).

The data models are intended for:

- users of the IoT-enabled platform at deployment sites levels that want to learn about the used data models to develop AHA interoperable applications,
- IoT platform providers that want to learn how to integrate their platform data with AIoTES services and global applications,
- the partners of the ACTIVAGE consortium, allowing them to develop their components in a way that ensures easy data sharing and data interoperability,
- external researchers and developers that want to design and/or use data models in multiple AHA domains beyond ACTIVAGE, as well as
- the project reviewers to better understand the work done in the project in terms of data modelling.

6.2 Data Model for AHA

6.2.1 Multi-domain & Cross-Layer Data Model for AHA

Developing a multi-domain data model is not a trivial task and great efforts and conventions are need to be taken, in ACTIVAGE we are taking the approach to build bets practices-driven multi-domain data model and for this we are collecting the most used (according to standards when applicable, otherwise dictated by the most extended in industrial practices and use).

A common communication standard was tried in the past with the OTA message specification in the traveling domain [13] a standard consisting on a set of (XML-demarcated) messages; or in the healthcare domain (related to the INTER-Health use case) with OpenEHR [14], which is an open domain-driven platform for developing flexible e-health systems. Here, multiple projects strive to establish interoperability between already known standards and the OpenEHR, e.g., establishing semantic interoperability of the ISO EN 13606 and the OpenEHR archetypes [15]. Similarly, the Think!EHR Platform (health data platform based on vendor-neutral open data standards designed for real-time, transactional health data storage, query, retrieve and exchange) [16]; aims at establishing interoperability of the OpenEHR and the HL7 standard (a framework for the exchange, integration, sharing, and retrieval of electronic health information). Interestingly, development of the Think!EHR Platform had to deal with the data standards problem caused by existence of HL7 RIMv3 [17], ISO13606, and OpenEHR standards. While it is possible to envision an approach similar to this, applied to individual domains, it is not likely to be easily generalizable to support interactions between domains. In ACTIVAGE we selected (according to domain specific) those best semantic interoperability models that has been designed, implemented and deployed in order to support the expected innovation level from the multiple deployment sites and their stakeholders in the ecosystem.

6.2.1.1 GOIoTP (Data Interoperability)

This generic ontology of IoT Platforms (GOIoTP) [10] [12] was developed in the frame of the INTER-IoT project, and it is the core ontology for semantic translations of the IPSM and SIL. In principle, it is a starting point for the construction of the ACTIVAGE ontology and the creation of AIoTES common data model, that will be extended for the creation of an AHA ontology.

GOIoTP offers modular data structures for description of entities most commonly appearing in IoT in the context of interoperating various different IoT artifacts (platforms, devices, services, etc). At the same time, GOIoTP [12] is the reference meta-data model proposed by INTER-IoT.

While reusing existing standardized and established ontologies used in IoT (most notably SSN and SOSA), GOIoTP proposes its own extensions, in order to provide a complete, modular ontology, useful in a wide range of IoT use cases.

GOIoTP is a core ontology, and, as such, it defines some stub classes - i.e. classes that do not have a robust definition, or subclasses. This is a design decision intended to enable different extensions, for those stub concepts, depending on particular needs of an implementation, and follows the usual design patterns for top-level ontologies. Other than offering less restrictions on implementers, stub classes are also easier to align to, and from, other core IoT ontologies.

In order to offer a more complete interoperability solution, it is also defined an extension to GOIoTP that expands the definitions and "stubs" in GOIoTP taxonomy. GOIoTPex is the multi-module ontology that extends the reference model of GOIoTP with concrete entities. It completes GOIoTP and is suitable for AHA and e-Health scenarios, in which a user is looking for a complete modeling solution.

GOIoTP and GOIoTPex import or use parts of the following ontologies:

- SSN/SOSA - These ontologies are imported as a whole, and are the basis of GOIoTP. They have the strongest impact on the device and observation modules, as those modules build directly on top of structures provided by SSN/SOSA.
- GeoSPARQL - The GeoSPARQL ontologies, specifically base, functions, and rules, used in the location module.
- NASA SWEET units - These subset of SWEET ontologies, specifically units, mathematical operations, representation, mathematical relations, and scientific relations are used to model concrete instances and types of units in the units and measurements module.
- Various provenance properties and classes from well-known ontologies, such as Dublin Core, FOAF, and Semantic Web Status, are used to annotate the GOIoTP and GOIoTPex ontologies themselves

6.2.1.2 Big IoT (Marketplace & Business APIs)

The BIG IoT Marketplace is a B2B (Business to Business) broker for trading access to IoT Information and Functions. Human actors are involved in Marketplace interactions and are defined as follow: a) Marketplace Operator operating the BIG IoT Marketplace itself, b) Platform Operators operating a BIG IoT Platform, c) Service Operators operating a BIG IoT Service and d) Application Operators operating a BIG IoT Application.

BIG IoT Platform implements new terminology such as a Provider, and a common API, which is called the BIG IoT API, to register offerings on a BIG IoT Marketplace, and grants BIG IoT Services or Applications (as Consumers) access to the offered Resources. BIG IoT platform makes use of application software that implements and uses the BIG IoT API, (as a Consumer) to discover offerings on a BIG IoT Marketplace, and to access the resources provided by one or more BIG IoT Services or Platforms (as Providers). The BIG IoT Services implements and uses the BIG IoT API to register offerings on a BIG IoT Marketplace (as a Provider) and/or to discover and access Offerings provided via a BIG IoT Marketplace (as a Consumer).

The BIG IoT Marketplace is a composite system consisting of sub-components: The Marketplace API serves as an entry point for all communications and interactions with the Marketplace; the Identity Management Service (IdM) which authenticates and authorizes providers and consumers; the Exchange, which allows registration and discovery of offerings using semantic technologies; the Web Portal for users of the Marketplace; and the Charging Service, which collects accounting information.

BIG IoT Lib implements the BIG IoT API that supports platform, service and application developers. The BIG IoT Lib consists of a Provider Lib and a Consumer Lib part. It translates function calls from the respective application or service logic, or the platform code into

interactions with the Marketplace, or peer Services or Platforms. The Provider Lib allows a Platform or Service to authenticate itself on a Marketplace and to register Offerings. The Consumer Lib allows an application or service to authenticate itself on a Marketplace, to discover available Offerings based on semantic queries, and to subscribe to Offerings of interest. The use of semantic technologies enables the Exchange to perform semantic matching even in case providers and consumers use different semantic models or formats, as long as a common meta-model defines the relations/mapping between the different semantic models and converters for the different semantic formats are supported.

6.2.1.3 OpenIoT (Sensors to Cloud Services)

The sensors or sensing devices in general are considered the most relevant instances of the Internet-connected objects area, more widely known as the Internet of Things. In the focus of OpenIoT project an ontology extending from the W3C SSN ontology [18] as its core part was developed in order to cope with the extensively expressed need to address cloud services. OpenIoT ontology incorporated components descriptions (such as Sensor, ObservationValue and FeatureOfInterest), and extended it by adding missing concepts required for the OpenIoT-specific requirements.

OpenIoT ontology introduces new concepts that haven't been covered by the existing vocabularies and may be of interest for the FIESTA-IoT ontology. Such concepts can be explored towards adding them in FIESTA-IoT ontology. These concepts include the notion of Virtual sensors and Utility metrics as described below. Additionally to existing vocabularies that have been reused, Virtual sensor [19] – the basic concept in Global Sensor Network (GSN) that is one core element of the OpenIoT platform – represents new data sources created from live data. These virtual sensors can filter, aggregate or transform the data. From an end-user perspective, both virtual and physical sensors are very closely related concepts since they both, simply speaking, measure data. Therefore, the concept of a virtual sensor is as a subclass of the sensor concept as defined in the SSN ontology.

In OpenIoT, utility metrics are used in a variety of utility-based algorithms for resource management, utility-driven privacy and utility-driven security mechanisms. In addition, utility metrics serve as a basis for accounting and management of Service Level Agreements (SLA) between the OpenIoT services and end users.

6.2.1.4 FIESTA-IoT (Services Platform Federation)

FIESTA-IoT has defined a series of concepts to federate multiple IoT platforms [20], from the point of view of meta-data being used to control the execution of data services. These concepts such as the Resource, the Virtual Entity and the IoT Service are complementary to other core services descriptions. According to these definitions [21] a Resource is a "Computational element that gives access to information about or actuation capabilities on a Physical Entity", A Virtual Entity is a "Computational or data element representing a Physical Entity" and an IoT service is a "Software component enabling interaction with IoT resources through a well-defined interface. Those terms can be also orchestrated together with non-IoT services (e.g., enterprise services, healthcare). Interaction with the service is done via the network and in FIESTA-IoT, the aim is to address and maximize interoperability as much as possible.

In FIESTA-IoT, the Resources are mainly related to Sensor, Actuator or Tag hosting devices and from the analysis of the ontologies that relate to IoT, SSN stands out by far as a well-adopted ontology, and hence should serve as the base of an interoperable ontology. The "Resource" concept is adopted in FIESTA-IoT and SSN adopts a more device-centric approach. It could be argued that the closest property in SSN that resembles the FIESTA-IoT Resource is "Process". In FIESTA-IoT, this property is used in a different context, in the Service sub-model. FIESTA-IoT specifies that a Resource is hosted on a Device, although no

information model has been provided for the Device. This is where SSN can play an important role in the FIESTA-IoT ontology. The Device concept can be adopted so that a FIESTA-IoT Resource is hosted on a ssn:Device. This *could* be made using an explicit property e.g. "isHostedOn". In this case, we need to define a "Resource" concept although this concept will not provide any added value to the information, especially upon querying it.

FIESTA-IoT consider that multiple Resources can be hosted on a single Device. Similarly, in SSN a Device can be made up of multiple smaller Devices (sensors). On this basis, an implicit link (without annotation) between a Resource and a Device can be made, whereby one Resource is hosted on one Device and hence can be treated as one "entity". Another aspect to consider is that the Device concept in SSN has a subclass that focuses on Sensing, i.e. the SensingDevice. However, currently SSN only addresses sensing aspects even though IoT has other aspects such as actuation and identification. Therefore, like for the FIESTA-IoT Resource, the SSN Device should have other instantiations for Actuators and Tags. This is where the FIESTA-IoT ontology can play an essential role include federated concepts.

6.2.1.5 SSN/SOSA Updates

Recently, an updated version of SSN ontology was made available as a working draft defining SOSA[2]. This version has many updates. For instance, some updates are related to: (a) separation of DUL ontology from SSN ontology (b) movement of `ssn:Sensor` subclass from `dul:PhysicalObject` to `dul:Object` and (c) the introduction of sosa:actuator subclass including all its the types and properties. However, as this is still a working draft and not a final release we have refrained ourselves in changing to the new namespace. As the ACTIVAGE Data Model is a live data model, we will consider modifying ACTIVAGE core ontology with the new namespace of SSN to SOSA as soon as it is released. Changing the SSN namespace and modifying the FIESTA-IoT ontology will have a widespread impact. All the testbeds would be required to modify their annotator to reflect the changes. Further, if they are already using SSN ontology to store their proprietary data, we assume that they would have already modified their process. Similarly, experimenters would have to modify their queries to include new namespace. Moreover, from the ACTIVAGE AIoTES framework side, already available data that was made available before the change would not be accessible as it uses old SSN namespace. Further, we also need to assess the changes in the imported SSN concepts in the ACTIVAGE core ontology. Thus, as this change in the namespace has a huge impact and we will follow carefully and closely the evolution not only of this but all other ontologies.

6.2.1.6 SAREF ontology and OneM2M base ontology Mappings

The Smart Appliances REFerence[3],[4] (SAREF) [6] ontology is designed for household and home appliances in residential buildings, especially for the purpose of energy management. SAREF aims to align existing ontologies in the domain of smart appliances. Many standards have been proposed to enable the interoperation of appliances from diverse vendors. However, the number of standards is so high that overlapping is inevitable. To address this problem, the European Commission launched a study for the purpose of proposing a reference ontology gathering the efforts of existing appliances standards relevant for energy efficiency. The final result of this study is the SAREF reference ontology [22] that is intended to be transferred to European Telecommunications Standards Institute (ETSI) Smart Machine to

[2] https://www.w3.org/TR/vocab-ssn/

[3] https://sites.google.com/site/smartappliancesproject/ontologies/oma-lightweight_m2m-ontology

[4] http://ontology.tno.nl/saref/

Machine (SmartM2M) that could contribute it to International Machine-to-Machine Standardization (oneM2M) initiative.

The oneM2M base ontology and the SAREF ontology are both standardized reference ontologies (oneM2M TS0012[5] [23] and ETSI TS103264[6] respectively) with a good visibility in the market. They were created and designed with a different aim:

- OneM2M base ontology aims to provide a high-level ontology for the IoT market in order to provide a minimal set of common knowledge that enables the cross-domain syntactic and semantic interoperability. As it is quite high-level and abstract, oneM2M expects external ontologies that describe a specific domain of interest in a more detailed way to be mapped to the oneM2M base ontology. With these mappings of different domain-specific ontologies to oneM2M base ontology, the internetwork between devices and things from different domains is enabled.
- SAREF aims to provide a common knowledge for the domain of Smart Appliance, especially on the energy consuming aspect. Compared to oneM2M base ontology, it is less high level and more applicable to describe devices.

The mapping between oneM2M base ontology and SAREF is performed by oneM2M WG5 MAS (Working Group 5 Management, Abstraction and Semantics). The working group who has drafted the TS0012 in which 4 relationships have been specified for mapping external ontologies to oneM2M base ontology: rdfs:subClassOf, owl:equivalentClass, rdfs:subPropertyOf, owl:equivalentProperty. Basically, SAREF is sub-classed to oneM2M base ontology. The following table provides a list of mapped classes using subclass relationship.

Table 9: Subclass mapping between SAREF and the base ontology

Class in SAREF	Mapping relationship	Class in oneM2M Base Ontology
saref:Device	rdfs:subClassOf	oneM2M:Device
saref:BulidingObject	rdfs:subClassOf	oneM2M:Thing
saref: BulidingSpace	rdfs:subClassOf	oneM2M:Thing
saref:Command	rdfs:subClassOf	oneM2M:Command
saref:Commodity	rdfs:subClassOf	oneM2M:Thing
saref:Function	rdfs:subClassOf	oneM2M:Functionality
saref:Property	rdfs:subClassOf	oneM2M:InputDataPoint OR oneM2M:OutputDataPoint
saref:Service	rdfs:subClassOf	oneM2M:Service
saref:UnitOfMeasure	rdfs:subClassOf	oneM2M:MetaData
saref:ActuatingFunction	rdfs:subClassOf	oneM2M:ControllingFunctionality
saref:MeteringFunction	rdfs:subClassOf	oneM2M:MeasuringFunctionality
saref:SensingFunction	rdfs:subClassOf	oneM2M:MeasuringFunctionality
saref:State	rdfs:subClassOf	oneM2M:InputDataPoint OR oneM2M:OutputDataPoint

[5]http://www.onem2m.org/images/files/deliverables/Release2/TS-0012-oneM2M-Base-Ontology-V2_0_0.zip

[6] http://www.etsi.org/deliver/etsi_ts/103200_103299/103264/01.01.01_60/ts_103264v010101p.pdf

saref:Profile	rdfs:subClassOf	oneM2M:Thing
saref:Task	rdfs:subClassOf	oneM2M:Thingproperty
saref:DeviceCategory	rdfs:subClassOf	oneM2M:Thingproperty
saref:FunctionCategory	rdfs:subClassOf	oneM2M:Aspect

6.2.2 AHA-Related Ontologies

The objective of the ACTIVAGE project is to address the need to integrate Active and Healthy Ageing (AHA) Platform is to integrate existing tools, hardware, and software that assist individuals in improving and/or maintaining a healthy lifestyle. This architecture is realized by integrating several hardware/software components that generate various types of data. Some examples include heart-rate data, coaching information, in-home activity patterns, mobility patterns, and so on. Various subsystems in the AHA platform can share their data in a semantic and interoperable way, through the use of AHA data-store and specific sets of terminologies.

Active and Healthy Ageing (AHA) is a major societal challenge, common to all populations. The interrelationships between healthy biological ageing and wellbeing with sex/gender, ethnicity, socio-economic factors and other lifetime determinants require further study, with an interest in individuals with discordant profiles (e.g. those maintaining psychological wellbeing and social participation despite functional decline). Strategies for AHA allow people to realize their potential for physical, social (economic, cultural, spiritual and civic affairs) and mental wellbeing across life.

6.2.2.1 AHA Wearables

ACTIVAGE project deals with devices that are not fixed in a particular geographical location/position, those devices that are portable and which main function is provide and consume data are called wearables and the most common required use case that ACTIVAGE is in need to support is the data exchange or Data Interoperability on Ambient Assisted Living Application(s). In AHA it has been proposed an ontology for wearable data interoperability in Ambient Assisted Living environments. The purpose is serving application development in Ambient Intelligence scenarios ranging from activity monitoring and smart homes to active healthy ageing or lifestyle profiling. In order to provide the AHA Platform with sensor data interoperability, an AHA platform ontology that includes variables and features to measure vital signs or other physical and cognitive abilities for personalized health has been introduced.

By integrating data from different wearable devices and sensors, a set of dimensions or features were selected to be modelled as entities, data properties (literals) and object properties (class-to-class relations). The AHA ontology design consists of a formal specification for the height and weight of the person, the geographical location at which the person is, the logical place at which the person is and logical activity that the person is employing. Other features modelled are the amount of physical activity (exertion) a person is employing, determined at a specific position on the person's body. For each position, maximally one activity count applies at any specific moment. The energy expenditure from physical activity that the person has employed, the heart rate and stress level of the person, the valence of the person, ambient light and temperature that the person is exposed to. More properties are the skin temperature of the person and acceleration at a specific position on the person's body (for each position, maximally one acceleration and skin conductance apply at any specific moment).

6.2.3 Healthcare-Related Ontologies

Health Level-7 or HL7 [24] refers to a set of international standards for transfer of clinical and administrative data between software applications used by various healthcare providers. These standards focus on the application layer, which is "layer 7" in the OSI model. The HL7 standards are produced by the Health Level Seven International, an international standards organization, and are adopted by other standards issuing bodies such as American National Standards Institute and International Organization for Standardization.

Hospitals and other healthcare provider organizations typically have many different computer systems used for everything from billing records to patient tracking. All of these systems should communicate with each other (or "interface") when they receive new information, or when they wish to retrieve information, but not all do so.

HL7 International specifies a number of flexible standards, guidelines, and methodologies by which various healthcare systems can communicate with each other. Such guidelines or data standards are a set of rules that allow information to be shared and processed in a uniform and consistent manner. These data standards are meant to allow healthcare organizations to easily share clinical information. Theoretically, this ability to exchange information should help to minimize the tendency for medical care to be geographically isolated and highly variable.

The HL7 Reference Information Model (RIM) [25] is the cornerstone of the HL7 V3 messaging design and development. The RIM model has six core classes: Act, ActRelationship, Participation, Roles, RoleLinks, and Entity. Everything happening in the domain is an Act. The Act is further specialized to define other acts such as Medications, Procedures, Observations, etc. The ActRelationship connects two or more Acts using relations such as composition, precondition, revision, etc. Participation defines the context for an Act. The participants are assigned a Role such as a patient, provider, practitioner, specimen, employee, etc. Multiple roles involved in the Act are related using RoleLink. Finally, Role is assigned to Entity such as a person, organization, place, etc. In summary, HL7 V3 messaging standard is based on the HL7 Reference Information Model with six backbone classes with well-designed hierarchies and abstraction. The HL7 V3 standard was designed using fundamental software modeling principles and formal object-oriented methodology primarily targeting biomedical informatics experts.

6.2.3.1 HL7 Medical Records

Interoperability is one of the most essential requirements for health care systems to reach the benefits promised by adopting HL7-based systems and Electronic Medical Records (EMRs). There are significant numbers of methodologies and architectures developed to address the issues of interoperability of the coalition's systems in recent years. Slavov et al. proposed an HL7-compliant data exchange software tool called Collaborative Data Network (CDN) aiming for clinical information sharing and querying. The clinical documents in CDN are modeled in compliant with HL7 v3 standard and encoded in eXtensive Markup Language (XML) format, which can be ultimately deployed in a cloud environment to support large-scale management and vast amounts of clinical data sharing.

The integration of HL7 standards and ontology technology has been widely applied in supporting system interoperability among applications in the medical domain. By assimilating HL7-compliant clinical message with ontology is possible to facilitate the flow of patient information across health care organizations. To ensure the interoperability of electronic health data (for fitness and health screening) in an ontological knowledge engine, a uniform medical information standard Health Level Seven (HL7) was adopted. As a global authority on standards for the sharing, integration, and retrieval of electronic health information, HL7 responds to the increased demand in healthcare interoperability that enhances care delivery, optimizes workflow and augments knowledge transfer.

6.2.3.2 HL7v3 Extension for Physical Activity

In general, Physical Activity (PA) is defined as any body movement that works your muscles and requires energy expenditure. It is proven that 30 minutes of daily physical activity can significantly reduce a variety of chronic diseases, improve mental health and overall well-being of an individual. The challenges in the physical activity domain are primarily due to lack of standards, both structural and semantic, for representing and sharing physical activity data. The idea is to use existing healthcare standards and other standard models/vocabulary to represent and share physical activity data across diverse healthcare systems. This can be done by extending the HL7 Reference Information Model (RIM) [25] with health and life sciences entities developed by extending Schema.org model (Rishi Saripalle). For example, entities such as: PhysicalActivity, ExercisePlan and Diet, where ExercisePlan isa PhysicalActivity which in turn isa MedicalEntity (part of Health and Lifesciences vocabulary), can be adapted into RIM model.

6.2.3.3 FHIR (Fast Healthcare Interoperability Resources)

Healthcare records are increasingly becoming digitized. As patients move around the healthcare ecosystem, their electronic health records must be available, discoverable, and understandable. Further, to support automated clinical decision support and other machine-based processing, the data must also be structured and standardized. (See Coming digital challenges in healthcare)

HL7 has been addressing these challenges by producing healthcare data exchange and information modelling standards for over 20 years. FHIR [26] is a new specification based on emerging industry approaches, but informed by years of lessons around requirements, successes and challenges gained through defining and implementing HL7 v2, HL7 v3 and the RIM, and CDA. FHIR can be used as a stand-alone data exchange standard, but can and will also be used in partnership with existing widely used standards. (See Comparing FHIR to other HL7 standards)

FHIR aims to simplify implementation without sacrificing information integrity. It leverages existing logical and theoretical models to provide a consistent, easy to implement, and rigorous mechanism for exchanging data between healthcare applications. FHIR has built-in mechanisms for traceability to the HL7 RIM and other important content models. This ensures alignment to HL7's previously defined patterns and best practices without requiring the implementer to have intimate knowledge of the RIM or any HL7 v3 derivations.

The SemanticWeb/RDF FHIR implementation is jointly maintained by the HL7 FHIR project and the W3C Semantic Web Health Care and Life Sciences Interest Group. FHIR resources can be represented as an RDF graph serialised in the Turtle format, or as JSON-LD. The RDF format is defined to assist the process of bridging between operational data exchange and formal knowledge processing systems. While the RDF form offers a fully functional representation of FHIR resources, it has different operational characteristics to the JSON and XML representations, and would be implemented for different reasons. Systems focused on operational exchange of data would not generally use choose to use RDF. In addition to the basic representation of HL7 resources, a RDF representation of the FHIR infrastructure and definitions is published for the following purposes:

- Providing the class definitions to support RDF based representation of resource instances
- Supporting knowledge-based analysis of the FHIR specification itself
- Providing knowledge of use at run-time for converting between FHIR and other content models

- Supporting reasoning across the information/terminology model boundary

6.2.4 Security-Related Ontology

6.2.4.1 HL7 Security

As part of the HL7 there is a specification dedicated to the Healthcare Privacy and Security Classification System (HCS) [27] that includes the type of data and the formats. In order to implement this correctly, there is a Classification System Guide (HCSG) that provides informative material to be of use for implementing the HL7 Health Care. HSCG describes a Healthcare Privacy and Security Classification System (HCS) suitable for automated privacy and security label "tagging" and segmentation of protected health information (PHI) for privacy policy enforcement through security access control services (ACS) within a security domain, these requirements are fulfilled by information classification system guidance issued by domain authorities. Security labels are key access control information (ACI) applied by policy within a security domain as attributes of security principal "Initiators" requesting access to information and system resources; the resource "Targets" for which access is sought; the request and the context in which it is made; and the asserted policy rationale, e.g., purpose of use, for access.

Security labels shall be part of the data format in ACTIVAGE and used to be standardized and computable where semantic interoperability is required for electronic cross-enterprise exchange and to enforce and document policy compliance. The clinical labels and attributes applied to clinical data or also called clinical facts are expected to be slow changing and relatively static for an instance of clinical fact retained within the ACTIVAGE Data Repository. Accordingly, while these rules for applying clinical attributes may change over time, such change is unlikely to have significant impact on the day to day use of clinical information retained within the ACTIVAGE data repository.

On the other hand, security and privacy rules shall be implemented by policy rules that are relatively dynamic and depend upon a number of factors external to the clinical facts themselves such as the patient consent directives, purpose of use of the information, environmental constraints, the identity and roles of the requestor, and various policies for use and re-disclosure of the information by the recipient, which cannot be known or predicted in advance. This variability in general cannot be resolved except in the context of a specific response managed adjudicated under the rules of a security and privacy access control service.

In ACTIVAGE applications of security and privacy labels may require careful policy consideration to account for the distinct possibility that providers may receive incomplete information. Achievement of the proper balance between clinical need and patient privacy is needed to ensure that patient safety is not compromised. Some points of consideration include:

- To address providers' concerns regarding sensitive information, masked (encrypted) or redacted (decrypted) information should be flagged as such in order to increase provider trust in the contents. If flagging notice of redacted content is permitted under applicable policy, then security labels may convey that a clinical fact has been redacted by using codes such as the HL7 Security Observation vocabulary for Data Alteration,
- Unmasked (decrypted) information may be consumed by clinical decision support system such as a drug-drug interaction application, which would alert a non-authorized clinician of potential safety impact.
- Policies must permit overrides masked (encrypted) information in order to access the data only under these conditions, no "back doors" to the data. The intent of such

notifications is always to achieve a workable balance between patient privacy and patient safety,

- Patients may allow a provider to access masked information on an ad hoc basis, e.g., when the provider inquiries about a “mask flag” or a drug-drug interaction alert, using a “shared secret” capability to retrieve the mask’s decryption key, always regulated by a policy.
- Masking offers options not available through redaction. Redacted information is not recoverable.

In ACTIVE while developing the data model to support HL7 HCS, certain assumptions regarding the use and interrelationship of clinical data and security tagging have been made and to be sure that ACTIVAGE AIoTES system is capable to support the privacy and security concerns of:

- Disaggregating health information into clinical data elements, which are the most granular level of clinically relevant information,
- Retrieving clinical attributes about the patient, clinical information category, and provenance such as information source and encoding clinical vocabulary,
- Applying clinical attributes as metadata tags on clinical data elements to generate clinical facts in accordance with clinical rules.
- Clinical facts have no intrinsic security or privacy value. The sensitivity of a clinical fact is determined by matching clinical attributes with the criteria for governance under privacy policies, including patient consent directives.
- Security labels can convey the relative risk associated with disclosure of clinical information based upon standardized security and privacy vocabularies applied to a HCS.

6.2.5 Other Ontologies

Note that there are many other IoT related ontologies that are available many ontologies extensions (for example Schema.org[7]) are planned to also have IoT support. However, we only report few ontologies relevant to us. Further, we also know that some of these ontologies might be standards and widely used we would look forward to seeing the changes. As these ontologies would lack implementation from AHA perspective, we envision no updates will be immediately effective on ACTIVAGE core ontology nor AHA ontology.

6.3 ACTIVAGE Ontology Engineering Process & Methodology

The creation and support of Data and Information Services is a concept where aggregation is done to achieve the objective of sharing information and where usually and as a result of a common interoperability models, the information is modelled and exchanged as an overall objective. Information services composition is to enable the combining of different data services to obtain a more sophisticated but complete service and thus take fully advantage of the data and information services that uses metadata. Since semantic web technologies are employed within ACTIVAGE Project, an intuitive way is to follow the use of Semantic Web. In the following sections it is described the overall process on how to build a formalized ontology coming from a data model and formal vocabularies and a taxonomy according to the particular data and information domain.

[7] https://github.com/schemaorg/schemaorg/issues/1272

6.3.1 AHA Domain Description

Health is a multi-dimensional concept, capturing how people feel and functioning and from many years until now formalisms to define and specify terminology in health have been discuss and used and the currently the most extended use is the Health Level 7 (HL7) that exist as a reference in medical domain. HL7 is one of the best examples on formalising vocabulary/terminology world-wide. As mention health is multi-dimensional and as such involves social, environmental and biomedical factors from early life, and across generations, have long-term impact on health and ageing. Biological ageing is the progressive deterioration of function that occurs in the post-maturity phase, at the individual, physiological systems and cellular levels. It is not necessarily associated with decreased social participation and wellbeing.

In HL7, this is referred to as vocabulary. The HL7 standards define several types of objects that implement various characteristics of vocabulary. Whereas other elements of the HL7 standards are primarily concerned with structure, vocabulary deals with content.

Consistent with the Version 3 philosophy of successively constraining an abstract information model, the least constrained category in vocabulary is a Concept Domain. An HL7 Concept Domain is a named category of like concepts (semantic type) that will be bound to one or more attributes in a static model or properties in datatypes where the data types of those attributes or properties are coded or potentially coded. Concept Domains exist because we want to constrain the intent of a coded element while deferring the association of the element to a specific coded terminology until later in the standards development or implementation process. Thus, Concept Domains are independent of any specific vocabulary or code system.

The categorization of Concept Domains is hierarchical allowing for further constraint on the breadth of the semantic category covered by the Concept Domain. Such constrained domains are known as “sub-domains”. Sub-domains allow for further specialization (constraint) on the intended values for a domain.

6.3.2 Vocabulary Abstraction and Taxonomy

In the context of ACTIVAGE project, it is important to define the correct vocabularies that will apply to the project, in this respect we make reference to the most relevant organization in health to understand first the concepts involved in domain of the project and second define the terminology that is meant to be used and thus correctly use vocabularies from related domains:

Health: The World Health Organization (WHO) defines health in its broader sense in its 1948 constitution as “a state of complete physical, mental, and social well-being and not merely the absence of disease or infirmity.”

Healthcare: Health care or healthcare is the maintenance or improvement of health via the diagnosis, treatment, and prevention of disease, illness, injury, and other physical and mental impairments in Human beings. Health care is delivered by health professionals, and

Well-being, wellbeing, welfare or wellness: is a general term for the condition(s) of an individual or group, to achieve a level of health according to particular conditions/situations, for example their social, economic, psychological, spiritual or medical state

The above concepts are important in order to differentiate what vocabularies are necessary to formalize in ACTIVAGE project, as per the terminology in healthcare is not well defined there is not established regulation or formats that regulates the terminology. In this process health analysis and technology there have been classified two big areas as indoor activities and outdoor activities. In health there is no difference if activities are done inside or outside,

however when it comes to evaluate the level of influence that the technology has it is important to differentiate those two environments.

Over the last decade a number of technologies have been developed that support individuals in keeping themselves active. This can be done via e-coaching mechanisms and by installing more advanced technologies in their homes. The objective of the ACTIVAGE project and the Active and Healthy Ageing (AHA) approach is to integrate existing tools, hardware, and software that assist individuals in improving and/or maintaining a healthy lifestyle. This approach defines a way to identify vocabularies by integrating several hardware/software components that generate various types of data. Some examples include heart-rate data, coaching information, in-home activity patterns, mobility patterns, and so on. Various subsystems in the ACTIVAGE platform can share their data in a semantic and interoperable way, through the use of an AHA data-store and a wearable devices ontology.

The ACTIVAGE Project is designed to support standard vocabularies that are common to AHA applications and Internet of Things in the form of a one-stop solution to browse, select and search data services. The selection of the vocabularies are part of the platform and its different deployment sites where on each place the data is produced and where the different technology resides.

6.3.3 Data Models Representation

6.3.3.1 GoIoTP Class Diagram

The SIL data model makes currently use of the semantic model of the GOIoTP [12] (Generic Ontology for IoT Platforms)[8] ontology and its extension, GOIoTPex [12], which provides useful concrete entities. This ontology consists of the following modules:

- **Device:** hardware or virtual sensor connected to IoT platforms
- **Platform:** IoT platforms
- **Observation and actuation**: information collected by sensors, or the intention to act upon something
- **Units & measurements:** systems of units and values
- **User:** human or software that uses or is a client of IoT entities
- **Geolocation:** position within the world, defined by coordinates
- **Service:** web services
- **Provenance:** origin, ownership and history of IoT entities

Next, the main classes and relations are described. For clarity, only the most important elements of each module are represented in the diagrams. GOIoTP defines a couple of properties that are not specific to any module, namely, iiot:hasName and iiot:hasDescription, which are meant to contain human-readable text to annotate any GOIoTP entity. The prefixes used in the description of the classes and properties are detailed in Table 10.

Table 10: GOIoTP ontology prefixes

Name	Prefix	URI
GOIoTP	iiot	http://inter-iot.eu/GOIoTP#
GOIoTPex	iiotex	http://inter-iot.eu/GOIoTPex#

[8] https://docs.inter-iot.eu/ontology/

SSN	ssn	http://www.w3.org/ns/ssn/
SOSA	sosa	http://www.w3.org/ns/sosa/
SSNX	oldssn	http://purl.oclc.org/NET/ssnx/ssn#
Open Geospatial URI base	ogc	http://www.opengis.net/
OGC simple feature geometries	sf	http://www.opengis.net/ont/sf#
OGC geometries	gml	http://www.opengis.net/ont/gml#
GeoSPARQL	geosparql	http://www.opengis.net/ont/geosparql#
GeoSPARQL functions	geosparqlf	http://www.opengis.net/def/function/geosparql/
GeoSPARQL rules	geosparqlr	http://www.opengis.net/def/rule/geosparql/
NASA SWEET units	sweet_units	http://sweet.jpl.nasa.gov/2.3/reprSciUnits.owl#
NASA SWEET mathematical operations	sweet_oper	http://sweet.jpl.nasa.gov/2.3/reprMathOperation.owl#
NASA SWEET representation	sweet_repr	http://sweet.jpl.nasa.gov/2.3/repr.owl#
NASA SWEET mathematical relations	sweet_mrela	http://sweet.jpl.nasa.gov/2.3/relaMath.owl#
NASA SWEET scientific relations	sweet_screla	http://sweet.jpl.nasa.gov/2.3/relaSci.owl#
W3C time	time	http://www.w3.org/2006/time#
Vocabluary annotation	vann	http://purl.org/vocab/vann/
Vocabluary of a Friend	voaf	http://purl.org/vocommons/voaf#
Friend of a friend	foaf	http://xmlns.com/foaf/0.1/
Dublin Core elements	dc	http://purl.org/dc/elements/1.1/
Dublin Core	dcterms	http://purl.org/dc/terms/
Semantic Web Status	vs	http://www.w3.org/2003/06/sw-vocab-status/ns#
OWL	owl	http://www.w3.org/2002/07/owl#
RDF	rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
RDF schema	rdfs	http://www.w3.org/2000/01/rdf-schema#
XML	xml	http://www.w3.org/XML/1998/namespace
XML datatypes	xsd	http://www.w3.org/2001/XMLSchema#

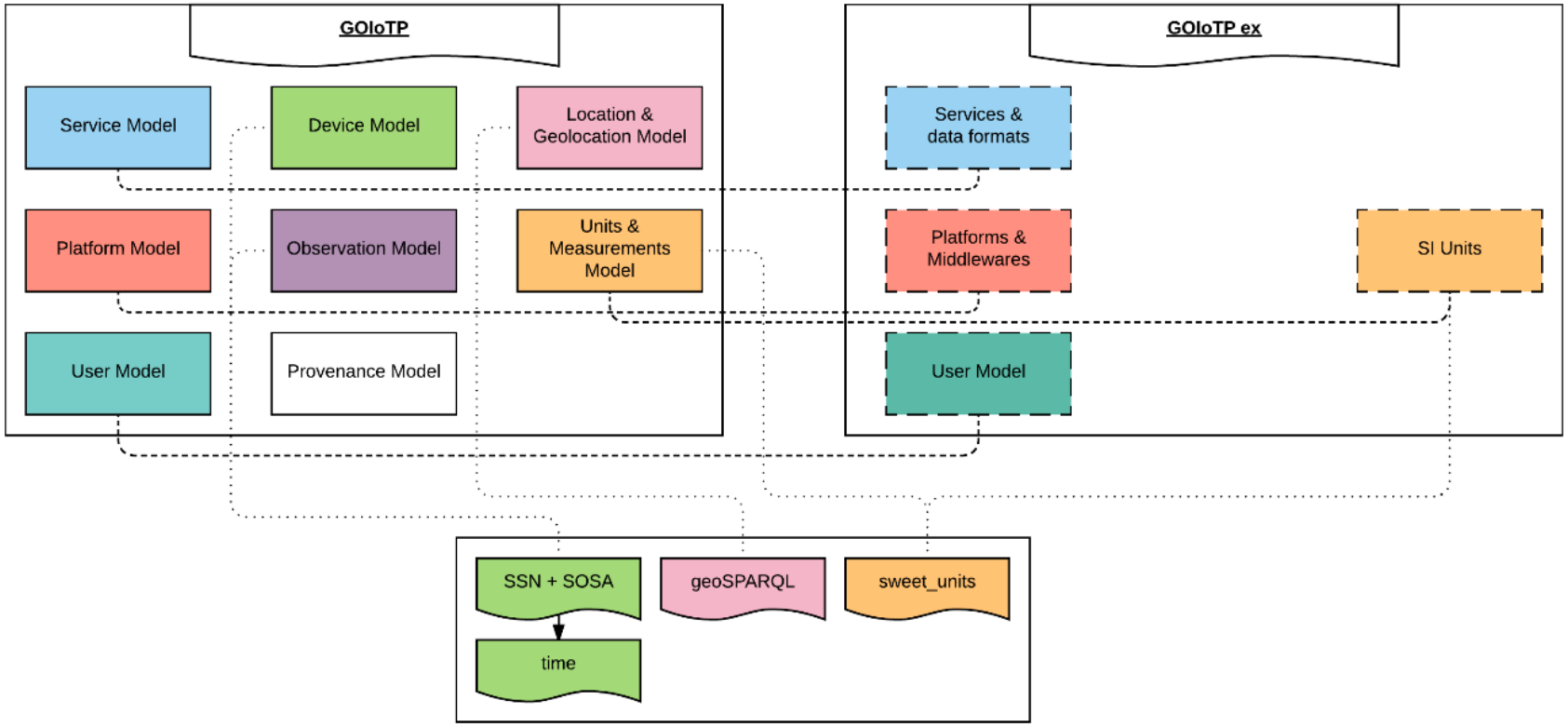

Figure 15: GOIoTP modules

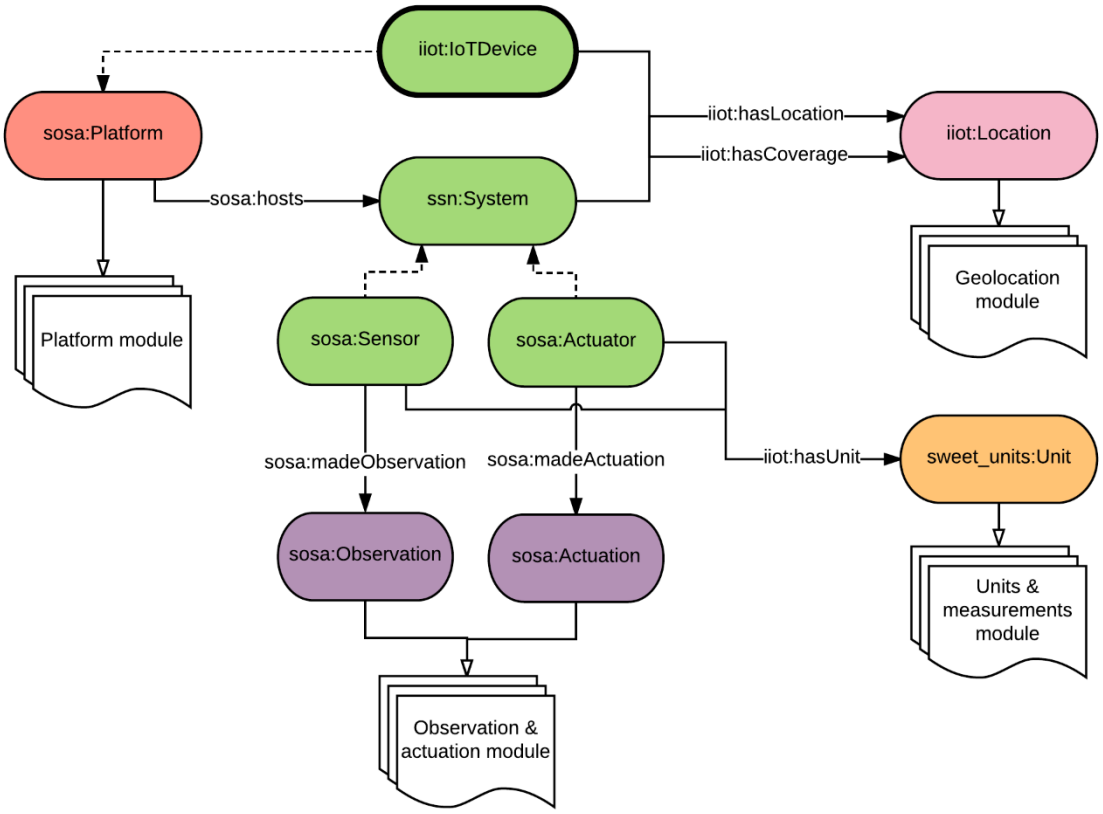

Figure 16: GOIoTP device module

The device module (Figure 16) represents the IoT devices. This module has four main classes: iiot:IoTDevice, sosa:Sensor, sosa:Actuator and sosa:Sampler. The iiot:IoTDevice class represents any kind of physical or virtual smart device. As a subclass of sosa:Platform, it hosts entities of type sosa:System. The classes sosa:Sensor and sosa:Actuator are subclasses of sosa:System and can be hosted on a iiot:IoTDevice or any other sosa:Platform. Sensors and actuators usually have a single purpose, such as the detection of changes in a single observable property (e.g. air temperature). Similarly, iiot:IoTDevice usually represents a sosa:Sensor (or sosa:Actuator), or hosts at least one instance of one of these classes.

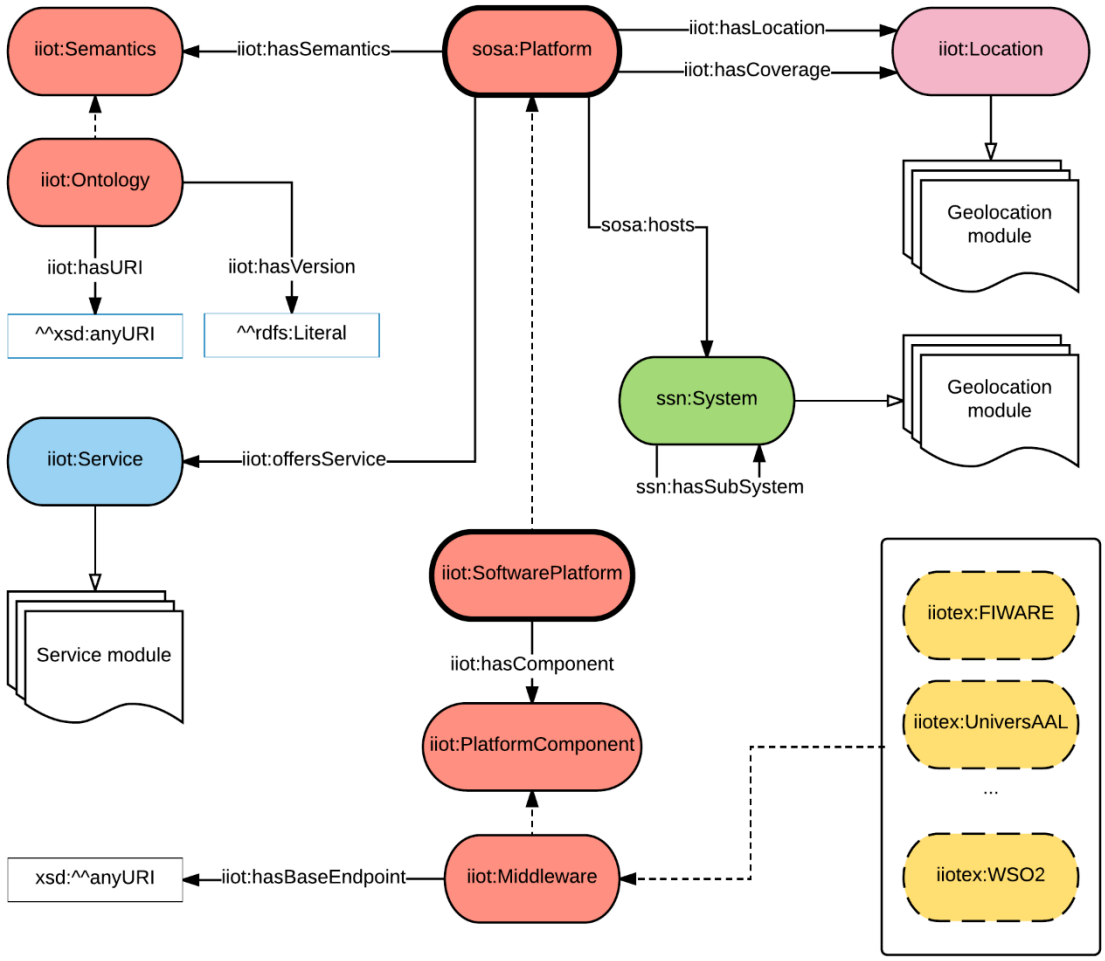

Figure 17: GOIoTP platform

The platform module (Figure 17) refers to the IoT platforms. Instances of the class sosa:Platform can have explicitly defined semantics, which are asserted via the iiot:hasSemantics. This property is useful for systems that interoperate elements with different semantics. A iiot:SoftwarePlatform has components that can be instances of iiot:PlatformComponent or its subclass iiot:Middleware. GOIoTPex, which extends GOIoTP, defines a collection of subclasses of iiot:Middleware that represent existing IoT Platforms, such as universAAL or Fiware.

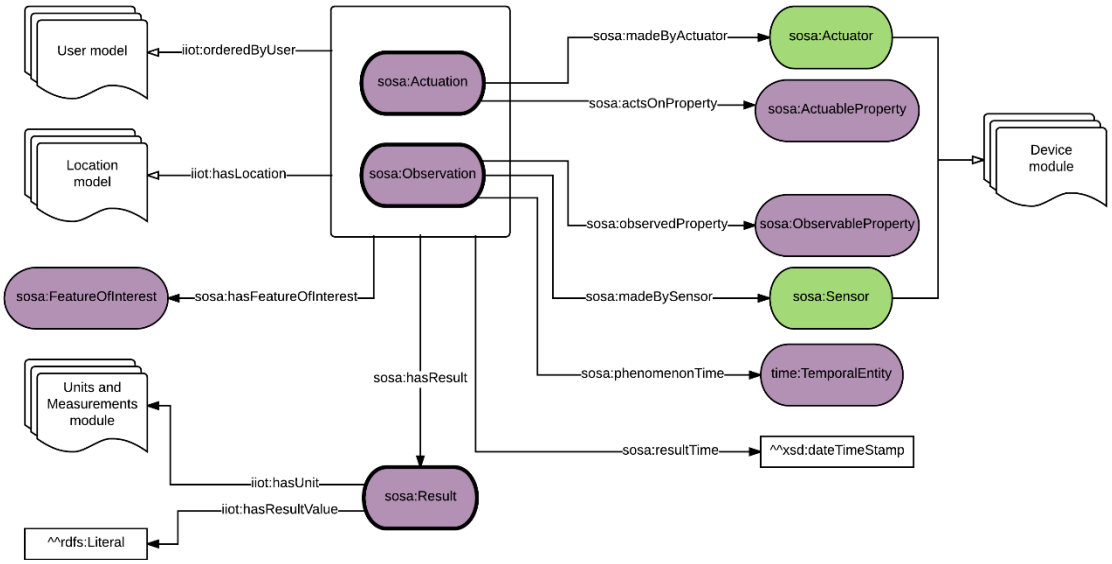

Figure 18: GOIoTP observation and actuation module

The observation and actuation module (Figure 18) defines data structures to represent the information obtained from a sensor or produced by an actuator. The class sosa:Result represents the most relevant information to sensing and actuation. The iiot:hasResultValue property of this class contains a literal value of the observation or actuation. The observation and actuation module relates to the device module via the sosa:madeActuation and sosa:madeObservation for sosa:Actuators and sosa:Sensors respectively. The classes sosa:Actuation and sosa:Observation relate to the class sosa:Result through the property sosa:hasResult. Observations and actuations may have more than one result. For instance, a blood pressure measurement of a patient would be modelled as a single observation with two results, representing the systolic and diastolic blood pressure values.

The units and measurements module (Figure 19) allows attaching information about unit of measurement to instances of sosa:Result making use of the class sweet_units:Unit. On the other hand, the class iiot:MeasurementKind is a subclass of ssn:Property and its instances are intended to store information about the type of measurement (e.g. mass, length, etc). Any system of units can be used under sweet_units:Unit and iiot:MeasurementKind. GOIoTPex provides a full suite of units and measurement types from the SI system. Subclasses of iiot:MeasurementKind are defined in order to represent these types of measurements, such as iiotex:Distance, iiotex:Luminance and many others. Instances that represent concrete SI units, such as sweet_units:kilogram, are also included in GOIoTPex under the taxonomy defined in SWEET Units ontology.

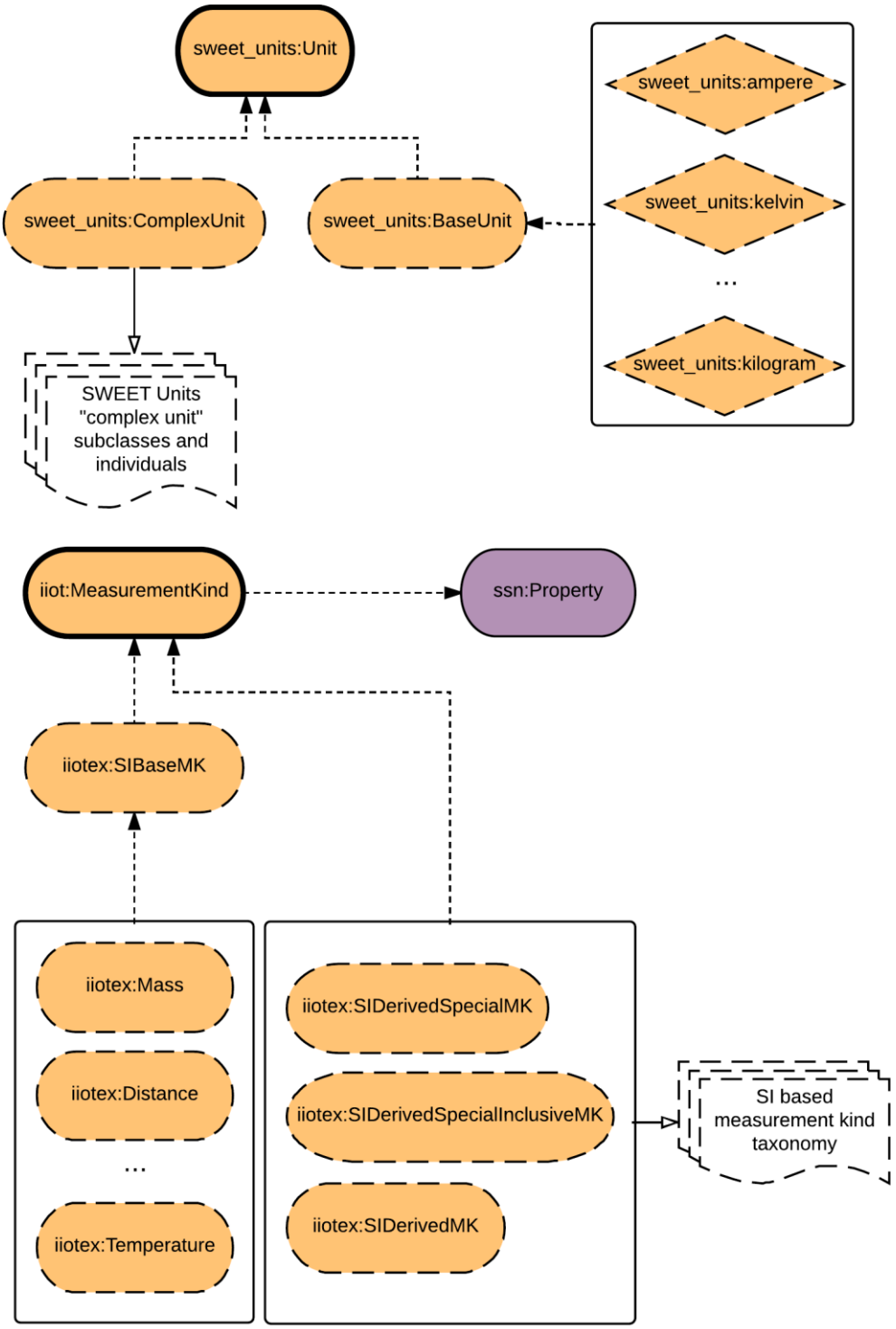

Figure 19: GOIoTP units and measurements module

The main class in the user module (Figure 20) is iiot:User, which represents any human or software that uses or is a client of an IoT system. The user module connects to the device, observation, services and platform modules via iiot:hasUser and iiot:orderedByUser properties. The iiot:hasUser represents a generic connection between a user and an entity. This property indicates that the functionalities of a device are being accessed by a particular user. On the other hand, the iiot:orderedByUser property indicates that an observation or

actuation was ordered to be done by a user. For example, it can inform that a particular doctor ordered a measurement of a patient's blood pressure, or that a software agent decided that an actuation should be performed by an actuator.

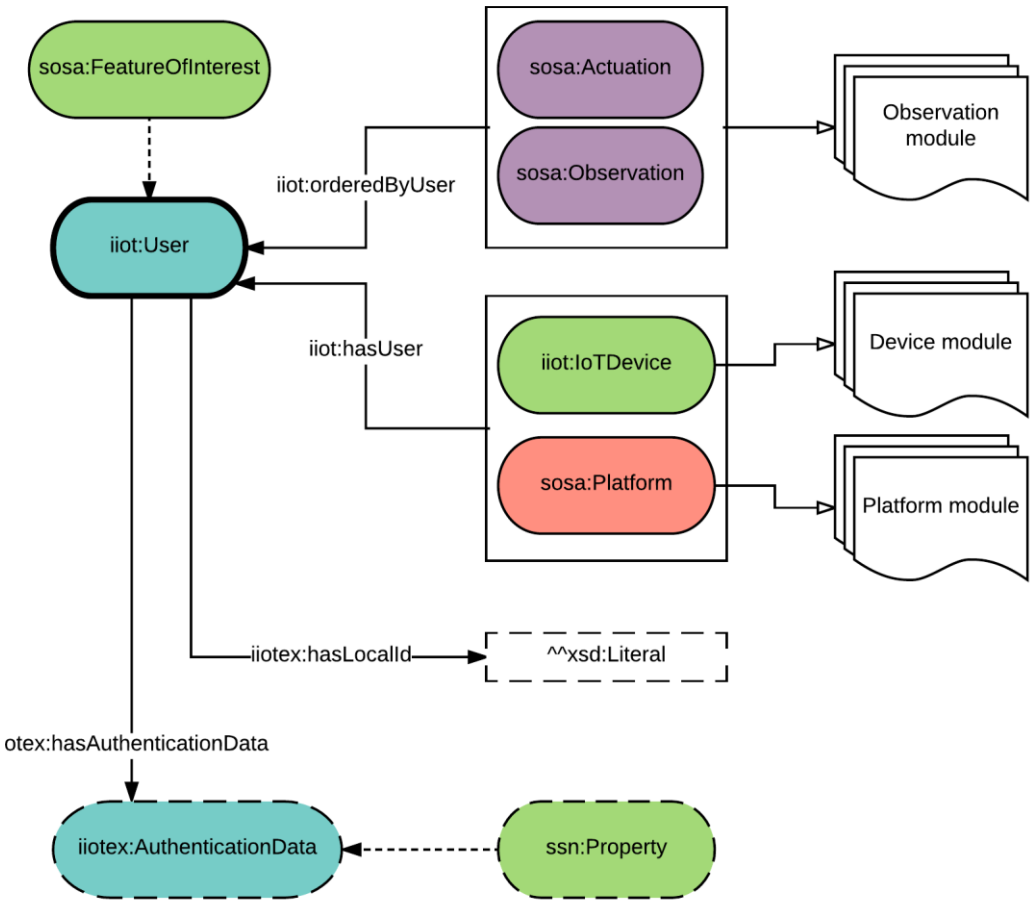

Figure 20: GOIoTP user module

The main class of the geolocation module (Figure 21) is iiot:Location, which describes any physical location, such as a building, a specific room, or a city. The geographical coordinates are attached to a iiot:Location with the geosparql:asWKT property, whose value is a geosparql:wtkLiteral, where WKT stands for "well known text". WKT is a string format that allows declaration of geographical points (e.g. "POINT[31.2543139, -24.2584805]"), lines and areas bounded by many different shapes. WKT also allows the use of optional textual descriptions, like "AREA["Netherlands offshore."]". Hence, the iiot:hasLocation property value may use any allowed geosparql:asWKT value in order to inform about an approximate location (e.g. an entity in a building). The iiot:hasLocation and iiot:hasCoverage properties, which can be applied to sosa:Result, ssn:System and sosa:Platform, connect the geolocation module with the device, observation and platform modules. The first property informs about a physical location of an entity, while the latter defines a point, or an area where the functionalities of an entity are available. For instance, a temperature sensor that covers a whole room (iiot:hasCoverage) can be placed in one specific corner of that room (iiot:hasLocation).

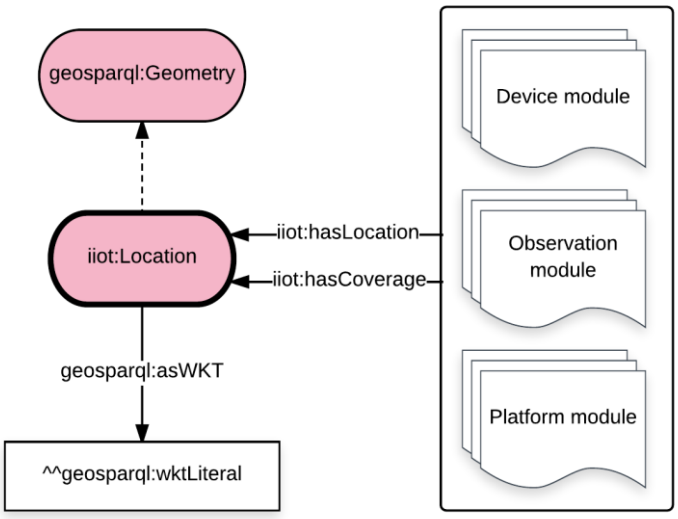

Figure 21: GOIoTP geolocation module

6.3.3.2 ACTIVAGE AHA Core Ontology

The current version of the ACTIVAGE AHA Core Ontology is a mapping/merge of existing ontologies from multiple domains into a single one called AHA Core. As it can be seen in the Figure 22, it fosters concepts from a number of "third-party" vocabularies and ontologies such as GOIoTP, OpenIoT, FIESTA-IoT, BIG-IOT, WGS84[9], W3C SSN/SOSA, IoT-lite, M3-lite Taxonomy, DUL, Time[10]. Below, HL7 and we present updates that we have performed:

The observations taken by the sensing/actuators devices are linked to: (a), the corresponding `GIoTP:IoTproperty` via `ssn:observedProperty`, (b) `ssn:observationSamplingTime` that further links to `time:Instant` to represent when the observation was taken (i.e. timestamp), (c) `geo:Point`, (d) `ssn:ObservationValue` via `ssn:SensorOutput` (`ssn:ObservationValue` is linked to the sensed value of the IoTproperty via data property `dul:hasDataValue`, and `m3-lite:Unit` concept), and (e) `m3-lite:hasMeasurementType` to know whether the measurement was `m3-lite:Manual`, `m3-lite:Automatic` or generated from an `m3-lite:Experiment`. One important change that has been made to the ontology is the replacement of `dul:TimeInterval` to `time:Instant`. `dul:TimeInterval` concept relates to the interval in which the observation was taken.

In ACTIVAGE AHA Core ontology we consider all the observations to be instantaneous. In case a resource provides an average observation value during the observation period we still link the observation value to a `time:Instant`. This `time:instant` can be any time during the observation period. The `time:Instant` is then linked to `xsd:dateTime` using `time:inXSDDateTime` data property. Previously, only `xsd:Double` datatype values were supported for `ssn:ObservationValue`. However, the observations could be of Boolean type depending on the type of sensors (for example, proximity sensor would result in a 'true' or 'false' value). We add the `xsd:boolean` to the range of `dul:hasDataValue`.

We add oneM2M classes as equivalent classes and properties as equivalent properties to facilitate oneM2M compliant Deployment Sites technology join AIoTES Architecture.

[9] WGS84 is actually a basic RDF vocabulary that provides the Semantic Web Community with a namespace for the representation of latitude, longitude and other information about spatially-located things. For reference please see https://www.w3.org/2003/01/geo/

[10] https://www.w3.org/TR/owl-time/

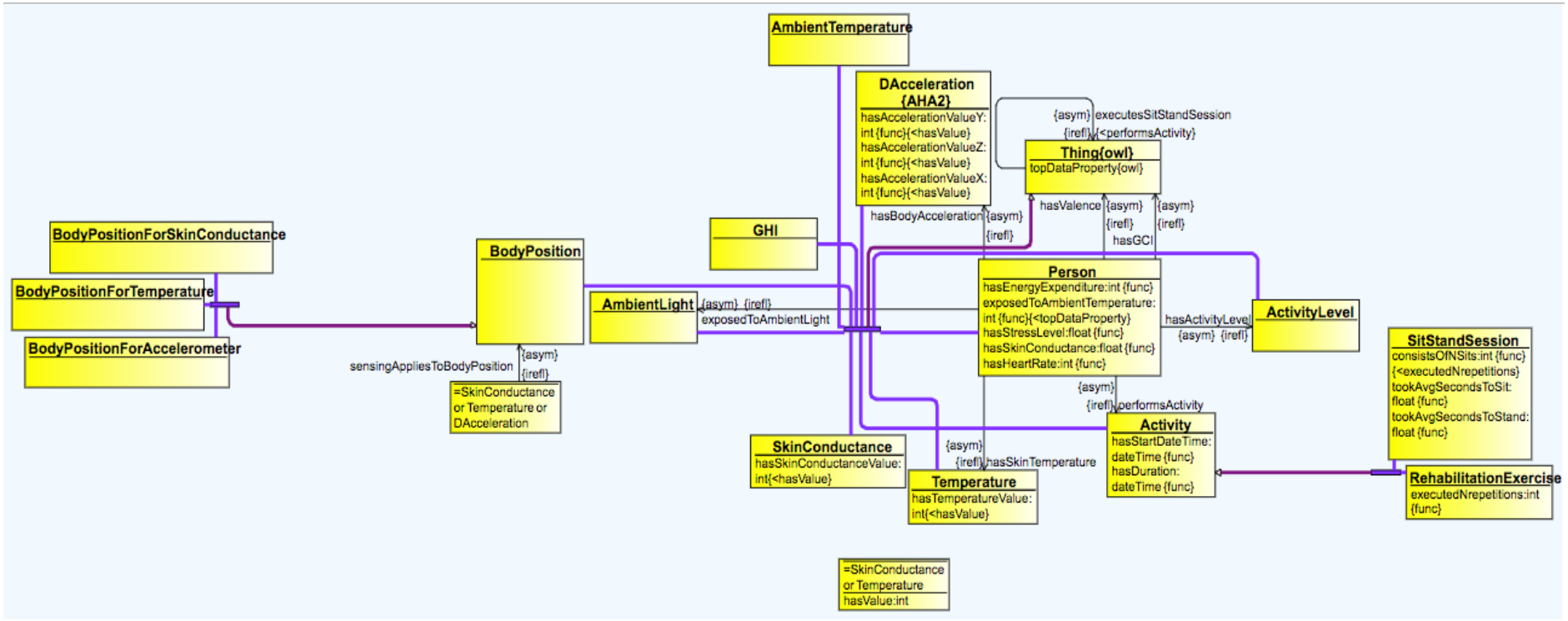

Figure 22: ACTIVAGE AHA Core Ontology Class Diagram

6.3.4 Ontology Alignment and Mappings

The Smart Appliances REFerence[11],[12] (SAREF) [22] ontology is designed for household and home appliances in residential buildings, especially for the purpose of energy management. SAREF aims to align existing ontologies in the domain of smart appliances. Many standards have been proposed to enable the interoperation of appliances from diverse vendors. However, the number of standards is so high that overlapping is inevitable. To address this problem, the European Commission launched a study for the purpose of proposing a reference ontology gathering the efforts of existing appliances standards relevant for energy efficiency. The final result of this study is the SAREF reference ontology that is intended to be transferred to European Telecommunications Standards Institute (ETSI) Smart Machine to Machine (SmartM2M) that could contribute it to International Machine-to-Machine Standardization (oneM2M) initiative.

6.3.5 Ontologies Aligned

OneM2M and SAREF are well known ontologies. IN ACTIVAGE it is consider the fact that many deployment sites using IoT technology would be using such ontologies to store their data. In order to help the deployment sites in adopting ACTIVAGE data model (AHA Core ontology and the associated Ontologies) we provide a mapping between oneM2M ontology and ACTIVAGE AHA-Core ontology. We add such mappings within AHA Core ontology using either `owl:equivalentClass` or `owl:equivalentProperties`. Tables 11 and 12 show concepts from oneM2M ontology that are mapped in ACTIVAGE AHA Core ontology.

Table 11: Classes mapping between oneM2M Base ontology and ACTIVAGE AHA Core ontology

Class in oneM2M Ontology	Mapping relationship	Class in ACTIVAGE AHA Core
oneM2M:Thing	owl:equivalentClass	ssn:Sensor
oneM2M:ThingProperty	owl:equivalentClass	m3-lite:QuantityKind
oneM2M:Device	owl:equivalentClass	m3-lite:SensingDevice
oneM2M:Service	owl:equivalentClass	ssn:Observation
oneM2M:OperationOutput	owl:equivalentClass	ssn:ObservationValue
oneM2M:MetaData	owl:equivalentClass	m3-lite:Unit

Table 12: Object properties mapping between oneM2M Base ontology and ACTIVAGE AHA Core ontology

Object property in oneM2M Ontology	Mapping relationship	Object property in ACTIVAGE AHA Core
oneM2M:hasThingProperty	owl:equivalentProperty	iot-lite:hasQuantityKind
oneM2M:hasService	owl:equivalentProperty	ssn:madeObservation
oneM2M:hasMetaData	owl:equivalentProperty	iot-lite:hasUnit

[11] https://sites.google.com/site/smartappliancesproject/ontologies/oma-lightweight_m2m-ontology

[12] http://ontology.tno.nl/saref/

6.3.5.1 Security and Privacy Class diagrams

ACTIVAGE follows the same approach for Privacy and Security Classification System (HCS) that HL7 Healthcare, since it is an international standard document describing the use of a Healthcare Privacy and Security Classification System (HCS) and it is suitable for protecting health care information by access control systems to enforce privacy and security policies. The full specification about the Privacy and Security can be found at: http://www.hl7.org/implement/standards/product_brief.cfm?product_id=345

6.3.5.1 GOIoTP Class Diagram

As described in previous section, INTER-IoT approach developed a generic ontology of IoT Platforms (GOIoTP). The GOIoTP is used as the centrepiece for establishing platform interoperability (allowing for, among others, data interoperability, message translation, etc.).

The Figure 24 present the GOIoTP Ontology Class Diagram representation. For details, please visit INTER-IoT project Data Model at http://docs.inter-iot.eu/ontology and http://www.inter-iot-project.eu .

6.3.5.1 GOIoTP Extensions Class Diagram

The GOIoTP is the key method for data semantics and interoperability implemented in INTER-IoT and it follows a set of tools, to support development of platform semantic interoperability layer. The GOIoTP has been extended to support interoperability use cases that also apply to ACTIVAGE. Details about these extensions are in http://docs.inter-iot.eu/ontology

6.3.5.1 Big IoT Class Diagram

The BIG IoT Data Services are implemented uses the BIG IoT schemas as reference to an ontology approach for a Data Marketplace. BIG-IoT classify a provider for the offerings concept that is used in a BIG IoT Marketplace (as a Consumer) for the data. More details can be found at: http://big-iot.eu

6.3.5.1 OpenIoT Class Diagram

OpenIoT ontology introduces new concepts that haven't been covered by the existing vocabularies and may be of interest for the ACTIVAGE Project. These concepts include the notion of Virtual sensors and Utility metrics as represented below using existing standard vocabularies more details at: https://github.com/OpenIotOrg

6.3.5.1 FIESTA-IoT Class Diagram

The current version of the FIESTA-IoT Ontology (see figure below) is a merge of existing IoT ontologies into a single one. As it can be seen in the Figure 28, it fosters concepts from a anumber of "third-party" ontologies such as WGS84[13], W3C SSN, IoT-lite, M3-lite Taxonomy, DUL, Time[14] and QU. Below, we present updates that we have performed.

[13] WGS84 is actually a basic RDF vocabulary that provides the Semantic Web Community with a namespace for the representation of latitude, longitude and other information about spatially-located things. For reference please see https://www.w3.org/2003/01/geo/

[14] https://www.w3.org/TR/owl-time/

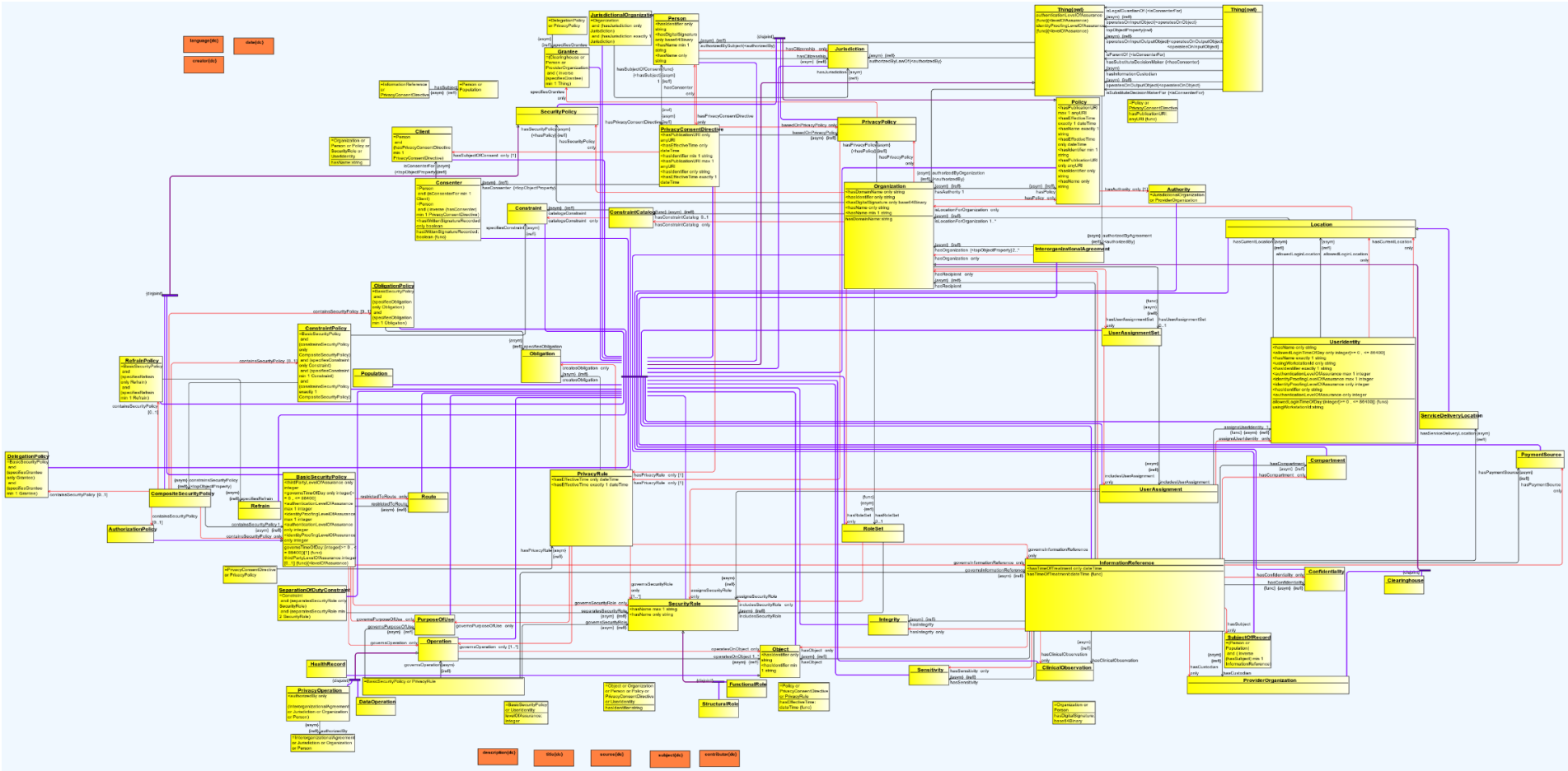

Figure 23: Security & Privacy Ontology Class Diagram[15]

[15] Downloadable high-resolution image at http://srvgal106.deri.ie:8013/ontology/SecurityAndPrivacyDiagram.svg

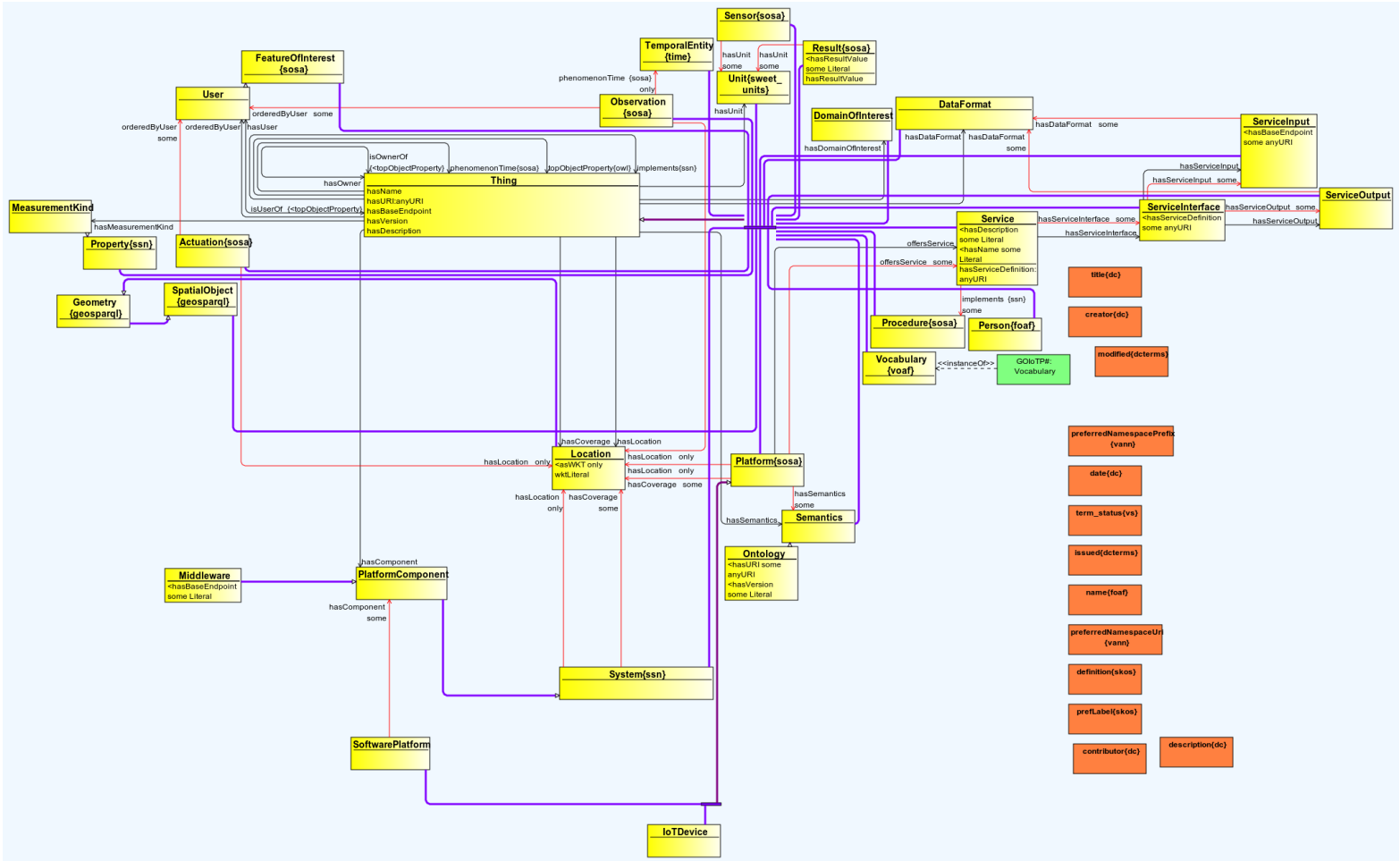

Figure 24: GOIoTP-IoT Ontology Class Diagram[16]

[16]Detailed broken-down diagram at http://docs.inter-iot.eu/ontology and online high-resolution image at http://srvgal106.deri.ie:8013/ontology/GOIoTPDiagram.svg

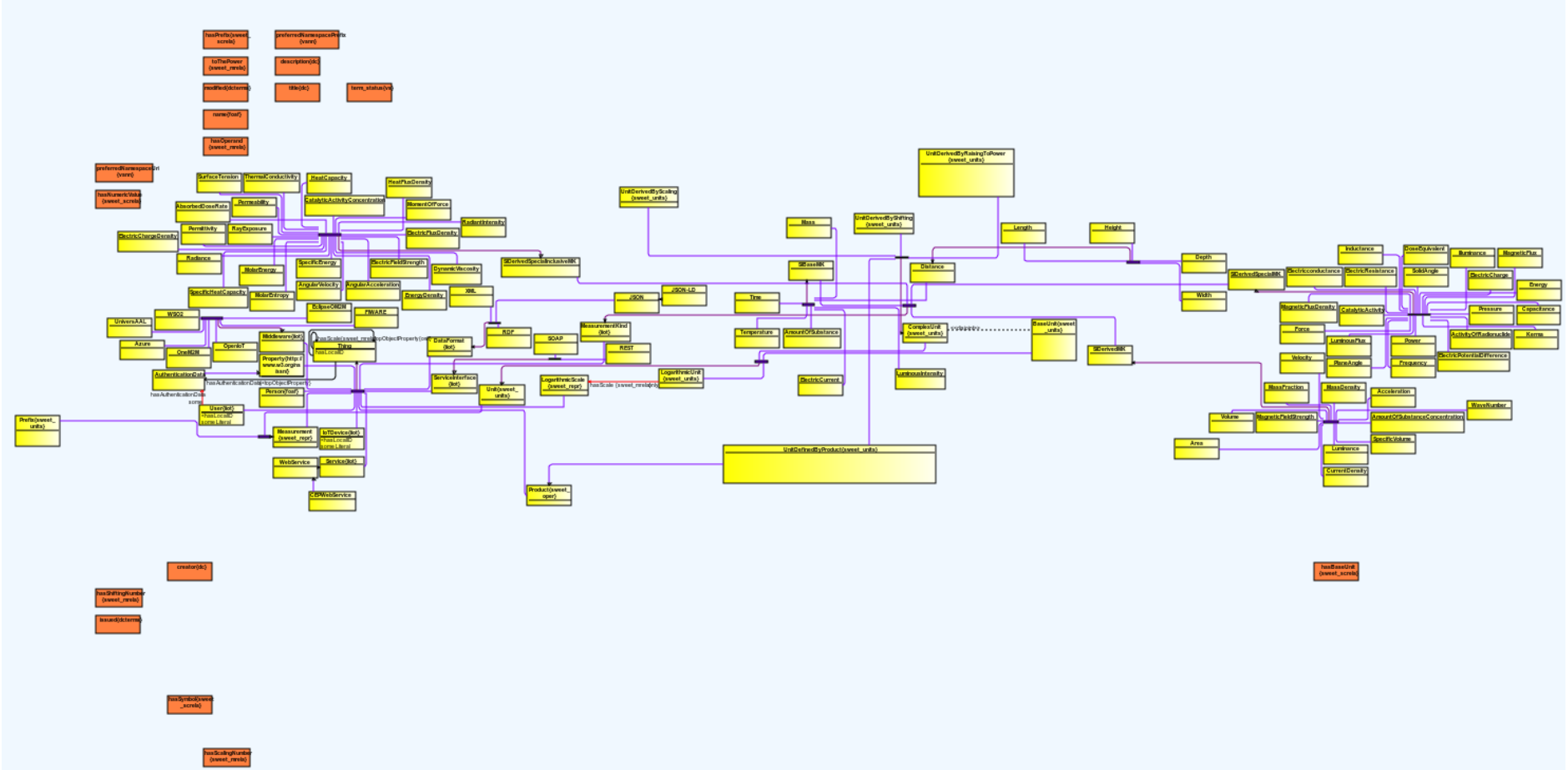

Figure 25: GOIoTPex Ontology Class Diagram[17]

[17] Detailed broken-down diagram at http://docs.inter-iot.eu/ontology and online high-resolution image at http://srvgal106.deri.ie:8013/ontology/GOIoTPexDiagram.svg

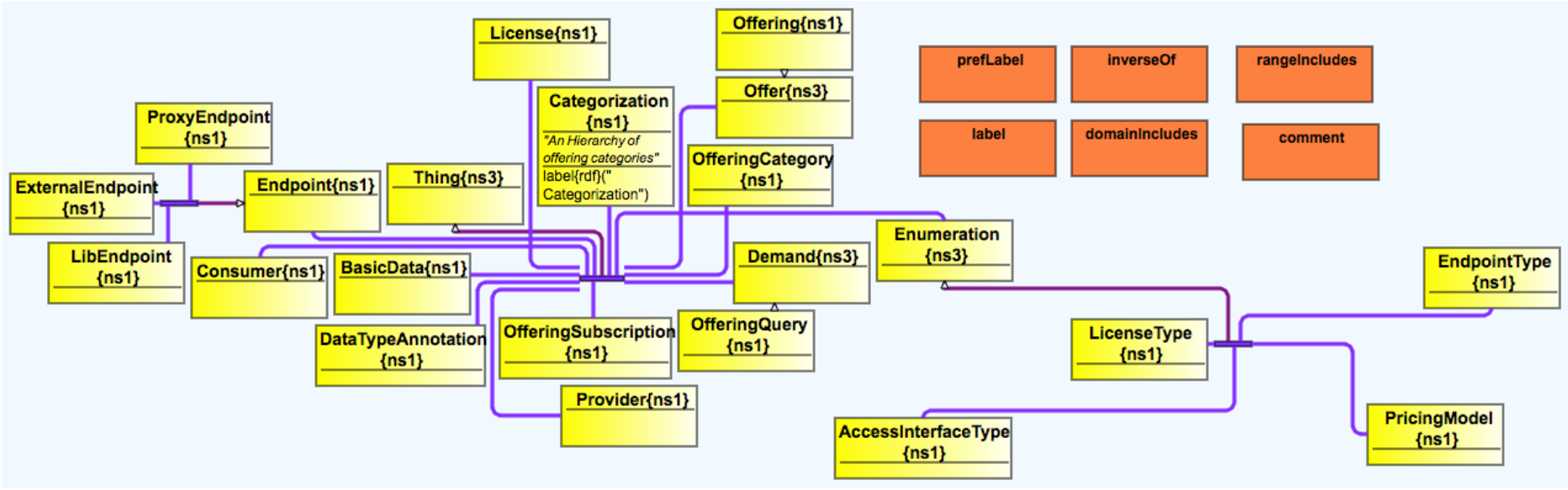

Figure 26: BIG IoT Ontology Class Diagram

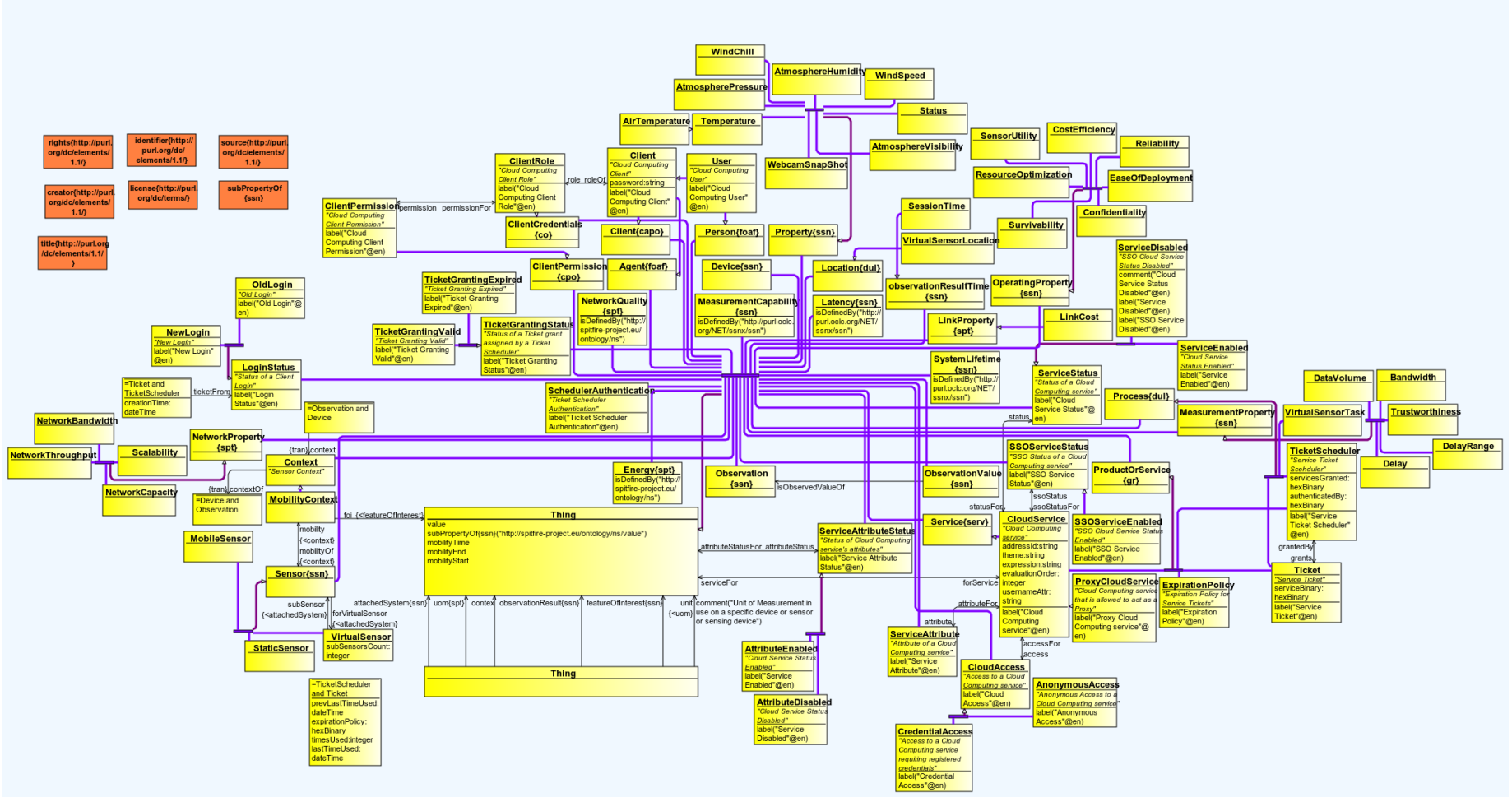

Figure 27: OpenIoT Ontology Class Diagram[18]

[18] Online high-resolution image at http://srvgal106.deri.ie:8013/ontology/OpenIoTDiagram.svg

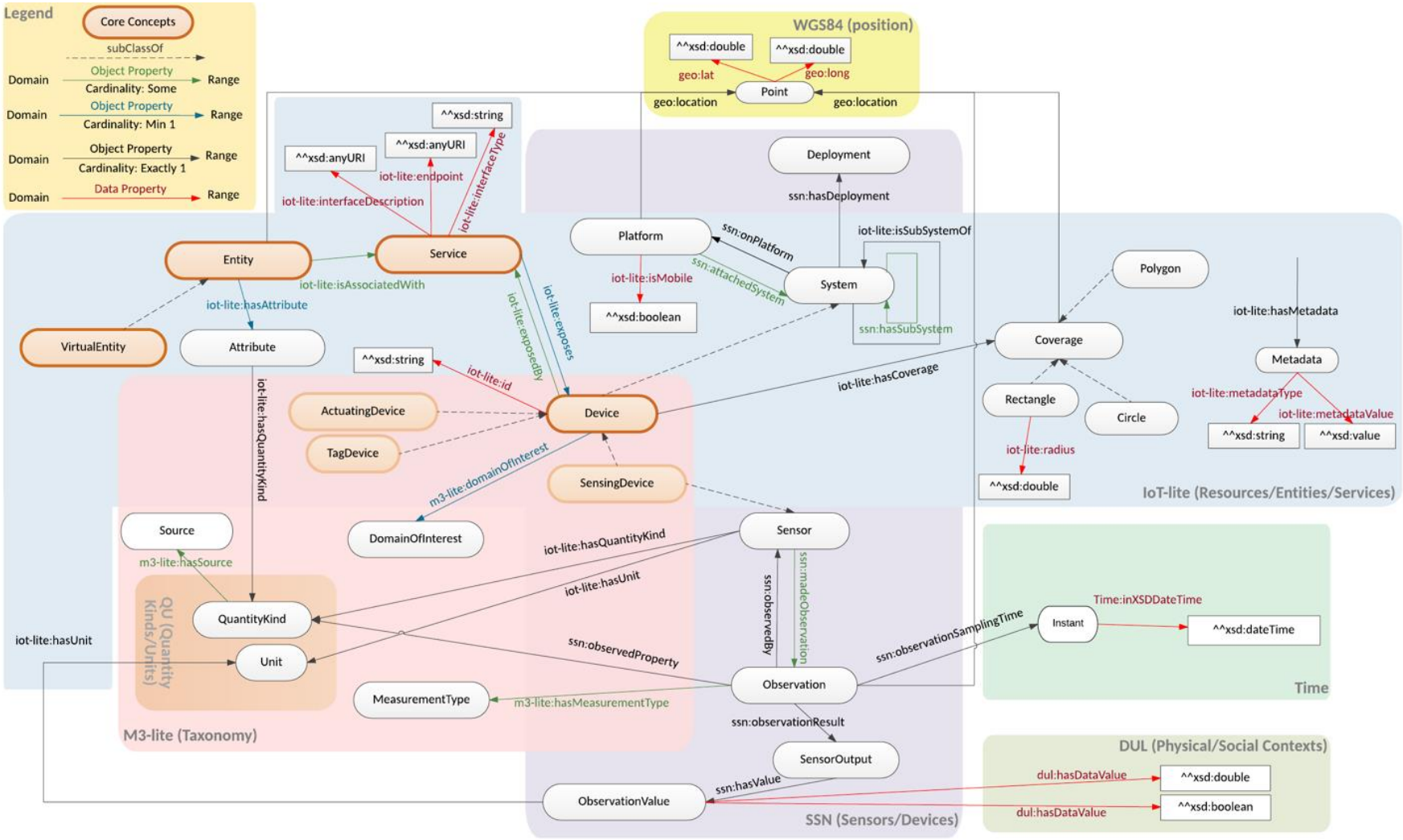

Figure 28: FIESTA-IoT Ontology Class Diagram

6.4 AIoTES Data Pack

6.4.1 Data Model Conventions

The data model at the deployment site is part of the local ecosystem containing the terminology, the logic, the operation of the local technology, all this information characterizes the ecosystem functionality and is sometimes also the identity. ACTIVAGE acts at the deployment site to support and predefine best practices for ecosystem interoperability and enlargement, thus one of the main aspects to address is the way information is organised and structured. In this section conventions for the deployment sites are proposed in order to have characteristics that are usually reflected in the form of closed market to move it towards a global market exploitation.

6.4.1.1 Deployment Sites

The following is the convention for describing deployment sites local data models, where to identify easily the geographical location and if it is a city or a region or a department in a country, it is agreed to follow this convention that not only serve as naming convention but also to have a way, machine can identify the information. The three first characters are dedicated for the three first letters of the city's name, starting with capital letter i.e. Xxx. When there is no a city but a region or department the three characters must be capitals. i.e. XXX. The use of a dash is to indicate the following two letters are the country and they follow the European standard in terms of country naming/specification.

XxxYY-DM.otr – City, Country

XXXYY-DM.otr – Region or Department, Country

otr = csv, txt, mat, tsv, pcap, bam, see Annex 4 for different data types

Table 13: Deployment Sites Naming Conventions in ACTIVAGE

DS Data Model	City - Geographical Location	Country
GalES-DM.otr	Galicia	Spain
ValES-DM.otr	Valencia	Spain
MadES-DM.otr	Madrid	Spain
RERIT-DM.otr	Region Emilia Romanax	Italy
ARMHE-DM.otr	Attiki Region Metamorphosis	Greece
ISEFR-DM.otr	Iseré	France
DarDE-DM.otr	Darmstadt	Germany
LeeUK-DM.otr	Leeds	UK
HelFI-DM.otr	Helsinki	Finland

6.4.1.2 AHA Domain Specific

The following is the convention for describing domain specific data models, the naming convention is limited to four letter that represent the acronym of the AHA deployment site use case. When there are cases that the naming is four or more words the most representative ones are selected. Note that adverts and prefix are not used for naming as follow:

<XXXX.xs> (XXXX-LD.jsonld) [XXXX.owl] - 4 or more words in the naming

<XXXx.xs> (XXXX-LD.jsonld) [XXXX.owl] - 3 words in the naming

<XxXx.xs> (XxXx-LD.jsonld) [XxXx.owl] - 2 words in the naming

<Xxxx.xs> (Xxxx-LD.jsonld) [Xxxx.owl] - 1 word in the naming

Table 14: Activity Naming Conventions in ACTIVAGE

DS Data Model	XML Format	JSON-LD Format	OWL
Daily Activity Monitoring at Home (Home Daily Activity Monitoring)	<HDAM.xs>	(HDAM-LD.jsonld)	[HDAM.owl]
Integrated Care for Chronic Conditions (Chronic Conditions Integrated Care)	<CCIC.xs>	(CCIC-LD.jsonld)	[CCIC.owl]
Monitoring Outside Home (Outside Home Monitoring)	<OHMg.xs>	(OHMg-LD.jsonld)	[OHMg.owl]
Emergency Trigger	<EmTr.xs>	(EmTr-LD.jsonld)	[EmTr.owl]
Exercise Promotion	<ExPr.xs>	(ExPr-LD.jsonld)	[ExPr.owl]
Cognitive Stimulation	<CoSt.xs>	(CoSt-LD.jsonld)	[CoSt.owl]
Prevention of Social Isolation (Social Isolation Prevention)	<SIPr.xs >	(SIPr-LD.jsonld)	[SIPr.owl]
Safety, Comfort and Security at Home	<HSCS.xs>	(HSCS-LD.jsonld)	[HSCS.owl]
Support for Transportation and Mobility	<TMSt.xs>	(TMSt-LD.jsonld)	[TMSt.owl]

6.4.2 Online Data Files and Schemas

The ACTIVAGE Data Pack is a set of metadata model diagrams that represent the way the data is structured, it also contains the metadata in two different formats, json-ld and owl. The two formats are the convergence of technology in ACTIVAGE, while some deployment sites uses json-ld other uses RDF which at the same time both are compatible.

The different data models are complementary each other and in some cases redundant, this is the reason information needs to be structured using metadata, reducing the problems of compatibility and increasing the changes of compatibility and re-use of information as described in the data model.

The Figure 29 shows the files that are included in the ACTIVAGE data model pack, there is also the ATIVAGE Core data model as has been explained above nevertheless is not shown in this image it is included and makes part of the Data Pack.

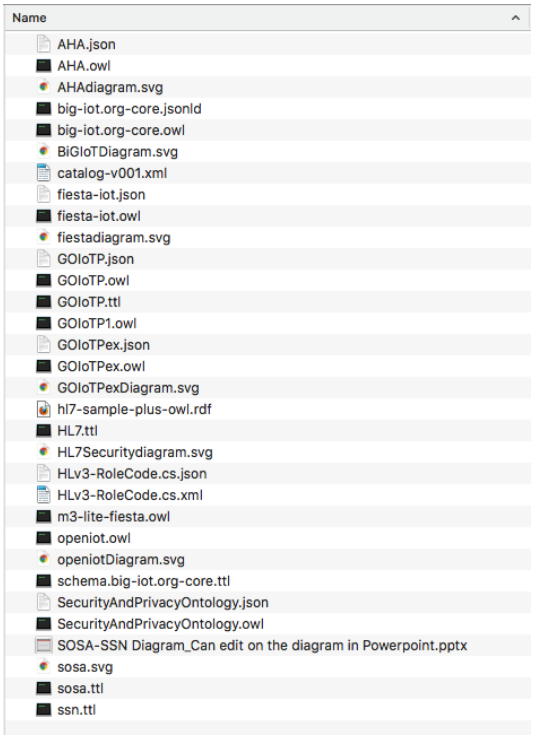

Figure 29: ACTIVAGE Data Model Pack

6.5 Tools & Best Practices in AIoTES

ACTIVAGE is about enabling interconnecting data resources (meta-data) from distributed geographical locations and when applicable their corresponding data values (measurements). Best practices should be applied to enhance efficiency in the ACTIVAGE AIoTES framework. Annotated data provide benefits by providing context about the readings or observations that are captured from the real world. But at the same time adds overhead in terms of communication and storage. Also access to information should be exposed as web resources, whereby they can be created, accessed, modified and removed. The following steps can be used as a guide to produce better data, this is an introductory section and more details will be provided in a next version of this document once the architecture and technologies are aligned within the different ACTIVAGE AIoTES framework components:

6.5.1 Registration of Data Resources

Registration of Resources and Virtual Entities must identify their respective semantic instances using identifiable URIs. This will allow descriptions to be managed and retrieved by using their URIs as URLs. Referenceable URIs must only be applied to instantiations of ACTIVAGE Resources, which in the ontology refers to the “ssn/sosa:Device” class or any of its subclasses (ssn:SensingDevice, sosa:ActuatingDevice) for example.

6.5.2 Describing Data

Descriptions can be annotated using any RDF serialization format, which include RDF/XML, JSON-LD and Triples (i.e. Turtle, or N3). It is expected to handle large amounts of data being produced by the Testbeds, and so annotation applied to raw/proprietarily-formatted data should be minimal, and any triples created must provide added-value for experimenter queries. This will alleviate unnecessary load on the platform or Testbed when it comes to delivery and storage.

6.5.3 Publishing Data

Announcing data is an exercise of two levels, metadata and data values, meta data, usually is open and used for announcements and distribution of related-sharing information, while it could refer also to personal data, it usually describes types of information but do not contain sensitive values (i.e. personal data values). Publishing data is necessary for enabling data discovery and data accessibility and particularly when data is part of the business model of a service or application this activity is important. Prior to publication, Testbed providers are encouraged to validate samples of their descriptions with an ontology validator to guarantee that the published data is compliant with the formats and schemas and thus be discoverable and accessible all the times.

6.5.4 Storing Data

Data storage is the mechanism for preserving digitised data measurements and produced information as part of the metadata structures and their values.

6.5.5 Accessing Data

Data access is one of the basic services of any Data/Information system. ACTIVAGE will provide two different service types with different data access patterns. The first one is a pull-based data access service. The second one is a push-based linked data stream service. For pull-based access, a client specifies the system entity, e.g. the ICO or sensor reading type that it wants to retrieve using its URI. The exact interfaces and their methods for pull and push-based mechanisms that these services will offer will be described in a subsequent version according to architectural designs.

6.6 Overview and Further Work

In this section we are including the activities towards defining and building the ACTIVAGE data pack. We have surveyed the state of the art data models in the relevant domains and identify and classify ontologies with respect to the available semantic models in the literature. The rationale to build the ACTIVAGE data pack is to provide the formats and structures about the data and related metadata that is or will be handle in ACTIVAGE ecosystem.

We focused on describing how data from different Deployment Site IoT-Platforms can be model, handle stored and utilized within the ACTIVAGE AIoTES framework, focusing on modelling and interoperability aspects but also including important aspects like Security & Privacy, Marketplace and Federation. In order to do so, semantic interoperability approach has been adopted as explained in this document. In ACTIVAGE there are 9 geographical locations named as deployment sites and each of them have specific IoT technology and as consequence local data model.

As part of the Interoperability activity in ACTIVAGE and in order to take part in the ACTIVAGE ecosystem, each deployment site will have to comply (according to their level of interest for interoperability compliance) with the defined ACTIVAGE Data Pack that contains the Ontology & Schemas and thus as required will have to annotate the data that they are providing to ACTIVAGE. This annotation will be the result of “bridging” the related IoT platform(s), using the ACTIVAGE data pack models and schemas the data can be processed and able to be provided via the AIoTES upon request using available Data APIs, Data and Information services and/or Data specific applications.

The motivation to build the ACTIVAGE data pack comes from:

(i) not overloading the domain with new ontologies but integrating various existing required ontologies (i.e. the needed concepts and terminology) into a single and holistic one in order to fulfil the needs of the multiple Deployment Sites,

(ii) reusing as much as possible the existing ontologies in order to help Deployment Sites not modify (as possible) their datasets in order to join ACTIVAGE Federation,

(iii) ensuring a better interoperability with existing semantics-based IoT platforms, projects, and standardizations.

iv) address security and privacy issues in a formal manner in order to not only protect data but also to be compliant with current regulatory activities in Europe (i.e. GDPR.)

The state of the art ontologies discussed in the deliverable also provide essence on why the ACTIVAGE data pack is suitable for being considered as contribution within Active and Healthy Ageing (AHA) community and build/complement the AHA ontology. Based on the initial analysis, the initial version of ACTIVAGE ontology currently relies on concepts from IoT, SSN/SOSA, HL7-V3 and Security & privacy ontologies and other taxonomy.

IoT integration of the overall existing Ontology work helps Deployment Sites, to not to recreate annotations that they already may have (although changes may be required according to the global approach rather than the local). SSN/SOSA, probably the most well-known IoT domain ontology pertaining to the sensing and actuating realm, provides the basic concepts, while other provide, above all, the taxonomy that cover all the different types of resources, physical phenomena and units of measurement fostered from the underlying Deployment Sites (it is worth highlighting that the vocabularies and taxonomy used across different deployment sites will be a living entity that might be updated with all the new elements coming from potential new Deployment Sites that become part of the ACTIVAGE federation). Furthermore, we are including concepts from other ontologies, like SAO, OpenIoT and SAREF, and other standardizations like oneM2M, HL7V3 (which we justifiably believe will have a big impact in the future). Above all, best practices have been followed within the ontology definition process. Within this frame; the ontology documentation, some sample annotations and the mapping between legacy Deployment Sites is provided, thus, any Deployment Sites as part of ACTIVAGE and in future also external could use this as a guideline to annotate their resources to the format defined in the scope of data pack in the ACTIVAGE ecosystem.

Moreover, we have also described some of the off-the-shelf annotation tools. Nevertheless, as the data is provided by the Deployment Sites, it is worth noting that ACTIVAGE platform does not execute or provide such annotation tools. ACTIVAGE only provides a reference annotation tool that can be used by the Deployment Sites as a base for the implementation of their own annotators that could be in the bridges integration or externally at the deployment site platform then interface with the AIoTES using the bridge. Apart from this annotation process, it is deemed necessary to validate the data injected from the underlying Deployment Sites in order to confirm that different datasets accomplish the templates defined in the ACTIVAGE data pack. This validation will only happen based on the policies (when and what to validate) specified by ACTIVAGE and that will be carried out within the AIoTES platform.

This section also provides insights to the guidelines and best practices for Deployment Sites to publish data (i.e. measurements) through the ACTIVAGE framework, either to store it or to forward it to experimenters. Such guidelines would ensure less overhead for ACTIVAGE in terms of data management and its usage.

There are number of open issues that have not been included in this version of the deliverable and will be addressed and included in the future iteration of this deliverable. Below we include a list of such currently open issues:

- As Deployment Sites might provide IoT data coming from mobile devices, it is worth investigating issues related to how this data will be stored and how the updates and/or availability will be published to ACTIVAGE framework.
- The need for real-time updates would potentially be satisfied by the publish-subscribe methodology and under investigation we will define if a component developed by ACTIVAGE could be integrated by Deployment Sites within their framework in the form of agents or this will be fully local responsibility of the deployment sites to cope with.
- The way to store different datasets (i.e. IoT services/resources, measurements, virtual entities, Deployment Sites, rules, etc.) is a point that will be thoroughly analysed. The most likely solution we have foreseen is to make use of JENA-based triple stores instanced in various functional components throughout the platform for metadata information.

Once we have saved the various meta-data in the ACTIVAGE framework, it will have to provide a way to interact with the data sources using the bridges so that consumers of the data/information services can be capable to produce value-added services by means extracting the information according to their needs. For that, the most widespread solution is no other but SPARQL.

The ACTIVAGE AHA ontology and its taxonomy are finished versions. As long as we identify missing elements (e.g. coming from the feedback from other tasks or external Deployment Sites fostered from the ACTIVAGE Open Call processes), we will modify and update the ontology.

6.6.1 ACTIVAGE AHA Specific Domains

The specific domain ontologies in ACTIVAGE are part of further design and ontology engineering work, once the use of the core AHA ontology and the ACTIVAGE AIoTES Data Pack will be generalised, deployment sites can potentially promote, the domain specific ontologies as extension to the core ontology, the domain of the deployment site is related to the activity the different stakeholders are involved and thus the domain specific ontology is called or associated to that activity as follow:

- Integrated Care for Chronic Conditions Ontology
- Monitoring Outside Home Ontology
- Emergency Trigger Ontology
- Exercise Promotion Ontology
- Cognitive Stimulation Ontology
- Prevention of Social Isolation Ontology
- Safety, Comfort and Security at Home Ontology
- Support for Transportation and Mobility Ontology

In this regard, it will be employed the naming conventions already explained in 4.4.1.2.

7 Security & Privacy Considerations

This section draws relevant considerations for security and privacy on data sharing and interoperation among IoT systems and platforms.

Sharing data is inherently insecure, the first process in a security analysis is to determine whether the data needs to be shared at all. Equally, the act of sharing of data is the enemy of privacy: any information which is susceptible of privacy considerations it is best not shared. Having said this, interoperability requires data exchange. Thus, it is critical to ensure the security and privacy of the interoperability operations.

Security, and particularly Privacy are especially relevant considerations in the AHA domain, as the nature of the data is especially sensitive. In this context, the Security and Privacy considerations many times go beyond APIs, and implementations. For this the deliverable D3.3 (and successive versions) we will provide a set of guidelines.

Each deployment site has their own security and privacy management, as well as different IoT platforms which may, or may not, implement different security and privacy mechanisms. If we aim to achieve interoperability between deployment sites, and thereof between IoT platforms, we must define security and privacy interoperability. For example, a common practice for ensuring security is end-to-end encryption. If each end is a module on different platforms, then this means establishing a set of protocols or rules so that this could be achieved across platforms.

7.1 Security

Security is a fundamental aspect to be considered when more than one client or application have access to the IoT system. All security mechanisms are based on Trust Management, mitigating the risks that untrusted entities attain the trust (confidentiality and integrity), while allowing trusted entities to operate normally (availability and non-repudiation). This trust is placed on many system components on device, data processing, connectivity, and at the overall system level; each with its unique challenges.

7.2 Privacy

Merriam-Webster defines privacy as “freedom from unauthorised intrusion” [28]. In the context of interoperability, it means that only authorised actors can access sensitive data. For this, Privacy will rely on security functions.

To facilitate interoperability of sensitive data, such as personal data, it should be structured, commonly used, machine-readable and expressed in an interoperable format. Data controllers should be encouraged to develop interoperable formats that enable data portability as indicated in the GDPR (General Data Protection Regulation) [29].

Yet to ensure the property of privacy is enforced, the SIL should add a layer of obfuscation to this data. This layer may materialize by data encryption and/or by data anonymization. Particularly in Europe GDPR must be accomplished. Deliverable D3.3 provides guidelines to cope with GDPR and deliverable D5.1 fully details the general use case and implications of complying with GDPR as in an AHA initiative the privacy (personal data protection) is a major concern.

Since the SIL is conceptualized within Semantic Interoperability, it is worth analysing the implications on Privacy. Semantics interoperability plays a key and ultimate role in the overall system interoperability, especially in applications making use of distributed resources, even more specifically when these resources are accessed through gateways. This is equally

important for privacy, since two different systems have to match some syntax and semantics at the upper levels of abstraction not only in the technical domain but also in the ethical and legal domains. In this regard, the GDPR is a key regulation to ensure compatibility across the "European" ecosystem by relying on a common legal and understanding basis. Subsequently, semantics may be also used to check that privacy rules as well as restrictions are understandable and agreed in the compliant ecosystem. As a consequence, the knowledge and the spirit of the GDPR should be transduced into a validated ontology. Work in that direction is recent but consistent, as shown by recent research papers [29] and initiatives [30].

7.3 Security and Privacy Module

One of the main system component related with the achievement of interoperability, as far as it enables a secure communication and data sharing, will be the security and privacy management module, as can be seen in Figure 30.

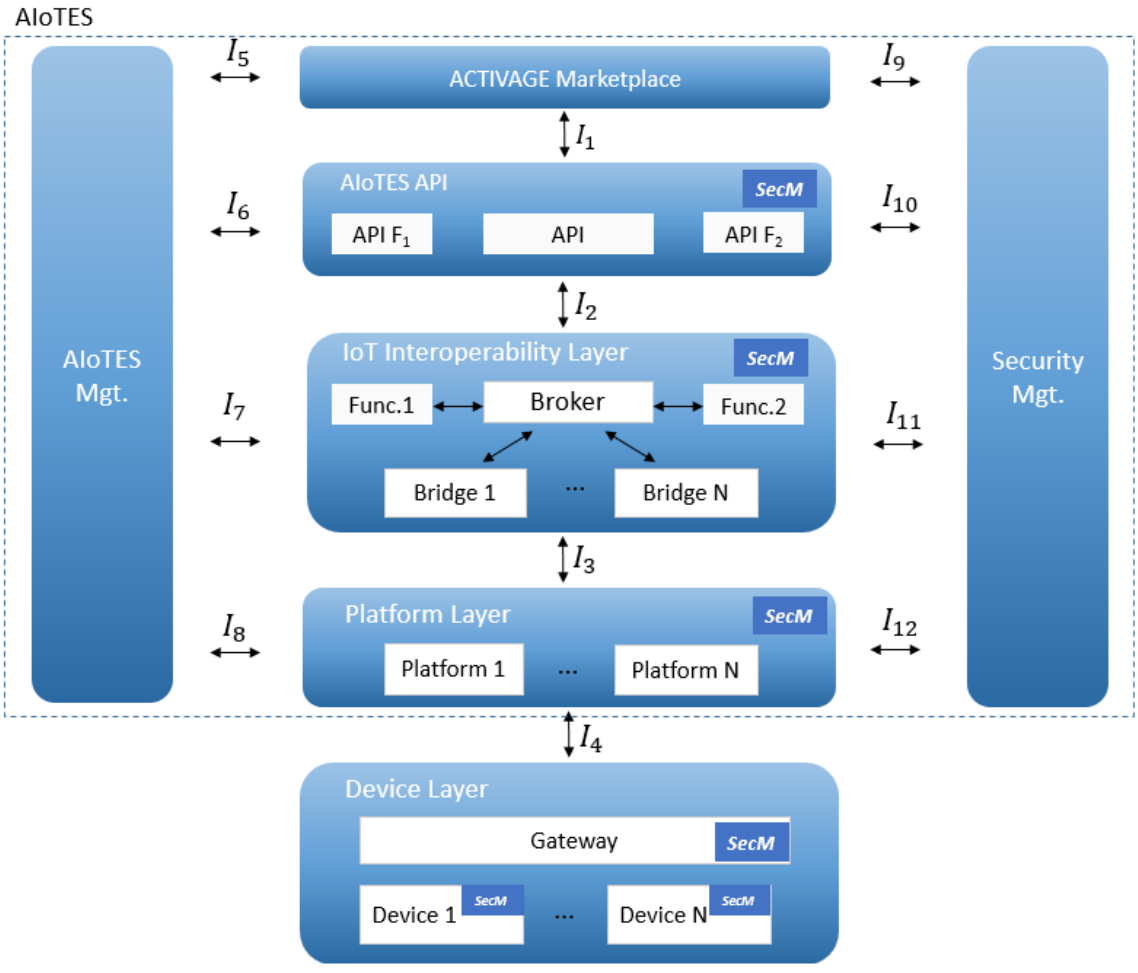

Figure 30: ACTIVAGE High-Level Architecture

This module will offer a common access to all actors to perform the following normalized security and privacy functions, which are described in more details in deliverable D5.1:

Access Control. It is in charge of identifying, authenticating and authorising users to the different resources, including the system, while keeping trackable records of these operations. In terms of interoperability, it is essential that when the different systems refer to a user, they

use the same identification, in order to ensure the data associated to a user ID is correctly interpreted across platforms, and applications. Another important caveat of Access Control which is particular in IoT systems and essential for AHA users is the single-sign-on(SSO), users use a single set of credentials to authenticate. For any user, but particularly for elderly users, it is increasingly difficult to memorize different set of credentials for every service, which forces the use of unsecure credential practices (such as sharing passwords, or writing them down on papers); thus in terms of security it is a benefit to implement SSO.

The current industry standard offering SSO, with secure underlying identification and authentication protocols (which may include multi-factor and/or biometric authentication) is Lightweight Directory Access Protocol (LDAP), the latest specification of which is Version 3, published as RFC 4511. All ACTIVAGE platforms, services and applications may use LDAP as the method to interoperate Access Control mechanisms. This usage will also resolve the issue of identification interoperability, by using LDAP URL schema. Each user can be identified by a URI (Unique Resource Identifier), or an IRI (Internationalized Resource Identifier). This allows uniquely identifying each user, independently of Deployment Site, service or Application.

Security Association Establishment. It establishes an authenticated channel between two entities A and B to ensure confidentiality and integrity in their communication. This service, available at AIoTES level, enables end-to-end secure communication. There are different stages of this establishment, such as handshake between entities, mutual entity authentication, key establishment and secure session management, before the data can be securely exchanged.

Information security standards are essential to ensure interoperability among systems and networks, compliance with legislation and adequate levels of security. These standards provide a means for protecting the user, creating a more secure and profitable environment for the industrial sector, from SMEs to large global companies.

Communication standards (protocols) that should be considered at system level implementation to ensure security interoperability are the following:

- **IPSec**: It is a protocol suite used to protect IP traffic through encryption and authentication. De facto standard Virtual Private Network (VPN) protocol, which is used to channel between networks or node and a remote network. IPsec protocols can work in transparent mode (the data payload is protected) or tunnel mode (the payload and headers are protected). In IPsec, AH (Authentication header) provides integrity and authentication, and ESP (Encapsulating Security payload) provides additional confidentiality.
- **SSL** (Secure Sockets Layer): It uses public key encryption and provides data encryption, server authentication, message integrity, and optional client authentication. In the protocol stack, SSL lies beneath the application layer and above the network layer, known as the transport layer meaning it channels communications between hosts.
- **TLS** (Transport Layer Security): It is the open-community and standardized version of SSL. The difference between SSL 3.0 and TLS are slight, but TLS is more extensible, it is backward compatible with SSL and interoperable with other technologies.
- **HTTPS** (Hypertext Transfer Protocol Secure): It is HTTP running over SSL. HTTP works at the application layer and SSL works at the transport layer.

Sensitive Data Handling. It assures the confidentiality, integrity and non-repudiation of the data at rest as well as the data in transit. In order to support these procedures, the service provides various functions, which focus on performing cryptographic operations such as encryption or decryption of the data and execution of hash functions, for digital signature. The

implementation of this sensitive data handling service must consider the cipher suite as the bases for its design and implementation.

Security Administration. It is a transversal set of functions that help configure and assure the correct function of the security and privacy functions. These functions typically are handled by system administrators, and include key management (such as Public Key Infrastructure), User Privileges versus Object Privileges, and remote security provisioning (including thing like remote permissions, and Certificate Authority).

Risk management and monitoring. They are the complementary tools to security administration, in the sense that they look at the system as a vulnerable system. These tools categorize Risks and attempt to find new risks during the operation, or modification, of the system. For this, they need a complete log auditing, which in turn will allow to detect intrusion attempts, potential threats, as well as actively deterring these attempts and minimize threats.

The most important base for monitoring is to be able to handle audit logs, which themselves need to be secured. For ACTIVAGE, a Security Information & Event Management (SIEM) analyser is recommended to centralize messages from different probes. Those messages should be standardised in order to allow flexibility and facilitate AIoTES dedicated extensions. Such standard should cope with “Intrusion Detection Message Exchange Format” (IDMEF - RFC4765).

Privacy Preservation. The most secure way of preserving privacy is to avoid disclosing any information, including the fact that a natural person “exists” in a digital form. Of course, this is at least inconvenient and may also be harmful if you are in need for digital services, in particular healthcare and assistance. Thus, we need mechanisms to allow the interoperability of services and data compliant with the person’s right to privacy.

A security ontology will be defined, in Deliverable 3.5 and not later than month 24 of the project, to describe specific classes (such as anonymized, to annotate already anonymized data) and particularly the properties of identifiers and quasi-identifiers. Identifiers are attributes which directly identify people who consents the processing of his/her personal data, whereas quasi-identifiers may identify the person through indirect means and combination of quasi-identifiers. In terms of ontologies, a specific statement might be added to specific ontologies to identify which properties can be considered Identifiers and which quasi-identifiers. This annotation property will allow the SIL itself to detect data which is not compliant with the user’s consent, and this way enforce the privacy preservation. Tools for consent management, both for user and data controllers, will be provided with AIoTES thanks to the work done in WP4. A consolidated version of these tools will be available at the beginning of the third year of ACTIVAGE.

8 Developer and user guide

This section details the several user guides available to support users willing to implement any use cases involving IoT platform interoperability as described in the section 4, which involve the deployment and use of the SIL. The first subsection, 8.1, presents the involved users in these interoperability use case. The next three subsections detail the several guides and trainings that should answer the most important questions to fulfil interoperability:

How to deploy and setup the SIL? How to use the SIL? How to make a platform interoperable with all others through the SIL?

8.1 Users

A detailed list of stakeholders in the ACTIVAGE project has been defined in the deliverable D2.1. From this complete list, we filtered and synthesized the users with interest on interoperability concerns: the three classes of stakeholders are Technical developers and service providers, deployment sites managers and newcomers. These three user classes are detailed below.

Technical developers and service providers: Technical developers presents the first positively affected categories by the support tools, mainly to enable the usage and the exploitation of available APIs and functions from different platforms, to ease the quick development of solutions though the Visual programming tools. The technical team should be also later able to easily extend the system, combine it with others, and easily add to remove further devices…Five different profiles of developers have been detailed in D4.1. We focus on the three categories of developers listed below:

- AIoTES Expert which typically integrates an existing IoT platform within AIoTES. This expert has a leading development role and the ability to specify roadmaps for further developments.
- AIoTES developer, which compared to the AIoTES expert doesn't have necessary knowledge about the underlaying platforms but can develop bridges for AIoTES.
- Application developer which develops on top of AIoTES services which have the ability to be replicable over many deployment sites. This role is highly bound to AIoTES and its successivity to develop efficient services thanks to AIoTES is strategic for the project.

Deployment sites managers, which are the technical referents of the deployment sites regarding the different uses cases and services deployed.

- DS Administrator which has authority onto the deployment sites, in particular regarding the user management (inclusion and removal of user), the management of platforms and devices, and have to report on a regular basis strategic KPI to the ACTIVAGE board and potentially to local authorities.
- DS Device installers and support teams which happens to be in front row with the users (beneficiaries, caregivers, etc) during the whole lifecycle of the devices. It is one of some entry point for those users to report issues or suggest improvements.

Newcomers

- Open call participant which can be one of the different roles but without prior knowledge to the project and AIoTES in particular.

- Other parties such as IoT or services suppliers willing to benchmark their tools and mechanisms using ACTIVAGE technology, test beds, and data sets, and to evaluate the adequacy of ACTIVAGE to their domains, and for developing extended and new use cases based on the ACTIVAGE platform.

Table 15: Summary of stakeholders with interoperability interests

Stakeholder	Description
Application developer	develops over AIoTES framework
AIoTES expert	deploys AIoTES, integrates platform in AIoTES
AIoTES developer	develops AIoTES bridges
DS administrator	manages the DS, including users, platforms, and devices
DS device installer	manages the devices of a DS
open call participant	individuals, local authorities, service providers, associations, organizations and businesses. Invited third parties interested to get involved in specific tasks that will be carried out with the ultimate goal to expand ACTIVAGE use cases and apply experiments of new cities
other third parties involved in development	IoT or service suppliers interested for evaluation by Activage ecosystem (GROW and SUSTAIN phases)

8.2 SIL integration guide

This section aims to outline the deployment and integration process of the Semantic Interoperability Layer (SIL) on AIoTES. The first subsection will describe the way in which the components, that made up the SIL, can be downloaded, configured and deployed; while the second subsection will provide the reader with clear examples of the interactions with the interfaces presented.

8.2.1 SIL deployment and configuration

The SIL is not a standalone service but a set of microservices that interact between each other in order to provide the required interoperability.

8.2.1.1 Prerequisites

Prerequisites for the SIL installation are:

- Docker and Docker Compose: Docker enables to use a Docker container image, a lightweight, stand-alone, executable package of a piece of software that includes everything needed to run it: code, runtime, system tools, system libraries, settings. Docker Compose is a tool for defining and running multi-container Docker applications. The SIL is virtualized and has a Docker image that enables an easy and fast deployment.
- access to the **commons** repository[19]

[19] https://git.inter-iot.eu/Inter-IoT/commons after log-in in https://git.inter-iot.eu

- access to the INTER-IoT Docker registry docker.inter-iot.eu

8.2.1.2 Deployment

This microservices are distributed as Docker images, some of them are public images from open source projects, and some others are custom developments, stored in private registries.

In order to simplify the deployment of this services a Docker-Compose file is created containing the following images:

Table 16: Microservices

Image	Description
Intermw	INTER-MW application
Rabbitmq	RabbitMQ message broker
Parliament	Parliament triple store
Ipsm	Inter-Platform Semantic Mediator
Kafka	Apache Kafka
Zookeeper	Apache ZooKeeper
Postgres	PostgreSQL database
Adminer	GUI SQL database admin panel

In this way the build and deployment of this services can be performed executing:

```
$ docker-compose up
```

8.2.1.3Configuration

Even when the deployment process has been simplified, a configuration phase is still required in order to reach custom scenarios that can satisfy special needs. for this purpose, a repository, accessible in the following link has been created with the structure depicted in the following Table.

```
$ git clone https://git.inter-iot.eu/Inter-IoT/commons
```

Table 17: Repository files

Repository file or table	Description
certs	sample (self-signed) SSL certificates used by the Apache Kafka container
certs-client	sample SSL client certificate, used by the IPSM container
db-scripts	initialization scripts used by the PostgreSQL container
kafka-scripts	initialization script for the Apache Kafka container
environment	files with definitions of various environment variables providing configuration for the Docker containers.

docker-compose.yml	definition of the deployment (to be used with Docker Compose).

8.2.1.4 Configure environmental variables

Inside the repository, navigate to the commons/intermw-ipsm-deployment directory and configure Compose variables in the .env file. The following are the default values for the environmental variables

Table 18: Configuration of environmental variables

```
INTERMW_VERSION=v2.1.0
INTERMW_REST_API_PORT=8080
INTERMW_CALLBACK_PORT=8980
RABBITMQ_PORT=5672
RABBITMQ_ADMIN_PORT=15672
RABBITMQ_USER=admin
RABBITMQ_PASS=admin
PARLIAMENT_PORT=8089

IPSM_VERSION=0.5.4.1
IPSM_REST_API_PORT=8888
ZOOKEEPER_PORT=2181
KAFKA_PORT=9092
```

8.2.1.5 Configure IPSM component

In order to install and configure the IPSM component the following steps are required:

1. Clone the main IPSM development git repository

   ```
   $ git clone https://git.inter-iot.eu/Inter-IoT/ipsm-deployment
   ```

2. Edit the environment/kafka_vars.env file and change the 10.0.2.15 IP address to the IP address of your deployment machine (don't use localhost nor 127.0.0.1). The other configuration options contained in the environment/*.env files can be left unchanged if the deployment does not explicitly require changing them.

8.2.1.6 Initialize docker volumes

Before invoking docker-compose one additional step is required. You need to create two named volumes for holding PostgreSQL and Apache Kafka data, security certificates, and container initialization scripts.

Under Linux (OS-X)

```
$ bash initialize-volumes.sh
```

Under Windows using the Powershell window:

```
$ .\initialize-volumes.ps1
```

If the deployment takes place under Windows make sure that the drive containing the cloned repository files has been "shared" via Docker->Settings->Shared Drives.

8.2.1.7 SSL configuration

By default, the Apache Kafka and IPSM utilize SSL certificates contained in the certs and certs-client folders, respectively. Also the SIL deployment configuration uses the same client certificate by default, so you don't need to make any changes here. A detailed description of the SSL configuration for Apache Kafka can be found in the **official documentation**[20].

8.2.1.8 Accessing logs

Logs of any of the deployed components can be accessed via docker-compose. You need to be in the main directory of the cloned repository (i.e., the one containing the docker-compose.yml file). To see the logs for a specific container, issue the command

```
$ docker-compose logs container-name
```

where "container-name" is one of: zookeeper, kafka, postgresql, adminer, and ipsm.

8.2.1.9 Testing current installation

To make sure that everything went as expected use your browser to access the REST API swagger interface. Both swagger interface and the REST API are by default available on port 8888.

8.2.2 Use of SIL

This section will show the usage of the SIL API through its main functions.

SIL installation can be tested using platform emulator to make sure that everything works correctly. Platform emulator, this tool is a universAAL component created for tests purpose, is available as a Docker image from the INTER-IoT docker registry. To start the emulator, run the following command:

```
$ docker run -d --name platform_emulator -p 4568:4568
docker.inter-iot.eu/intermw-platform-emulator:0.0.1
```

The emulator port 4568 is bound to the host machine.

In order to outline the SIL capabilities an example is going to be followed. Assuming that we have an UniversAAL platform, this section shows the operations over the SIL.

The operation order is not casual but mandatory. First, the IPSM component must be configured, then the client has to be registered in order to register the platform and the device. In the same way, the platform must be registered before the device. Therefore, the successful consecution of the deploy and configuration only can be done by following the steps described in the order provided.

8.2.2.1 Configure IPSM - Creating Channel

This configuration step must be done only the first time the platform is deployed.

[20] http://kafka.apache.org/documentation.html#security_ssl

Expand Channels section and use POST /channels operation to create new channel. Use the following JSON data as input:

```
{
"source": "mw-ipsm-platform-format-UniversAALEmulated",
"inpAlignmentId": <alignment-id>,
"outAlignmentId": 0,
"sink": "ipsm-mw-interiot-format-UniversAALEmulated",
"parallelism": 1
}
```

where <alignment-id> is the ID of UniversAAL_CO_align alignment created in previous step.

8.2.2.2 Registering client

Register client with id pwt001:

```
curl -X POST --header 'Content-Type: application/json' -d
'{"pullMessagesLimit":5}'
"http://localhost:9080/mw.api.rest/api/intermw/client/pwt001"
```

8.2.2.3 Registering platform

Register platform of type UniversAALEmulated:

```
curl -X POST --header 'Content-Type: application/json' -d
'{"id":{"id":"http://inter-iot.eu/universaal-
emulated"},"type":{"typeId":"UniversAALEmulated"},"capabilitie
s":[],"baseURL":"http://172.17.0.1:4568/platform_emulator","na
me":"UniversAALEmulator platform"}'
"http://localhost:9080/mw.api.rest/api/intermw/platform/pwt001
"
```

where baseURL is platform emulator address which must be accessible from intermw container. Since emulator port is bound to host machine, we can use host machine IP address 172.17.0.1:4568.

Retrieve confirmation message using the command:

```
curl -X POST --header 'Content-Type: application/json'
"http://localhost:9080/mw.api.rest/api/intermw/pull/pwt001/1"
```

If successful, the confirmation message will contain following types:

```
"@type" : [ "InterIoTMsg:Response",
"InterIoTMsg:Platform_register", "InterIoTMsg:meta" ]
```

8.2.2.4 Registering device

Register device UniversAAL-401-Sensor-weight:

```
curl -X POST --header 'Content-Type: application/json' -d
'{"attributes":[],"platformId":{"id":"http://inter-
```

```
iot.eu/universaal-
emulated"},"thingId":{"id":"http://www.example.com/UniversAAL-
401-Sensor-weight"}}'
"http://localhost:9080/mw.api.rest/api/intermw/thing/pwt001"
```

Retrieve confirmation message using the command:

```
curl -X POST --header 'Content-Type: application/json'
"http://localhost:9080/mw.api.rest/api/intermw/pull/pwt001/1"
```

If successful, the confirmation message will contain following types:

```
"@type" : [ "InterIoTMsg:Response",
"InterIoTMsg:Thing_register", "InterIoTMsg:meta" ]
```

8.2.2.5 Subscribing to device

Subscribe to observations from device UniversAAL-401-Sensor-weight:

```
curl -X POST --header 'Content-Type: application/json' -d
'{"attributes":[],"platformId":{"id":"http://inter-
iot.eu/universaal-
emulated"},"thingId":{"id":"http://www.example.com/UniversAAL-
401-Sensor-weight"}}'
"http://localhost:9080/mw.api.rest/api/intermw/subscribe/pwt00
1"
```

Retrieve confirmation message using the command:

```
curl -X POST --header 'Content-Type: application/json'
"http://localhost:9080/mw.api.rest/api/intermw/pull/pwt001/1"
```

If successful, the confirmation message will contain following types:

```
"@type" : [ "InterIoTMsg:Response", "InterIoTMsg:Subscribe",
"InterIoTMsg:meta" ]
```

8.2.2.6 Retrieving observations

Retrieve observation messages for the client pwt001, five at a time:

```
curl -X POST --header 'Content-Type: application/json'
"http://localhost:9080/mw.api.rest/api/intermw/pull/pwt001/5"
```

Observation messages contain following types:

```
"@type" : [ "InterIoTMsg:Response", "InterIoTMsg:Observation",
"InterIoTMsg:meta" ],
```

Emulator is sending observations one by one with a few seconds delay.

8.2.3 Add a platform-specific bridge to SIL

This section intends to explain the step that must be followed to integrate a platform-specific bridge in the SIL.

Bridge installation to an SIL instance consists of the following steps:

- copying bridge JAR file together with its dependencies to the library inside the inter-mw Docker container
- copying bridge configuration file (if any) to the inter-mw configuration directory
- restarting inter-mw image
- creating IPSM alignments and channels for the bridge

8.2.3.1Installing Bridge JARS

Build and package the bridge by running following Maven command:

```
$ mvn clean package
```

Copy the bridge JAR file created in the previous step to `lib` directory of INTER-MW webapp inside the INTER-MW Docker container using the following command:

```
$ docker cp <path-to-bridge-jar-file> <intermw-container>:/usr/local/tomcat/webapps/ROOT/WEB-INF/lib
```

To check whether bridge JAR file is available within the INTER-MW library, run the following command:

```
$ docker exec <intermw-container> ls -l /usr/local/tomcat/webapps/ROOT/WEB-INF/lib | grep <bridge-name>
```

In addition, you also have to copy into the INTER-MW library (i.e. `lib` directory) all runtime dependencies that are required by the bridge project and don't already exist there. Make sure that dependencies added to INTER-MW library don't cause dependency conflict. The mw.bridges.api dependency together with its sub-dependencies are already included in the INTER-MW library.

You can get a list of all dependencies available in INTER-MW library by listing `lib` directory content:

```
$ docker exec <intermw-container> ls -l /usr/local/tomcat/webapps/ROOT/WEB-INF/lib
```

To display dependency tree for the bridge project (runtime dependencies), run the following command:

```
$ mvn dependency:tree -Dscope=runtime
```

8.2.3.2 Installing Bridge Configuration File

In case the bridge requires a configuration file (e.g. bridge.properties), it has to be copied to the SIL configuration directory:

```
$ docker cp src/main/resources/<bridge-config-file> <intermw-container>:/etc/inter-iot/intermw
```

8.2.3.3 Restarting SIL

Finally, SIL has to be restarted to detect the new bridge. Run following command inside the SIL Docker Compose deployment directory from where the SIL was deployed:

```
$ docker-compose restart intermw
```

To make sure new bridge was detected and registered successfully, search for the string `Registering bridges...` in INTER-MW log file:

```
$ docker-compose logs intermw | grep -A 5 "Registering bridges"
```

In case of ExampleBridge, the following log statements can be found in the logs:

```
DEBUG
eu.interiot.intermw.bridge.Context.registerBridges(Context.jav
a:208) - Registering bridges...
DEBUG
eu.interiot.intermw.bridge.Context.registerBridges(Context.jav
a:210) - Scanning package eu.interiot...
DEBUG
eu.interiot.intermw.bridge.Context.registerBridges(Context.jav
a:214) - Following bridges have been found: [class
eu.interiot.intermw.bridge.example.ExampleBridge]
DEBUG
eu.interiot.intermw.bridge.Context.registerBridges(Context.jav
a:220) - Bridge
eu.interiot.intermw.bridge.example.ExampleBridge for platform
type http://example.inter-iot.eu/example-platform has been
registered.
DEBUG
eu.interiot.intermw.bridge.Context.registerBridges(Context.jav
a:224) - Bridge registration has finished successfully.
```

The term bridge registration means that the bridge class has been related to the corresponding platform type. Bridge is instantiated (an instance of Bridge class is created) when an actual platform is registered for specific platform type.

8.3 SIL development guide

8.3.1 Guide for bridge development

Bridges are implemented as Maven projects. A bridge project has two main classes, the bridge implementation class, which manages the communication between the Interoperability Layer and the IoT platform, and the translator class, which performs the syntactic translation.

A bridge requires the following Maven dependencies:

```
<dependency>
    <groupId>eu.interiot.intermw</groupId>
    <artifactId>mw.bridges.api</artifactId>
    <version>${intermw.version}</version>
</dependency>


<dependency>
    <groupId>eu.inter-iot.translators</groupId>
    <artifactId>syntactic-translators</artifactId>
    <version>1.0</version>
</dependency>
```

Bridge implementation class

The bridge implementation class must extend the AbstractBridge class, which implements the Bridge interface, and provide a target platform type using the annotation @Bridge:

```
@Bridge(platformType = "http://example.inter-iot.eu/example-platform")
public class ExampleBridge extends AbstractBridge { ... }
```

When a platform is registered, the Interoperability Layer creates an instance of its corresponding bridge implementation class (according to the platform type). The bridge constructor has two input parameters, which provide information about the bridge configuration and the instance of the platform, respectively:

```
public ExampleBridge(Configuration configuration, Platform platform)
```

The bridge configuration is read from the properties files that match the filter *.bridge.properties. The common bridge configuration properties are stored in the bridge.properties file. The configuration properties of a specific bridge should have a bridge name prefix and be stored in a separate file (e.g. example.bridge.properties).

```
# ExamplePlatform
example.myproperty=my-property-value
```

The bridge implementation class must implement the methods defined in the AbstractBridge class:

- Message **registerPlatform**(Message message): Registers the specified platform in the SIL.
- Message **unregisterPlatform**(Message message): Unregisters the specified platform.
- Message **subscribe**(Message message). Subscribes client to observations provided by the platform for the specified device. Must implement a listener for notifications and translate the platform data to SIL messages.
- Message **unsubscribe**(Message message): cancels the specified subscription created by the subscribe method.
- Message **query**(Message message): performs a query about a status and last observation made by a specified device.

- Message **listDevices**(Message message): performs a query to a platform to get all devices managed by it.
- Message **platformCreateDevices**(Message message): an instruction for a platform to start managing a new device.
- Message **platformUpdateDevices**(Message message): an instruction for a platform to update information about a device.
- Message **platformDeleteDevices**(Message message): An instruction for a platform to stop managing a device.
- Message **observe**(Message message): pushes given observation message from the SIL to the platform.
- Message **actuate**(Message message): pushes actuation instructions to the platform.
- Message **error**(Message message): error handling.
- Message **unrecognized**(Message message): error handling

Translator class

The translator class must extend the abstract class SyntacticTranslator<FormatX>, where FormatX represents the data format of the IoT platform.

```
public class FIWAREv2Translator extends SyntacticTranslator<String> { ... }
```

The SyntacticTranslator class constructor has two input parameters, the base URI for the RDF entities generated by the translator and a String representing the name of the platform's data format (understandable by humans).

The translator must implement the two methods of the SyntacticTranslator class:

- public Model **toJenaModel**(FormatX formatX): Syntactic translation from the platform's format to Inter-MW JSON-LD.
- public Model **toFormatX**(Model jenaModel): Syntactic translation from INTER-IoT JSON-LD to the platform's format.

8.3.2 Guide for alignment development

Alignment can be interpreted as a set of uni-directional mappings for transforming an input RDF graph into an output RDF graph. These mappings allow the translation from a semantic model of a platform to the INTER-IoT/AIoTES semantic model. In this way it is necessary to create an xml file with this mapping information. This file must be employed in the IPSM configuration for that specific ontology-to-ontology semantic translation.

8.3.2.1 Structure

As mentioned, the structure is based on XML. Note, that in the short future, IPSM will support RDF alignment files, with the next IPSM version. The resulting change to the document structure will not require to modify the logic of translation, but to place several elements differently. In the following sections two ways of persisting the information will be presented if it is necessary.

The following code snippet presents an alignment structure in XML.

```
<Alignment id="align_id" version="align_version" creator="align_creator"
description="align_desc">
<onto1>
   <align:location>[source ontology location]</align:location>
   <align:formalism>
      <align:Formalism align:name="[source ontology formalism name]"
align:uri="[source ontology formalism URI]" />
   </align:formalism>
</onto1>
<onto2>
   <align:location>[target ontology location]</align:location>
   <align:formalism>
      <align:Formalism align:name="[target ontology formalism name]"
align:uri="[target ontology formalism URI]" />
   </align:formalism>
</onto2>
<steps>
 <step order="[cell order]" cell="[cell id]"/>
 { more steps }
</steps>
<map>
 <Cell id="[cell id]">
   <entity1> { source RDF pattern } </entity1>
   <entity2> { target RDF pattern } </entity2>
   <transformation>
     { functional constraints }
   </transformation>
   <filters> { datatype constraints } </filters>
   <typings> { typing info } </typings>
 </Cell>
 { more Cells }
</map>
</Alignment>
```

The following code snippet presents an alignment structure in XML/RDF.

```
<?xml version='1.0' encoding='utf-8' standalone='no'?>
<rdf:RDF xmlns="http://www.inter-iot.eu/interiot#"
   {other xml namespaces} >
<align:Alignment>
   <dcelem:title>[alignment title]</dcelem:title>
   <exmo:version>[alignment version]</exmo:version>
   <dcelem:creator>[alignment creator]</dcelem:creator>
   <dcelem:description>[alignemnt description]</dcelem:description>

   <align:xml>yes</align:xml>
   <align:level>2IPSM</align:level>
   <align:type>**</align:type>

   <align:method>[method]</align:method>
   <align:time>[time]</align:time>

   <align:onto1>
      <align:Ontology rdf:about="[source ontology uri]">
         <align:location>[source ontology location]</align:location>
         <align:formalism>
            <align:Formalism align:name="[source ontology formalism name]"
align:uri="[source ontology formalism URI]" />
         </align:formalism>
      </align:Ontology>
```

```
   </align:onto1>
   <align:onto2>
      {target ontology information}
   </align:onto2>

   <interiot:steps rdf:parseType="Literal">
      <interiot:step interiot:order="[cell order]" interiot:cell="[cell
id]"/>
      { more steps }
   </interiot:steps>

    <align:map>
      {alignment cell 1}
    </align:map>
        ...
    <align:map>
        {alignment cell n}
    </align:map>
</align:Alignment>
</rdf:RDF>
```

The most crucial part - translation logic - is contained inside the *cell* elements. Each *cell* is applied in an order specified in the *steps* element.

8.3.2.2 Metadata

The following table contains descriptions of fields treated as metadata - they are parsed when the alignments are stored in the IPSM semantic repository, but not in the translation process itself.

Element/ Attribute	Element/ Attribute in new format version	Meaning
Id	dcelem:title	Name of the alignment e.g. SSN_Observation_SAREF_Mesurement
Version	exmo:version	Version of the alignment e.g. 1.0
Creator	dcelem:creator	Author of the alignment
Description	dcelem:description	Comment on what is the scope/aim of the alignment

8.3.2.3 Alignment cells

Each alignment cell represents one “step” in the translation process. The cell applies SPARQL UPDATE to the RDF graph resulting from previous step (or source graph in case of first step). The SPARQL UPDATE statement is generated from “triples pattern” found in elements *entity1, entity2, transformations, typings.*

The following example can be read as subject *interiot:node_CTA* is related to object *interiot:node_CTB* with predicate *sosa:madeBySensor*. Prefix *interiot* with arbitrary name is used to denote variables that can be reused e.g. to later specify content of *entity2*.

```
<interiot:node_CTA>
      <sosa:madeBySensor>
       <interiot:node_CTB/>
      </sosa:madeBySensor>
<interiot:node_CTA>
```

If additionally, we want to restrict on type, the example can be changed to the following:

```
<interiot:node_CTA>
   <rdf:type rdf:resource="&sosa;Observation"/>
      <sosa:madeBySensor>
         <interiot:node_CTB>
            <rdf:type rdf:resource="&sosa;Sensor"/>
            </interiot:node_CTB>
      </sosa:madeBySensor>
<interiot:node_CTA>
```

This time we are not only matching two entities related with a property but entities of specified type related with a property.

The RDF graph structure pattern can be more complex e.g.

```
<interiot:node_CTX>
   <rdf:type rdf:resource="&p1ont;PositionObservation" />
   <ssn:observationResult>
       <interiot:node_CTY>
           <rdf:type rdf:resource="&p1ont;PositionSensorOutput" />
           <ssn:hasValue>
               <interiot:node_CTU/>
           </ssn:hasValue>
           <ssn:isProducedBy>
               <interiot:node_CTS/>
           </ssn:isProducedBy>
       </interiot:node_CTY>
   </ssn:observationResult>
   <ssn:observationResultTime>
       <interiot:node_CTZ/>
   </ssn:observationResultTime>
   <ssn:featureOfInterest>
       <interiot:node_CTW/>
   </ssn:featureOfInterest>
   <ssn:observedBy>
       <interiot:node_CTV/>
   </ssn:observedBy>
</interiot:node_CTX>
```

Here, we want to match all instances of *&p1ont1;PositionObservation* class that have asserted properties *ssn:observationResult*, *ssn:observationResultTime*, *ssn:featureOfInterest*, *ssn:observedBy*. Value if ssn:observationResult is an entity of type *&p1ont1;PositionSensorOutput* with asserted property *ssn:hasValue*.

Lets assume that we want to translate the above structure into:

```
<interiot:node_CTX>
   <rdf:type rdf:resource="&sosa;Observation"/>
   <sosa:observedProperty>
       <rdf:Description>
           <rdf:type rdf:resource="&co;Position" />
       </rdf:Description>
   </sosa:observedProperty>
   <sosa:hasFeatureOfInterest>
       <interiot:node_CTW/>
   </sosa:hasFeatureOfInterest>
   <sosa:hasResult>
       <interiot:node_CTU/>
   </sosa:hasResult>
```

```
    <sosa:madeBySensor>
        <interiot:node_CTS/>
    </sosa:madeBySensor>
    <sosa:resultTime>
        <interiot:node_CTZ/>
    </sosa:resultTime>
</interiot:node_CTX>
```

This RDF graph has the same information content, however it is expressed in SSN/SOSA ontology and not SSN as it was in the case of input RDF graph. In IPSM the pair or source and target RDF graph is transformed into the following SPARQL UPDATE statement.

```
DELETE {
      ?node_CTX <http://purl.oclc.org/NET/ssnx/ssn#observedBy> ?node_CTV.
  ?node_CTX <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://platform1.eu/sensors#PositionObservation>.
    ?node_CTX <http://purl.oclc.org/NET/ssnx/ssn#observationResult>
?node_CTY.
    ?node_CTX <http://purl.oclc.org/NET/ssnx/ssn#observationResultTime>
?node_CTZ.
    ?node_CTX <http://purl.oclc.org/NET/ssnx/ssn#featureOfInterest>
?node_CTW.
    ?node_CTY <http://purl.oclc.org/NET/ssnx/ssn#isProducedBy> ?node_CTS.
    ?node_CTY <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://platform1.eu/sensors#PositionSensorOutput>.
    ?node_CTY <http://purl.oclc.org/NET/ssnx/ssn#hasValue> ?node_CTU.
}
INSERT {
      _:b0 <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://inter-
iot.eu/central#Position>.

    ?node_CTX <http://www.w3.org/ns/sosa/resultTime> ?node_CTZ.

    ?node_CTX <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://www.w3.org/ns/sosa/Observation>.
    ?node_CTX <http://www.w3.org/ns/sosa/observedProperty> _:b0.
    ?node_CTX <http://www.w3.org/ns/sosa/hasFeatureOfInterest> ?node_CTW.
     ?node_CTX <http://www.w3.org/ns/sosa/hasResult> ?node_CTU.
    ?node_CTX <http://www.w3.org/ns/sosa/madeBySensor> ?node_CTS.
}
WHERE {
      ?node_CTX <http://purl.oclc.org/NET/ssnx/ssn#observedBy> ?node_CTV.
    ?node_CTX <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://platform1.eu/sensors#PositionObservation>.
    ?node_CTX <http://purl.oclc.org/NET/ssnx/ssn#observationResult>
?node_CTY.
     ?node_CTX <http://purl.oclc.org/NET/ssnx/ssn#observationResultTime>
?node_CTZ.
    ?node_CTX <http://purl.oclc.org/NET/ssnx/ssn#featureOfInterest>
?node_CTW.
    ?node_CTY <http://purl.oclc.org/NET/ssnx/ssn#isProducedBy> ?node_CTS.
    ?node_CTY <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://platform1.eu/sensors#PositionSensorOutput>.
    ?node_CTY <http://purl.oclc.org/NET/ssnx/ssn#hasValue> ?node_CTU.
}
```

8.4 Related training guides and courses

In order to support the various players of interoperability, other courses have been defined in WP5 to enable these players to build up skills on AIoTES and to master the fundamentals. Courses are focused in AIoTES rather than only on the SIL, which is a core part of AIoTES that enables interoperability. These courses will be available in the short-term future in an educational platform (further information will be provided in D5.2.2). The list of courses related with skills on ACTIVAGE interoperability are listed in Table 20. Also, the list of users and their recommended training is shown in Table 19.

Table 19: Training courses for stakeholders with interoperability interests

Stakeholder	Proposed training
Application developer	General knowledge about AIoTES Use of AIoTES API AIoTES development tools Application troubleshooting Privacy and security aspects
AIoTES expert	General knowledge about AIoTES AIoTES deployment Integration of IoT platforms in AIoTES User management Privacy and security aspects
AIoTES developer	General knowledge about AIoTES Integration of IoT platforms in AIoTES AIoTES troubleshooting Privacy and security aspects
DS administrator	General knowledge about AIoTES Use of AIoTES management tools Knowledge about IoT devices management DS management DS troubleshooting Privacy and security aspects
DS device installer	General knowledge about AIoTES Use of AIoTES management tools Device troubleshooting Privacy and security aspects
open call participant	General knowledge about AIoTES Background and expertise in the AHA & IoT domains Privacy and security aspects
other third parties involved in development	General knowledge about AIoTES Background and expertise in the AHA & IoT domains AIoTES development tools Privacy and security aspects

A short description of each course is given in the Table 20.

Table 20. Brief description of the different training courses related to AIoTES

Course title	Brief description
General knowledge about AIoTES	Introductory and general course on global architecture of the ACTIVAGE IoT Ecosystem Suite (AIoTES)
Use of AIoTES API	Specific course onto the usage of the different APIs provided by AIoTES for exchange data over DS and also to develop new applications
AIoTES development tools	Courses addressing developers at multiple degrees of experience relating to the different developments tools for AIoTES : IoT platforms tools, ontology designer, etc
Privacy and security aspects	Basic overview of security and privacy state of art and knowledge of the detailed architecture of the different security and privacy blocks specified and integrated within the AIoTES framework
AIoTES deployment	Functions for discover both IoT services and applications registered by site administrators and application developers, in order to facilitate new IoT applications on deployment sites
Integration of IoT platforms in AIoTES	Mastering the techniques for developing new bridges and alignments for the inclusion of a new platform in AIoTES
User management	Management for user such as creating credential, provisioning the user authentication artefacts, managing the roles and permissions of user, managing the revocation of users
Use of AIoTES management tools	Presentation of the different management tools associated to AIoTES and their operation.
Knowledge about IoT devices management	In order to ensure the proper integration of devices, basic knowledge about topology of devices and their capabilities.
DS management	Course onto the DS management to handle the lifecycle of the different unit and component within the DS. Operation includes creating a DS Unit, managing the different instance of ioT platforms, managing the service deployment, etc.
DS troubleshooting	Use of the AIoTES support platform in order to report and manage issues related to the DS management operations
AIoTES troubleshooting	Use of the AIoTES support platform in order to report and manage issues related to the AIoTES components
Device troubleshooting	Use of the AIoTES support platform in order to report and manage issues related to the IoT devices deployed within the DS Unit
Application troubleshooting	Use of the AIoTES support platform in order to report and manage issues related to the AIoTES APIs
Background and expertise in the AHA & IoT domains	State of art regarding areas covered by Activage : Active and Healthy Ageing (AHA) and the Internet of Things (IoT).

9 Conclusions & Future Work

9.1 Conclusions

This deliverable continues the work presented in deliverable 3.2 “Interoperability layer architecture”, by going a step further in the definition of components, tools and techniques employed within ACTIVAGE project to facilitate interoperability through the implementation of the Semantic Interoperability Layer (SIL).

Chapter 3 dives into the rationale of why interoperability is required and analyses the different scenarios of interoperability identified in ACTIVAGE, focusing on the semantic interoperability that is the core of the SIL.

A solution for achieving semantic interoperability is the creation of a common data model on top of all the IoT platforms in the AIoTES ecosystem, which derives into a service ontology for services based on AHA. Also the use of ontology-to-ontology translations through the IPSM component of the SIL.

Some preliminary interoperability scenarios have already been achieved using the Inter Middleware coming from INTER-IoT (which represents the SIL Interoperability Layer), implementing bridges between the middleware and specific IoT platforms. The advances in the implementation of the SIL include the integration of the Inter Platform Semantic Mediator (IPSM), the Universal Semantic Translation thorough a common modular central ontology GOIoTP.

Main efforts have been done on the integration of the IPSM into the SIL, what allows the establishment and configuration of broker communication channels to enable the transmission of data to/from this semantic translator, and the translation among one platform ontology and GOIoTP.

Some considerations regarding security and privacy are taken into account. What regards to security and privacy, the tools that the SIL will provide have been identified and described in detail.

Finally, to allow the growth of the developers’ ecosystem and enable the creation of new services on top of AIoTES, a “developer and user” guide has been prepared to guide future developers into the creation of new platform bridges and, and some specific formative events have been sketched in accordance to the training courses already mentioned in Deliverable 5.2.

9.2 Future Work

This deliverable describes the status of the work until month 18 for tasks T3.3 (Building bridges to platforms and protocols) and T3.4 (Implementing the SIL). A great deal of work has already been done, but these tasks will continue in the incoming months. The actions to be performed will be a continuation and upgrade of the work already completed:

- Data model: the presented ‘interoperability data model’ is a first iteration that will be enriched with requirements coming from the interoperability needs in AIoTES and the DS, as interoperability scenarios are implemented. Further information regarding the data model, as well as implementation details, will be provided in D3.11.

- Security and privacy have been analysed and some requirements have been extracted. The data model still requires the corresponding ontology to identify anonymized data and data consent management.

- Regarding the interoperability framework that the SIL represents, further information will be presented in the next version of this deliverable (D3.11).

- In this deliverable, there are training materials to show how SIL is installed and how bridges and semantic alignments can be developed and integrated into the SIL. These materials, currently addressing internal project developers, will be aligned and complemented with AIoTES training materials that will be provided in the frame of WP5 (further information will be provided in D.5.2.2).

All these future works will be properly reported on D3.11 on month 30.

References

1. Jabbar S, Ullah F, Khalid S, Khan M, Han K. Semantic Interoperability in Heterogeneous IoT Infrastructure for Healthcare. Wirel Commun Mob Comput. 2017;2017:1-10.

2. ISO 14258:1998 (Industrial automation systems -- Concepts and rules for enterprise models).

3. Healthcare Information and Management Systems Society. Definition of Interoperability. Himss. 2013;2013.

4. Krco S, Pokric B, Carrez F. Designing IoT architecture(s): A European perspective. En IEEE; 2014 [citado 2 de julio de 2018]. p. 79-84. Disponible en: http://ieeexplore.ieee.org/document/6803124/

5. Ganzha M, Paprzycki M, Pawłowski W, Szmeja P, Wasielewska K. Semantic interoperability in the Internet of Things: An overview from the INTER-IoT perspective. J Netw Comput Appl. marzo de 2017;81:111-24.

6. Papazoglou MP. Service-oriented computing: concepts, characteristics and directions. En IEEE Comput. Soc; 2003 [citado 2 de julio de 2018]. p. 3-12. Disponible en: http://ieeexplore.ieee.org/document/1254461/

7. Alonso G, Casati F, Kuno H, Machiraju V. Web Services. En: Web Services [Internet]. Berlin, Heidelberg: Springer Berlin Heidelberg; 2004 [citado 2 de julio de 2018]. p. 123-49. Disponible en: http://link.springer.com/10.1007/978-3-662-10876-5_5

8. Pautasso C, Zimmermann O, Leymann F. Restful web services vs. «big»' web services: making the right architectural decision. En ACM Press; 2008 [citado 2 de julio de 2018]. p. 805. Disponible en: http://portal.acm.org/citation.cfm?doid=1367497.1367606

9. Vinoski S. CORBA: integrating diverse applications within distributed heterogeneous environments. IEEE Commun Mag. febrero de 1997;35(2):46-55.

10. Ganzha M, Paprzycki M, Pawłowski W, Szmeja P, Wasielewska K. Towards Semantic Interoperability Between Internet of Things Platforms. En: Gravina R, Palau CE, Manso M, Liotta A, Fortino G, editores. Integration, Interconnection, and Interoperability of IoT Systems [Internet]. Cham: Springer International Publishing; 2018 [citado 2 de julio de 2018]. p. 103-27. Disponible en: http://link.springer.com/10.1007/978-3-319-61300-0_6

11. Yacchirema, D., Gonzalez-Usach, R., Palau, C. EM. IoT Platform Interoperability applied to the domain of transport and logistics. En.

12. Generic Ontology for IoT Platforms [Internet]. [citado 2 de julio de 2018]. Disponible en: https://docs.inter-iot.eu/ontology/

13. Open Travel Alliance [Internet]. Disponible en: http://www.opentravel.org

14. OpenEHR. Disponible en: http://www.openehr.org/what_is_openehr

15. Hachem S, Teixeira T, Issarny V. Ontologies for the internet of things. En ACM Press; 2011 [citado 2 de julio de 2018]. p. 1-6. Disponible en: http://dl.acm.org/citation.cfm?doid=2093190.2093193

16. , L. den Hartog F& RJD. Study on semantic assets for smart appliances interoperability. {D}-{S}4: {FINAL} {REPORT}. 2015.

17. UPnP. UPnP device Architecture 1.1. 2008.

18. Compton M, Barnaghi P, Bermudez L, García-Castro R, Corcho O, Cox S, et al. The SSN ontology of the W3C semantic sensor network incubator group. Web Semant Sci Serv Agents World Wide Web. diciembre de 2012;17:25-32.

19. M. Leggieri, C. von der Weth, and M. Serrano. Deliverable 3.1.2: Semantic Representations of Internet-Connected Objects. 2013.

20. FIESTA-IoT. Deliverable 2.4: FIESTA-IoT Meta Cloud Architecture. 2015.

21. IoT-A. IoT-A Terminology.

22. Daniele L, Hartog F ran. den, Roes J. Study on Semantic Assets for Smart Appliances Interoperability. 2015.

23. V. Haren. TOGAF Verson 9.0. 2009.

24. Sahay R, Fox R, Zimmermann A, Polleres A, Hauswrith M. A Methodological Approach for Ontologising and Aligning Health Level Seven (HL7) Applications. En Springer, Berlin, Heidelberg; 2011. p. 102–117. Disponible en: http://link.springer.com/10.1007/978-3-642-23300-5_9

25. Iqbal AM. An OWL-DL Ontology for the HL7 Reference Information Model. En Springer, Berlin, Heidelberg; 2011. p. 168–175. Disponible en: http://link.springer.com/10.1007/978-3-642-21535-3_22

26. FHIR v3.0.1 [Internet]. [citado 18 de julio de 2018]. Disponible en: https://www.hl7.org/fhir/overview.html

27. HL7 Standards Product Brief - HL7 Version 3 Standard: Security and Privacy Ontology, Release 1 [Internet]. [citado 18 de julio de 2018]. Disponible en: http://www.hl7.org/implement/standards/product_brief.cfm?product_id=348

28. Dictionary by Merriam-Webster: America's most-trusted online dictionary [Internet]. [citado 18 de julio de 2018]. Disponible en: https://www.merriam-webster.com/

29. Elluri L, Joshi KP. A Knowledge Representation of Cloud Data controls for EU GDPR Compliance. 2018;

30. GDPR Ontology Representation Using Semantic Web Technologies for assisting in GDPR compliance. :2106.

31. Heath T, Bizer C. Linked data: evolving the web into a global data space. 1. ed. San Rafael, Calif.: Morgan & Claypool; 2011. 122 p. (Synthesis lectures on the semantic web: theory and technology).

32. Heitmann B, Kinsella S, Hayes C, Decker S. Implementing Semantic Web applications: Reference architecture and challenges. CEUR Workshop Proc. 2009;524:16–30.

33. Sporny M, Kellogg G, Lanthaler M. Json-Ld 1.0. JSON Based Ser Linked Data. 2013;(January):1–33.

34. W3C. RDF 1.1 XML Syntax [Internet]. [citado 30 de julio de 2018]. Disponible en: https://www.w3.org/TR/rdf-syntax-grammar/

35. Herman I, Adida B, Sporny M, Birbeck M. RDFa 1.1 Primer - Second Edition – Rich Structured Data Markup for Web Documents (W3C Working Group Note 07 June 2012). 2013;(August):1–37.

36. W3C. SPARQL 1.1 Query Language [Internet]. Disponible en: https://www.w3.org/TR/sparql11-query/

37. Davies J, Studer R, Warren P. Semantic Web technologies: trends and research in ontology-based systems. Chichester, England ; Hoboken, NJ: John Wiley & Sons; 2006. 312 p.

38. Noy NF, McGuinness DL. Ontology Development 101: A Guide to Creating Your First Ontology. Stanf Knowl Syst Lab. 2001;25.

Appendix A UML conventions

A.1 Simplified UML conventions for use cases descriptions

A.1.1 GENERAL VIEW for APPS on IoT platform in ACTIVAGE DS

One/multiple existing App/s running on top of an ACTIVAGE IoT platform in a DS with no AIoTES

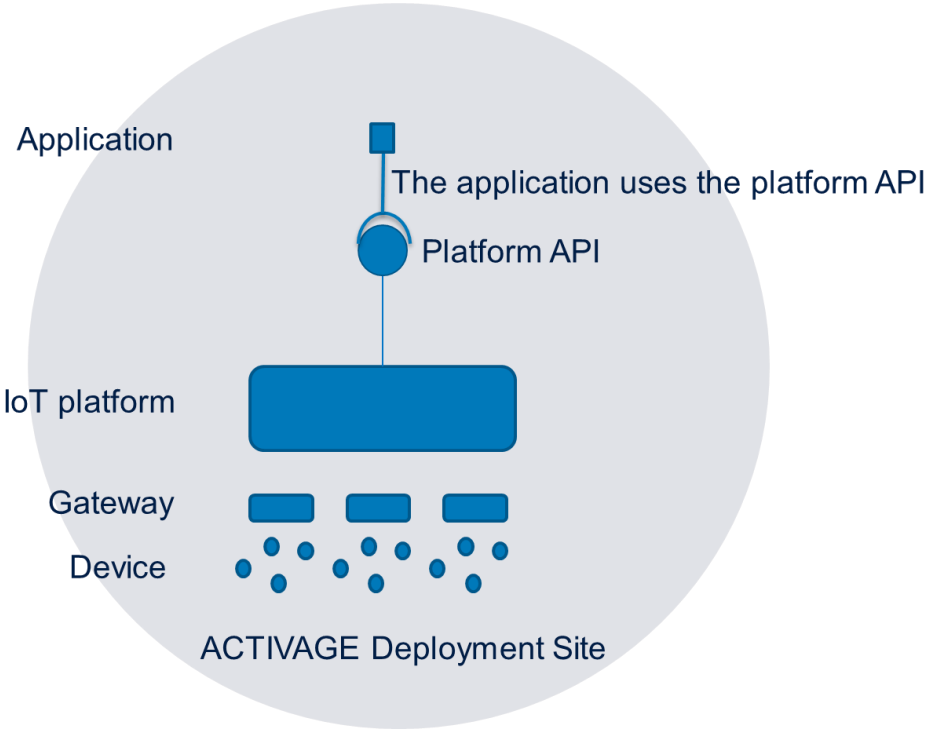

Figure 31: Simplified UML for use case descriptions

A.2 Semantic Concepts

A.2.1 Linked Data

The term 'Linked Data' is usually applied to a set of techniques for publishing and interlinking structured data on the Web. Recently with the extension of the Web into a physical Web for IoT the use of Linked Data has been increasing expectations for data exchange and thus different kinds of data sources are expected to be seamlessly shared (see [31]). Linked Data is based on four principles, [31] and [32] briefly summarized as follow:

The first principle is to **use URIs as names for things**. After the identification of items in a domain of interest, they are described in the data set with their properties and relationships. Each thing must be assigned a globally unique name, usually an HTTP URI (since this makes it easy to enforce global uniqueness).

The second principle is to **use HTTP URIs**, so clients can retrieve a description of the names thing or resource using the HTTP protocol. For humans the description can be provided as HTML, for machines it can be provided as RDF triples.

The third principle is to **use standards to provide information**. This usually means that standards like RDF or SPARQL are used to model and access data. We explain the most important standards later in this chapter.

The last principle is to **link to other URIs** and to enable the possibility to discover more things. That means that there should be external links pointing to other data sources on the Web and thus a larger (distributed) data space can be explored automatically.

In a nutshell, Linked Data enables the implementation of generic applications operating over a huge, interconnected (distributed) data space (storage) by using Web standards and a common distributed decentralized data model (format or series of formats).

Linked Data philosophy has expanded and in the same form the way to implement it and currently it varies from basic data formats (SSC, XML) with tagging as methodology for representing the data to very structured and formal ones (RDF and OWL) with RDF serialization formats such as N-Triples and Turtle, having in between them large variations of data formats and representations.

This large number of variations and also the lack of common understanding about the best use of the linked data formats in the community of data and information management systems produces extra overhead in the form of a steeper learning curve when integrating new systems to consume linked data. To counter this, the ACTIVAGE project consortium decided to use standard formats based on the common serialization formats such as XML and JSON. Thus, the two remaining options are JSON-LD [33] and RDF/XML.

A.2.2 JSON-LD

JSON-LD was chosen as the data format that is widely used for all Linked Data items in IoT-enabled deployed platforms amongst the Deployment Sites in the ACTIVAGE ecosystem. JSON-LD is a JSON-based serialisation for Linked Data with the following design goals:

- **Simplicity**: There is no need for extra processors or software libraries, just the knowledge of some basic keywords.
- **Compatibility**: JSON-LD documents are always valid JSON documents, so the standard libraries from JSON can be used.
- **Expressiveness**: Real-world data models can be expressed because the syntax serialises a directed graph.
- **Terseness**: The syntax is readable for humans and developers need little effort to use it.
- **Zero Edits**: Most of the time JSON-LD can be devolved easily from JSON-based systems.
- **Usable as RDF**: JSON-LD can be mapped to / from RDF and can be used as RDF without having any knowledge about RDF.

JSON-LD's terseness and simplicity (from the above) are the main desirable conditions to motivate innovation in the ACTIVAGE ecosystem. JSON-LD also allows for referencing external files to provide context. This means contextual information can be requested on-demand and makes JSON-LD better suited to situations with high response times or low bandwidth usage requirements.

The use of JSON-LD in ACTIVAGE deployment sites and the AIoTES framework will reduce the complexity of ACTIVAGE development by (1) making it possible to reuse a large number of existing tools and (2) reduce the inherent complexity of RDF documents. Ultimately, this will increase ACTIVAGE's uptake and success. In the following we give a short overview of the main JSON-LD features and concepts. More information can be found in [33] and in the JSON-

LD syntax specification document available at: <https://json-ld.org/spec/latest/json-ld/>. The data model underlying JSON-LD is a labelled, directed graph. There are a few important keywords, such as **@context**, **@id**, **@value**, and **@type**. These keywords are the core part of JSON-LD. Four basic concepts should be considered:

- **Context**: A context in JSON-LD allows using shortcut terms to make the JSON-LD file shorter and easier to read (as well as increasing its resemblance with pure JSON). The context maps terms to IRIs. A context can also be externalised and reused for multiple JSON-LD files by referencing its URI.
- **IRIs**: Internationalised Resource Identifiers (IRIs) are used to identify nodes and properties in Linked Data. In JSON-LD two kinds of IRIs are used: absolute IRIs and relative IRIs. JSON-LD also allows defining a common prefix for relative IRIs using the keyword **@vocab**.
- **Node Identifiers**: Node identifiers (using the keyword @id) reference nodes externally. As a result of using **@id**, any RDF triples produced for this node would use the given IRI as their subject. If an application follows this IRI it should be able to find some more information about the node. If no node identifier is specified, the RDF mapping will use blank nodes.
- **Specifying the Type**: It is possible to specify the type of a distinct node with the keyword **@type**. When mapping to RDF, this creates a new triple with the node as the subject, a property **rdf:type** and the given type as the object (given as an IRI).

A.2.3 JSON-LD Framing

The extension of the web into IoT applications has make possible for the developers to think on using JSON-LD for their information systems design and thus restructuring JSON-LD data before the application processes it to leads to simpler code when processing data from external sources and when this needs to be integrated with web services. JSON-LD Framing allows developers to structure data retrieved from the Web according to the specific needs of their application and at the same time perform query by example and force a specific tree layout to a JSON-LD document. JSON-LD framing additional information alike the standard description and best practices can be found at the online specification: <https://json-ld.org/spec/latest/json-ld-framing/#framing> To illustrate a JSON-LD resulting description, we give a short example for a sensor description (in JSON-LD format) using pre-defined types in OpenIoT ontology as shown in Figure 32: Example Sensor Description using json-LD notation

```
{
  "@context": "http://example.org/contexts/sensor.jsonld",

  "name": "TemperatureSensor No.123",
  "type": "ActivageSensor",
  "description": "This is an example sensor",
  "uri": "http://www.example.com/activage/sensor/123",
  "hasLastKnownLocation": {
    "type": "geo:Point",
    "geo:lat": "53.2719",
    "geo:long": "-9.0489"
  },
  "ssn:observes": [
    {
      "type": "http://lsm.deri.ie/OpenIoT/Light",
      "uri": "http://www.example.com/activage/sensor/123/light"
    },
    {
      "type": "http://lsm.deri.ie/OpenIoT/Temperature",
      "uri": "http://www.example.com/vital/sensor/123/temperature"
    }
  ],
  "ssn:madeObservation":
    "http://www.example.com/activage/sensor/123/obsvn/1"
}
```

.

Figure 32: Example Sensor Description using json-LD notation

JSON-LD is simply JSON using Linked Data philosophy and with set of linked data conventions and associated defined tools. Therefore, it can be stored in a NOSQL database (e.g MongoDB) and queried accordingly. Having said this, if you want to do things like graph traversals/visualisation, you could still store it in Mongo and use a tool like Google Cayley to query/visualise relationships. JSON-LD data mostly can be imported in most RDF databases (triplestores) and then just query the data with SPARQL.

A.2.4 RDF

The most common data model used in the context of Linked Data is the Resource Description Framework (RDF) [34]. RDF is a popular standard for describing things (known as resources or entities). By itself it is a graph-based data model that represents information as labelled directed graphs. This graph is built of triples that describe the data in the form of subject, predicate(s) and object. Each triple **(s, p, o)** consists of a subject **s**, a predicate **p** and an object **o**. Take for example the information "The sensor has the temperature value of 25 degrees celsius". This would be modelled as a triple with "sensors" as the subject, "has the temperature value" as the predicate and "25 Degree Celsius" as the object. Note that the predicate in the middle always denotes the relationship between subject and object. Both the subject and the predicate are identified by URIs (assigning them globally unique identifiers)[21] while the object can be a URI or a literal value (i.e. a string or a number). The triple in our example could e.g. look like this (given in N-Triple notation) is in Figure 32.

```
<http://example.com/sensor> <http://example.com/hasTemperatureValue> "25 °C"
```

Figure 33: Example Sensor Description using RDF N-Triple notation

Using RDF in a Linked Data context has some advantages. In the following we briefly outline the most important ones of these advantages. The reader is referred to [31] for more detailed information. Possible advantages are:

- If the identifiers of data items (both used as subjects and objects) and vocabulary terms (used as predicates) are HTTP URIs, the RDF data model can be used at global scale and anybody is able to refer to anything.
- Each RDF triple is included in the Web of Data and can be the starting point for explorations in the data space, because any URI can be looked up in an RDF graph over the Web.
- It is possible to set RDF links between data from different sources.
- Sets of triples can be merged in a single graph to combine information.
- Terms taken from different vocabularies can be mixed in a RDF graph.

As RDF is just a data model and not a data format, there are a number of data formats that can be used to write RDF data, either directly as triples or as nodes that can be mapped to RDF triples, e.g. RDF/XML [34], RDFa [35], Turtle, N-Triples and JSON-LD. In the VITAL Project JSON-LD is used.

Linked Data can be used in cases where data originates from different sources. To integrate all data, be it from one or different sources, there have to be some rules. Some rules determine how the RDF graph is to be built and how triples may be connected or not. These rules are given by ontologies as it is explained in the corresponding section in this document i.e. OWL.

[21] A subject can also be identified by a so-called blank node. A blank node is a local identifier.

A.2.5 SPARQL

Assuming that there is RDF data, then a developer needs a language to query it. SPARQL (SPARQL Protocol and RDF Query Language) is able to query single RDF graphs and it is used to retrieve and manipulate data, which is stored in RDF. There are four query variations that SPARQL can distinguish: **SELECT**, **CONSTRUCT**, **ASK** and **DESCRIBE** queries. The most basic construct of a SPARQL query are graph patterns, explained in [36]. A basic graph pattern is similar to a RDF triple except that the subject and predicate can be variables as well. The example in Figure 34 shows an example of a sensor description. We provide few more examples of SPARQL queries that can be run on the data that satisfies the OpenIoT Ontology. The Figure 33 provides an example about a query that provides all sensors at a given location at a given time.

```
Prefix ssn: <http://purl.oclc.org/NET/ssnx/ssn#>
Prefix openiot: <http://lsm.deri.ie/OpenIoT.owl#>
Prefix dul: <http://www.loa.istc.cnr.it/ontologies/DUL.owl#>
Prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
Prefix time: <http://www.w3.org/2006/time#>
Prefix xsd: <http://www.w3.org/2001/XMLSchema#>
select ?s ?val
where {
    ?o a ssn:Observation.
    ?o ssn:observedBy ?s.
        ?o ssn:observedProperty openiot:Sound.
    ?o ssn:observationSamplingTime ?t.
    ?o geo:location ?point.
    ?point geo:lat "48.8393406"^^xsd:double.
    ?point geo:long "2.3931164"^^ xsd:double.
    ?t time:inXSDDateTime "2016-08-08T20:00:09.016Z"^^xsd:dateTime.
    ?o ssn:observationResult ?or.
    ?or  ssn:hasValue ?v.
    ?v dul:hasDataValue ?val.
} group by ?s ?val
```

Figure 34: SPARQL Example Query for Sensors at a Given Location at a given time.

As mentioned, the Figure 33 provides an example about a query provides all sensors at a given location between a given time interval. This would return all those sensors (in the required time interval) that have been previously at the specified location and/or are at the given location.

```
Prefix ssn: <http://purl.oclc.org/NET/ssnx/ssn#>
Prefix openiot: <http://lsm.deri.ie/OpenIoT.owl#>
Prefix dul: <http://www.loa.istc.cnr.it/ontologies/DUL.owl#>
Prefix geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
Prefix time: <http://www.w3.org/2006/time#>
Prefix xsd: <http://www.w3.org/2001/XMLSchema#>
select ?s
where {
    ?o a ssn:Observation.
        ?o ssn:observedBy ?s.
    ?o ssn:observedProperty openiot:Sound.
    ?o ssn:observationSamplingTime ?t.
    ?o geo:location ?point.
    ?point geo:lat "48.8393406"^^xsd:double.
    ?point geo:long "2.3931164"^^xsd:double.
    ?t time:inXSDDateTime ?ti.
    FILTER (?ti >= "2016-08-07T20:00:09.016Z"^^xsd:dateTime>
            && ?ti < "2016-08-09T20:00:09.016Z"^^xsd:dateTime)
} group by ?s
```

Figure 35: SPARQL Example Sensor Description

A.2.6 OWL

In addition to RDF and SPARQL another very important technology in Linked Data are ontologies. An ontology specifies formally the conceptualisation of a domain of interest. As the conceptualisation is formal, a language is defined to be used and express all the relations and concepts around the domain, also a computer can automatically reason on it using those relations and concepts within the domain. There are practical ontologies for different domains of interest. An ontology consists of concepts (also referred to as classes), relations (also called properties), instances and axioms. It defines basic terms and relationships.

To specify ontologies, the W3C published the Web Ontology Language (OWL[22]), which builds on RDF. OWL facilitates mechanism for creating concepts, instances, relations and axioms. Concepts can have super and sub concepts. Axioms provide information about classes and properties. This topic is explained in detail by [37] and [38].

Reuse of ontologies is crucial. If some new data models arise they should be attached to an existing ontology. This ontology grows by doing so and helps users on any of its nodes to reach every other node in this ontology graph. The user will get much more information as just of his own model – if he/she wants to.

Table 21 provides an overview of the main ontologies identified as relevant for ACTIVAGE and classified according to the 4 main domains involved in the Deployment Sites, their namespaces and the prefixes that we use to refer to them are included as reference.

[22] www.w3.org/2004/OWL

Table 21: ACTIVAGE Relevant Ontology / Language Prefixes

Prefix	Ontology / Language	Namespace
dcn	Delivery Context ontology	http://www.w3.org/2007/uwa/context/deliveryContext.owl#
dul	DOLCE+DnS Ultralite ontology	http://www.ontologydesignpatterns.org/ont/dul/DUL.owl#
foaf	Friend of a Friend	http://xmlns.com/foaf/
geo	Basic Geo (WGS84) ontology	http://www.w3.org/2003/01/geo/wgs84_pos#
hrest	hRESTS ontology	http://www.wsmo.org/ns/hrests#
otn	Ontology of Transportation Networks	http://www.pms.ifi.lmu.de/rewerse-wga1/otn/OTN.owl
oneM2M	oneM2M base Ontology	http://www.onem2m.org/ontology/Base_Ontology#
owl	Web Ontology Language	http://www.w3.org/2002/07/owl#
qudt	Quantities, Units, Dimensions and Data Types Ontologies	http://qudt.org/schema/qudt#
rdfs	RDF Schema ontology	http://www.w3.org/2000/01/rdf-schema#
Prefix	Ontology / Language	Namespace
s4ac	Social Semantic SPARQL Security for Access Control	http://ns.inria.fr/s4ac/v2#
ssn	Semantic Sensor Network ontology	http://purl.oclc.org/NET/ssnx/ssn#
sawsdl	Semantic Annotations for WSDL and XML Schema ontology	http://www.w3.org/ns/sawsdl#
saref	Smart Appliances REFerence Ontology	http://ontology.tno.nl/saref.ttl
time	OWL Time ontology	http://www.w3.org/2006/time#
wsl	WSMO-Lite ontology	http://www.wsmo.org/ns/wsmo-lite#
xsd	XML Schema Definition	http://www.w3.org/2001/XMLSchema#

A.3 ACTIVAGE Overall Data Format Conventions

Table 22: ACTIVAGE other data format conventions

File Content	Extension	Common Use
MATLAB formatted data	MAT	MATLAB workspace
		Partial access of variables in MATLAB workspace
Text	Any, including: CSV TXT	Comma delimited numbers
		Delimited numbers
		Delimited numbers, or a mix of text and numbers
		Column-oriented delimited numbers or a mix of text and numbers
Spreadsheet	XLS XLSX XLSM	Worksheet or range of spreadsheet
	XLSB (Systems with Microsoft® Excel® for Windows® only) XLTM (import only) XLTX (import only) ODS (Systems with Microsoft Excel for Windows only)	Column-oriented data in worksheet or range of spreadsheet
Extensible Markup Language	XML	XML-formatted text
Data Acquisition Toolbox™ file	DAQ	Data Acquisition Toolbox
Scientific data	CDF	Common Data Format
	FITS	Flexible Image Transport System
	HDF	Hierarchical Data Format, version 4, or HDF-EOS v. 2
	H5	HDF or HDF-EOS, version 5

File Content	Extension	Common Use
	NC	Network Common Data Form (netCDF)
Image	BMP	Windows Bitmap
	GIF	Graphics Interchange Format
	HDF	Hierarchical Data Format
	JPEG JPG	Joint Photographic Experts Group
	JP2 JPF JPX J2C J2K	JPEG 2000
	PBM	Portable Bitmap
	PCX	Paintbrush
	PGM	Portable Graymap
	PNG	Portable Network Graphics
	PNM	Portable Any Map
	PPM	Portable Pixmap
	RAS	Sun™ Raster
	TIFF TIF	Tagged Image File Format
	XWD	X Window Dump
	CUR	Windows Cursor resources
	ICO	Windows Icon resources
Audio (all platforms)	AU SND	NeXT/Sun sound
	AIFF	Audio Interchange File Format
	AIFC	Audio Interchange File Format, with compression codecs
	FLAC	Free Lossless Audio Codec

File Content	Extension	Common Use
	OGG	Ogg Vorbis
	WAV	Microsoft WAVE sound
Audio (Windows)	M4A MP4	MPEG-4
	any	Formats supported by Microsoft Media Foundation
Audio (Mac)	M4A MP4	MPEG-4
Audio (Linux®)	any	Formats supported by GStreamer
Video (all platforms)	AVI	Audio Video Interleave
	MJ2	Motion JPEG 2000
Video (Windows)	MPG	MPEG-1
	ASF ASX WMV	Windows Media®
	any	Formats supported by Microsoft DirectShow®
Video (Windows 7 or later)	MP4 M4V	MPEG-4
	MOV	QuickTime
	any	Formats supported by Microsoft Media Foundation
Video (Mac)	MP4 M4V	MPEG-4
	MPG	MPEG-1
	MOV	QuickTime
	any	Formats supported by QuickTime, including .3gp, .3g2, and .dv
Video (Linux)	any	Formats supported by your installed GStreamer plug-ins, including .ogg