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

D3.4 Bridges to the IoT protocols and platforms State of the art and planned implementations

Deliverable No.	D3.4		
WorkPackage No.	WP3	**WorkPackage Title**	ACTIVAGE Secure Interoperability Layer
Version	2.0		
Status	Release		
Date of issue	October 2017		

S. Document

0.1 Authors

Author	Company	e-mail
Stéphane Bergeon	CEA	Stephane.BERGEON@cea.fr
Levent Gurgen	CEA	Levent.GURGEN@cea.fr
Byron Ortiz	TELEVES	byrort@televes.com
Andrea Carboni	CNR	andrea.carboni@isti.cnr.it
Michele Girolami	CNR	michele.girolami@isti.cnr.it
Nikolaos Kaklanis	CERTH	nkak@iti.gr
Iosif Hamlatzis	CERTH	hamlatzis@iti.gr
Konstantinos Votis	CERTH	kvotis@iti.gr
Dimitrios Tzovaras	CERTH	dimitrios.tzovaras@iti.gr
Aqeel Kazmi	NUIG	aqeel.kazmi@insight-centre.org
Philippe Dallemagne	CSEM	philippe.dallemagne@csem.ch
Pierre Barralon	TECNALIA	pierre.barralon@tecnalia.com
Mary Panou	CERTH	mpanou@certh.gr
Saied Tazari	Fh-IGD	saied.tazari@igd.fraunhofer.de

0.2 Document History

Date	Version	Editor	Change	Status
April 2017	0.0	S. Bergeon	Initial ToC proposal	draft
April 2017	0.1	S. Bergeon	Status of IoT protocol (both service and edge) bridges not only edge ones	draft
May 2017	0.4	S.Bergeon	Added section Platform – interoperability layer bridges Removed HOMARD platform Added Finnish platform (name to be precised)	draft
	0.5	S.Bergeon	Removed 4.1 and 6.1	draft

			Added contributors	
	0.6	Byron Ortiz	Added table in 3.1.2 SOFIA2 section	draft
June 2017	0.7	S.Bergeon	Added 3.1.8 IoTivity section and renamed 3.1.7 Seniorsome Finish platform	draft
July 2017	0.8	A.Carboni – M.Girolami (CNR PISA) S.Bergeon	Added content in section 3.1.3 and references Added and updated tables from templates	draft
	0.9	S.Bergeon	Added 6.1 coverture discussion	draft
	0.10	N.Kaklanis, I. Hamlatzis, K. Votis, D. Tzovaras	Added content for IoTivity mainly in 2.1.8 as well as in the tables of all sections related to IoTivity	draft
September 2017	0.12	A.Kazmi	Added content in 2.1.4 OpenIoT section	draft
October 2017	0.18	S.Bergeon	Final polishing	Peer review
	0.19	P.Barralon, S.Bergeon	English polishing Removed not relevant DS4 recommendation in section 6 added planning information in conclusion	Peer review
	0.20	M.Panou, S.Bergeon	Corrected conclusion Added recommendation overview table 22 in 6.3 Remains opened comments in 3.1.3 Remains empty section 5	Peer review
	0.23	M.Girolami P.Barralon A.Medrano S.Bergeon	Added first proposition for section 5 (SB) Final polishing	Peer review
	1.0	S. Tazari S.Bergeon	Formal polishing	Version for review
	2.0	S.Bergeon	7 ACTIVAGE platforms	

0.3 Document KeyData

Keywords	Internet of Things, UniversAAL, FIWARE, Sofia2, OpenIoT, sensiNact, IoTivity, SENIORSome, OneM2M, IoT platform, IoT protocol, middleware, interoperability, bridge
Editor Address data	Name: Stéphane Bergeon Partner: CEA

0.4 Abstract

This deliverable contains a report on the initial ACTIVAGE Internet of Things (IoT) platforms. It begins with a summary of selected ACTIVAGE IoT platforms and edge protocols that are used in ACTIVAGE deployment sites. The second part presents the southbound bridges as they are implemented by each ACTIVAGE IoT platform. This section also presents planned southbound implementations for each platform. The third section shows the existing bridges among platforms when exist, and what are the planned ones. A fourth section introduces future works for the implementation of bridges between the seven ACTIVAGE IoT platforms and the ACTIVAGE interoperability layer. The Final conclusion makes a synthesis of needed/existing/planned/missing IoT protocols bridges at deployment site level, and proposes solutions using the ACTIVAGE interoperability feature to fulfil IoT device deployment requirements.

0.5 Scope

This document presents a review of the available bridges included in the 8 selected IoT platforms considered within the ACTIVAGE project:

- bridges from/to sensor and actuator devices protocols
- bridges between IoT platforms

A planning of the upcoming bridges implementations is also presented. A next step will consist of prioritizing such developments.
The final goal is to implement the ACTIVAGE interoperability layer integrating the bridge features from ACTIVAGE IoT platforms.

Table of contents

List of tables

List of figures

1 Introduction

Deliverable D3.1 proposed an overall view of the IoT platform market in Europe by presenting the current state-of-the-art of IoT platforms and a selection of those to be used in ACTIVAGE. The objective is to provide a secure and flexible interoperability layer capable of offering access to all the services (of the seven platforms) from a single entry point. For the purposes of the ACTIVAGE project, the following seven open-source application-oriented IoT platforms were selected:

- universAAL IoT [1]
- Sofia2 [2]
- OpenIoT [3]
- sensiNact [4]
- FIWARE [5]
- IoTivity [6]
- SENIORSome [7]

For each of the aforementioned platforms, the main conceptual characteristics will be presented, along with a description of the communication protocols and underlying data models. Such data models are used to transform raw data collected by the devices to meaningful (semantic) information that can be used in the application logic. Based on this identification, the reusable assets from each open-source European platform have been identified.

This report, by making the state of the art on bridges implementation among the ACTIVAGE IoT platforms, aims to complement and precise the assets that will be used during the implementation of the ACTIVAGE components of WP3-5.

2 Definition of IoT platform bridges

A **bridge**, in the IoT domain, is a software component that can read and write data necessary to communicate with a dedicated IoT platform. When reading, the IoT bridge unpacks the data from the received message, following a dedicated format as specified by the communication protocol. When writing, the bridge packs the data in the protocol telegram (also called payload).

We use the layer where the communication occurs to classify the IoT bridges:

- **Device layer** (or **Sensor layer**) **bridges** deal with communication towards and from the sensors and actuators.
- **Service layer** (or Middleware layer) **bridges** manage communication towards and from services (e.g. for applications, or for other IoT platforms). When a protocol is particularly devoted to communication between more than one IoT platform, we can precise **Interoperability layer**.

We also classify the bridges into five categories, depending of the source and destination edges of the communication:

- **Device to gateway** communication: device communicates directly to the gateway either through cable or radio with dedicated antenna (on-chip inside or with dongle plugged to the gateway). The bridge is hosted in the gateway among the IoT platform component, sending and receiving data to/from the sensors and actuators connected with the gateway.

  Examples of IoT protocol in this category: ZWave, EnOcean
- **Gateway to gateway** communication: the gateway communicates with another IoT platform (that could be hosted in another gateway, but could also be hosted in a cloud server). We use this dedicated naming for the protocol NGSI 9 & 10 v1 and v2.
- **Gateway to Wide Area Network (WAN)** communication: the gateway communicates with the rest of the world. The bridge is hosted in the gateway among the IoT platform components, sending and receiving data to/from the WAN, to communicate with other IoT platforms, or to provide top level services accessible from the application layer.

  Examples of IoT protocol in that category: RESTfull API, MQTT, CoAP
- **Device to Local Area Network (LAN)** communication: sensor and actuators are not directly connected to the gateway hosting the IoT platform. There is neither a direct cable connexion nor a direct Radio Frequency connexion with a dongle (or an antenna). The device is a node in the Local Area Network, as the gateway. Device and gateway communicate through Ethernet on this LAN. The IoT bridge hosted in the gateway manages the Ethernet communication with other nodes on the LAN.

  Examples of IoT protocol in that category: OpenHab, Philips hue
- **Device to WAN** communication: the sensors and actuators communicate to a third party IoT platform deployed in the cloud. The bridge (hosted either in the gateway or in the cloud) manages the Ethernet communication with the WAN node of the third party platform.

  Examples of IoT protocol in that category: LoRa, Sigfox

Sometimes the notion of southbound and northbound bridge is used. It has sense when a tree-like communication model is used with IoT platforms as nodes and sensors/actuators as leafs.

Southbound bridges (Figure 2) are bridges dedicated to feed the IoT platform. The IoT platform gathers data from sensors and dispatches commands to actuators through southbound bridges. Mainly southbound bridges are at the device layer, but by gathering data from other IoT platforms and services, southbound bridges are also in the service and interoperability layers.

Northbound bridges are dedicated to communicate with a higher level node in the communication tree: an aggregator IoT platform, the interoperability layer or applications.

3 ACTIVAGE IoT platforms and edge protocols

Focusing on the ACTIVAGE IoT platforms collection of bridges, and on the deployment sites requirements in terms of edge protocols, this section makes the mapping between the required and the available bridges. The first sub-section presents the available bridges of the selected IoT platforms, the second sub-section describes a first draft of DS requirements and the last sub-section shows the mapping between the requirements and available bridges.

3.1 Overview of supported IoT protocols among the ACTIVAGE IoT platforms

This sub-section presents the IoT protocol bridges (implemented or to be implemented) of each ACTIVAGE IoT platform.

3.1.1 FIWARE bridges status

FIWARE [5] is an open middleware platform for IoT, supported by the European Commission Union under the Future Internet Public Private Partnership Programme [9]. FIWARE provides public and royalty-free API specifications and interoperable protocols for the creation of new internet services and applications. Moreover, reference open-source implementations of its components are freely available. Table 1 below provides the IoT protocol bridges for FIWARE.

IoT Protocol bridges for FIWARE					
Protocol	**Domain**	**Layer**	**Connection edges**	**Status**	**Comment**
COAP	Generic	Sensor	Device to Gateway	To be implemented (done in a previous version)	RF
HTTP REST	Generic			Implemented	Abstract bridge; useful for further HTTP bridges
HTTP WSDL-SOAP	Generic	Service	Gateway to WAN (2way)	Implemented	
MQTT	Generic	Service	Gateway to WAN	Implemented	Northbound and southbound
NGSI 9 & 10 v1	Generic	Interoperability	Gateway to Gateway	Implemented	Related to FiWare
NGSI 9 & 10 v2	Generic	Interoperability	Gateway to Gateway	Implemented	v2 available release candidate

Protocol	Domain	Layer	Connection edges	Status	Comment
Sigfox	Generic	Sensor	Device to WAN	Implemented	RF
Zigbee	Generic	Service	Device to Gateway	To be implemented	RF
Z-Wave	Generic	Sensor	Device to Gateway	To be implemented	RF

Table 1: FIWARE supported and planned IoT protocol bridges

Table 2 below presents the bridges between FIWARE and the other ACTIVAGE IoT platforms.

IoT platform bridges for FIWARE						
	UNIVERSAAL	**SOFIA2**	**OPENIOT**	**sensiNact**	**SENIORSOME**	**IoTivity**
FIWARE	Trough InterIOT	Through InterIOT	Through InterIOT	available NGSI/HTTP	no	Through MQTT

Table 2: FIWARE supported and planned bridges with other ACTIVAGE IoT platforms

3.1.2 SOFIA2 bridges status

Sofia2 [10] is a semantics-based IoT platform, utilizing ontological models for the connection between the low-level device information and the applications. Table 3 below provides the IoT protocol bridges for SOFIA2.

IoT Protocol bridges for SOFIA2					
Protocol	**Domain**	**Layer**	**Connection edges**	**Status**	**Comment**
Z-Wave	Generic	device	Device to Gateway	Implemented	
KNX-RF	Generic	device	Device to Gateway	Implemented	
BLE	Generic	device	Device to Gateway	Implemented	
Zigbee	Generic	device	Device to Gateway	Implemented	

Table 3: SOFIA2 supported and planned IoT protocol bridges

Table 4 below presents the bridges between SOFIA2 and the other ACTIVAGE IoT platforms.

IoT platform bridges for SOFIA2						
	UNIVERSAAL	**OPENIOT**	**FIWARE**	**sensiNact**	**SENIORSOME**	**IoTivity**
SOFIA2	Trough InterIOT	no	Through InterIOT	no	no	Through MQTT

Table 4: SOFIA2 supported and planned bridges with other ACTIVAGE IoT platforms

3.1.3 universAAL IoT bridges status

universAAL [11] is an IoT open source platform that enables seamless interoperability of devices, services and applications in distributed systems. Its open source middleware can be integrated into many devices, products, and service solutions, regardless of branding. When embedded “universAALized”, components will communicate automatically, exchanging data and functionality that can be processed and reacted to.

In universAAL, the conceptual approach for integrating not “native” components – e.g., sensors and actuators using different communication protocols – is following a pattern that is very common among IoT platforms, namely a general-purpose reference architecture consisting of three layers, as depicted in Figure 1.

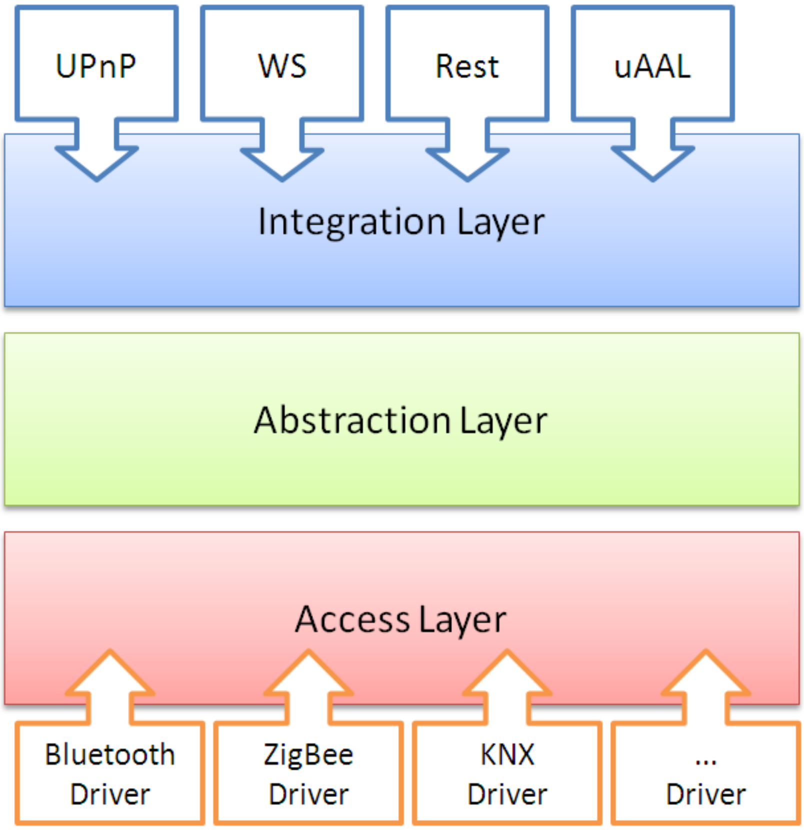

Figure 1: universAAL Access, Abstraction and Integration layers

Service or Access Layer (see Section 2) to is usually composed of a number of technology drivers, sometime already integrated in the operating system with well-defined API (e.g. Bluetooth), sometime written from scratch either with an ad hoc solution or using established libraries if standardized API are not available at the OS level (e.g ZigBee, KNX). So, this is the level to deal with the communication protocols.

The **Abstraction layer** is about the abstract representation of devices independently from the communication protocols. At this level, proxies are created for each device that can be accessed via the Access Layer. The proxies hide the protocol-specific communication details and represent the actual devices in a technology-neutral way.

Finally, the **Integration Layer** publishes the proxies instantiated in the abstract layer by creating new endpoints in the universAAL network; however, Figure 1 reveals that this approach is general enough for using the device representations on the Abstraction Layer for integrating the devices not only into the universAAL network but also in other networks (e.g., a UPnP network) or make them available globally with a REST API or as a Web Service). This enables the universAAL community to share device bindings up to the Abstraction Layer also with other communities.

The whole of this three-layer architecture (including the integration into universAAL) serves as what has been called “the device layer bridge” in section 2. In centralized deployments of universAAL on one single device serving as “gateway”, the “device layer bridges” implementing this three-layer architecture will reside on the gateway. In decentralized deployments of universAAL, such device layer bridges can be distributed on any universAAL node in the uSpace (the virtual space covered by a certain universAAL installation – whether in centralized or in decentralized form – that is protected by a “space”-specific key).

3.1.3.1 KNX universAAL bridge

KNX is a standardized (EN 50090, ISO/IEC 14543), OSI-based network communications protocol for intelligent buildings. KNX defines several physical communication medias: Twisted pair wiring, Powerline networking, Radio, Infrared and Ethernet. It is designed to be independent of any particular hardware platform. A KNX Device Network can be controlled by anything from an 8-bit microcontroller to a PC, according to the needs of a particular implementation.

KNX features a twofold structure: on one hand there are single devices (sensors/actuators) where one device can have multiple inputs or outputs, which are called *Communication Objects*. Those *Communication Objects* can be grouped in so called *Group Addresses*. *Group Addresses* can contain *Communication Objects* from sensors (e.g. switch) and actuators (e.g. light controller). Therefore *Group Addresses* can be seen as Functions (e.g. switching/controlling a light, control the heating/cooling, control shutters/blinds, etc.). All devices which should be controlled or monitored must be included in a *Group Address*.

For the Home/Building Automation domain universAAL provides a solution for integrating KNX, with a bundle suite (github.com/universAAL/lddi/tree/master/lddi.knx) that provides interaction facilities with KNX sensor networks. The suite includes the KNX message protocol and provides monitoring and control features for KNX devices (sensors and actuators) and exports them to universAAL buses. UniversAAL LDDI focuses on *Group Addresses*, not on single inputs/outputs (sensors/actuators).

3.1.3.2 ZigBee universAAL bridge

ZigBee [12] is a specification for small, low-power digital radios based on an IEEE 802 standard for personal area networks. Applications include wireless light switches, electrical meters with in-home-displays, and other consumer and industrial equipment that requires short-range wireless transfer of data at relatively low rates. The technology defined by the ZigBee specification is intended to be simpler and less expensive than other WPANs (wireless personal area networks), such as Bluetooth. ZigBee is targeted at radio-frequency (RF) applications that require a low data rate, long battery life, and secure networking.

The core components (Access and Abstraction layers) are provided by the ZB4OSGi project [13] as an open contribution to the OSGi community. Therefore, the integration of ZigBee into universAAL (github.com/universAAL/lddi/tree/master/lddi.zigbee) builds on top of this general-purpose OSGi solution. It provides for the following features:

- ZigBee Commissioning: Simulated Device to bind on cluster and configure reporting
- ZigBee Exporter: Implements universAAL wrappers for ZB Home Automation devices
  - Sends context events for incoming sensor events to the universAAL context bus
  - Provides services on the universAAL service bus for querying devices.

3.1.3.3 Bluetooth universAAL bridge (Continua-certified products)

universAAL supportsthe Continua Health Alliance certified devices (also called agents or sources) from the healthcare domain that use the wireless and low-cost Bluetooth technology (github.com/universAAL/lddi/tree/master/lddi.bluetooth).

The integration of Continua certified devices [14] inside universAAL is based on three different components (some of them can be shared between all environments without exception):

A component in charge of managing Bluetooth HDP connections and disconnections. Interface with the lower layers and Bluetooth drivers.

A component in charge of understanding received HDP frames from Continua agents and accordingly move through the appropriate x73 state machine diagram.

A component in charge of integrating HDP messages/Health data inside universAAL middleware.

The first component had to be ad-hoc developed for Windows and GNU/Linux OS because the Bluetooth stack is not the same in both architectures. The second and the third components can be re-used on different platforms without adding external libraries neither dependencies.

The provided universAAL bridge for Bluetooth has been tested with at least two Continua-certified devices namely a Weighing Scale and a Blood Pressure Monitor. They both transmit over **Bluetooth Health Device Profile (HDP)**. Furthermore, the Continua-proposed standard for in-home sensors measuring user activity, so called *Independent Living Activity Hub* (ISO 11073-10471) is adopted for sensor integration in universAAL.

3.1.3.4 FS20 universAAL bridge

FS20 is a simple wireless protocol developed by the German company ELV for the low-cost market segment of home appliances [15]. Its main advantage is the cheap price of hardware devices. Though the protocol is not that robust than comparable protocols in this domain (e.g. KNX or ZigBee).

The protocol is based on the exchange of simple events/commands between event-receiver and event-sender. In the universAAL project the integration of FS20 is based on existing software/driver components from the PERSONA project.

The FS20 integration (github.com/universAAL/lddi/tree/master/lddi.fs20) uses a two layer design comparable to the KNX Integration. Contrary to the OSGi DAS all FS20 devices are registered in the access layer via a service registration to OSGi. The exporter bundle in the integration layer notices the OSGi ServiceEvents and handles them.

3.1.3.5 ZWave universAAL bridge

ZWave is a home automation communication specification, similar to ZigBee although it was developed as a proprietary solution. The integration into universAAL is dependent on specific vendor devices, unlike the other, standard-based technology integrations. The existing ZWave Exporter (github.com/universAAL/lddi/tree/master/lddi.zwave) provides for the following features:

- Connects to ZWave gateway hardware
- Implements universAAL wrappers for ZWave Home Automation devices

- Sends context events for incoming sensor events to the universAAL context bus.
- Provides services on the universAAL service bus for querying devices.

3.1.3.6 Eclipse Smarthome (openHAB) universAAL bridge

Eclipse Smarthome (ESH) is the 2.0 version of OpenHAB [16]. It is a framework that integrates with several home automation standards, protocols and technologies to represent all their devices in a unified way. By integrating it, universAAL can get access to a lot of devices (github.com/universAAL/lddi/tree/master/lddi.smarthome), leaving the hardware integration to be performed by ESH.

Although ESH and universAAL share the same runtime environment but there are some incompatibilities due to system bundles and configurations. For this reason it is not possible to simply install standard universAAL Karaf features, instead there is a dedicated Karaf feature that has everything sorted out to install universAAL Middleware and the ESH Exporter in the ESH runtime. Then it is safe to rely on universAAL connectivity to link to other normal universAAL instances.

3.1.3.7 Summary of universaal IoT bridges status

IoT Protocol bridges for universAAL					
Protocol	**Domain**	**Layer**	**Connection edges**	**Status**	**Comment**
OpenHab	Domotic	Device	Device to LAN	Implemented	
Z-Wave	Domotic	Device	Device to Gateway	Implemented	
KNX	Domotic	Device	Device to Gateway Device to LAN	Implemented	
BLE-continua-x70	Health	Device	Device to Gateway	Implemented	Integration with the Continua certified devices
Zigbee	Generic	Device	Device to Gateway	Implemented	Home Automation Profile
Fs20	Domotic	Device	Device to Gateway	Implemented	German Domotic proprietary protocol
Philips Hue	Domotic lighting	Device	Device to LAN	To be implemented	
http/WSDL-SOAP	Generic	Service	Gateway to WAN (2way)	Implemented	
HTTP/REST	Generic	Service	Gateway to WAN (2way)	Implemented	

Table 5: universAAL supported and planned IoT protocol bridges

	SOFIA2	OPENIOT	FIWARE	sensiNact	SENIORSOME	IoTivity
UNIVERSAAL	Trough Interiot	Trough Interiot	Trough Interiot	no	no	no

Table 6: universAAL supported and planned bridges with other ACTIVAGE IoT platforms

3.1.4 OpenIoT bridges status

The OpenIoT (Open source blueprint for large scale self-organizing cloud environments for IoT applications) is an open source Internet of Things platform [17], which allows the collection, storage and processing of data stemming from heterogeneous set of IoT entities and resources. The platform introduces Sensing as a Service paradigm by interweaving the cloud computing with Internet of Things technologies. In order to collect data from multiple sensing devices, OpenIoT uses the Global Sensor Networks (GSN) [18] middleware. OpenIoT extends GSN (that has been called X-GSN) with RDF features and Linked Data, and provides a GSN-CoAP wrapper to access sensor data using a RESTful approach. GSN can retrieve data from various data sources. Table 7 below provides IoT protocol bridges for OpenIoT. In the context of the OpenIoT Project, OpenIoT leverages M2M/IoT techniques (such as CoAP), but also adds richer semantics to the inter-object communication. Furthermore, it provides user-friendly service interfaces for people to object interactions. The OpenIoT internal communication between sensors within a deployed network is considered as a “black box”. The input for OpenIoT sensor data delivery chain solely results from the data as provided by the gateway node.

3.1.4.1 ZigBee Bridge

ZigBee is an IEEE 802.15.4-based specification for a suite of high-level communication protocols used to create personal area networks with small, low-power digital radios, such as for home automation, medical device data collection, and other low-power low-bandwidth needs, designed for small scale projects which need wireless connection. In OpenIoT, the IoT devices connected to Gateway can transfer data using the ZigBee protocol.

3.1.4.2 BLE Bridge

Implementation of a Bluetooth Low Energy (BLE) bridge for OpenIoT is work in progress.

3.1.4.3 MQTT Bridge

MQTT is a simple and lightweight messaging protocol designed for constrained devices and low-bandwidth, high-latency and unreliable networks. It uses the publish/subscribe model of communication. The protocol minimizes network bandwidth and device resource consumption while keeping the reliability and assurance of delivery. OpenIoT has implemented MQTT bridge in order to communicate from its middleware to applications.

3.1.4.4 CoAP Bridge

CoAP protocol is standardized by the Constrained RESTful Environments (CoRE) workgroup, and is available as an active (IETF) Internet-draft. CoAP is based on a request/response communication model between entities (e.g. a sensing node and server), including key concepts found on the Web, thus enabling easy integration with the standard Web services. OpenIoT uses CoAP bridge for wireless sensor network data collection.

3.1.4.5 HTTP-GET Bridge

In the OpenIoT architecture, GSN serves as sensor middleware to publish sensor data form all kinds of physical devices via a common Web service interface. LSM sits on top of it, fetching the data from GSN via HTTP, transforming it into Linked Data, enriching it with semantic information, and storing the data into an RDF storage system.

3.1.4.6 HTTP REST Bridge

LD4Sensors (OpenIoT's tool for data annotation) is a RESTful Web server implemented using Java with the Jena library and the Jena Triple DB, leveraging on the OpenIoT GUI. The API allows to access, store, update, and delete specific resources that are typically involved in sensor measurements and sensor networks, after having semantically annotated them. Data can be accessed by querying a SPARQL endpoint, in addition to the REST API. This bridge also connects to a remote gsn-services instance and uses the REST API to poll data from the server. Thus, going through the oauth2 authentication and authorization process. All components within OpenIoT provide RESTful web services which are accessible via URLs.

3.1.4.7 HTTP SOAP Bridge

SOAP (Simple Object Access Protocol) is a protocol specification for the exchange of structured information and the implementation of web services. It uses XML as message format and typically uses HTTP for message negotiation and transmission. The raw sensor data collected by GSN is stored in a database system and can be accessed through an HTTP SOAP API.

3.1.4.8 RSS Bridge

This bridge allows extracting an RSS feed from a given URL. Due to the close relationship to HTTP GET bridge, the two parameters of the RSS wrapper are also a URL and the polling interval in milliseconds.

3.1.4.9 UDP Bridge

This bridge allows receiving arbitrary data on a UDP port of the machine on which it is running. There is only the port number as parameter for the UDP bridge.

3.1.4.10 XMPP Bridge

The eXtensible Messaging and Presence Protocol is an open standard based on XML for the near-instant exchange of messages and presence notifications. The main units of transferred information are called *stanzas*, which, in the context of XMPP, are self-contained XML snippets, for example, including the source and target of a stanza. In contrast to one-shot queries that are executed once and return a result, a continuous query over data streams generates new results each time new input data matches the query. To deliver results to any requesting application, OpenIoT supports various standard protocols including XMPP. Other protocols in this category include PubSubHubbub and WebSockets.

Protocol	Domain	Layer	Connection Edges	Status
ZigBee	Generic	Device	Device to Gateway	Implemented
BLE	Generic	Device	Device to Gateway	To be implemented
MQTT	Generic	Middleware	Gateway to WAN	Implemented
CoAP	Generic	Device	Device to Gateway	Implemented
HTTP-GET	Generic	Device	Gateway to OpenIoT	Implemented
HTTP REST	Generic	Middleware	OpenIoT to WAN	Implemented
HTTP SOAP	Generic	Device	Gateway to OpenIoT	Implemented
RSS	Generic	Device	Device to Gateway	Implemented
UDP	Generic	Device	Gateway to OpenIoT	Implemented
XMPP	Generic	Middleware	OpenIoT to WAN	Implemented

Table 7: OpenIoT supported IoT protocol bridges

Table 8 below provides a summary of the planned bridges between OpenIoT and other platforms. This is an initial planning; more platform bridges may be added in future as the project progresses.

IoT platform bridges for OpenIoT						
	UNIVERSAAL	**SOFIA2**	**FIWARE**	**sensiNact**	**SENIORSOME**	**IoTivity**
OpenIoT	Via InterIoT	no	Via InterIoT	no	no	Through CoAP/MQTT

Table 8: OpenIoT supported and planned bridges with other ACTIVAGE IoT platforms

3.1.5 OneM2M bridges status

Even if not selected as one of the seven ACTIVAGE plain IoT platforms, this section gives information on the status of available bridges in the OneM2M standard.

The oneM2M™ standard, which facilitates the integration of various industrial domains. This makes the platform suitable for large-scale projects and applications, including smart cities, smart factories, transportation and health care. From an architectural point of view, the platform is designed as a protocol-independent kernel on which the various components of a specific application can be plugged in as separate and reusable modules. The end points of an application, e.g. devices, sensors, gateways, servers, etc., are considered as nodes which expose various services for which higher-layer applications can subscribe in order to use. Such services include networking, device management, security, etc. The platform supports the integration of third-party devices, which accelerates the deployment of a functional IoT application.

IoT Protocol bridges for OneM2M					
Protocol	**Domain**	**Layer**	**Connection edges**	**Status**	**Comment**
COAP	Generic	Sensor	Device to Gateway	Implemented	
HTTP REST	Generic	Service		Implemented	
HTTP/REST	Generic	Service	Gateway to WAN (2way)	Implemented	
LoRa	Outdoor	Sensor	Device to Gateway	Implemented	yes REST
LoRaWAN	Outdoor	Sensor	Device to WAN	Implemented	yes REST
MQTT	Generic	Service	Gateway to WAN	Implemented	
NGSI 9 & 10 v1	Generic	Interoperability	Gateway to Gateway	Implemented	
NGSI 9 & 10 v2	Generic	Interoperability	Gateway to Gateway	Implemented	
OpenHab	Domotic	Device	Device to LAN	OSGi?	

Table 9: OneM2M supported and planned IoT protocol bridges

IoT platform bridges for OneM2M							
	UNIVERSAAL	**SOFIA2**	**OPENIOT**	**FIWARE**	**sensiNact**	**SENIOR SOME**	**IoTivity**
OneM2M	Trough InterIOT	no		NGSI/RESTful API via HTTP	available (MQTT)		Through CoAP/MQTT

Table 10: OneM2M supported and planned bridges with other ACTIVAGE IoT platforms

3.1.6 sensiNact bridges status

The sensiNact platform is dedicated to IoT and particularly used in various smart city and smart home applications. sensiNact aims at managing IoT protocols and devices heterogeneity and provides synchronous (on demand) and asynchronous (periodic or event based) access to data/actions of IoT devices, as well as access to historic data with generic and easy-to-use API.

The sensiNact platform interconnects IoT devices using different southbound IoT protocols such as Zigbee, EnOcean, LoRa, XBee, MQTT, XMPP, as well as platforms such as FIWARE and allows access to them with various northbound protocols such as HTTP REST, MQTT, XMPP, JSON RPC and CDMI. The Figure 2 below shows an overview of the bridges (southbound and northbound) around the sensiNact platform.

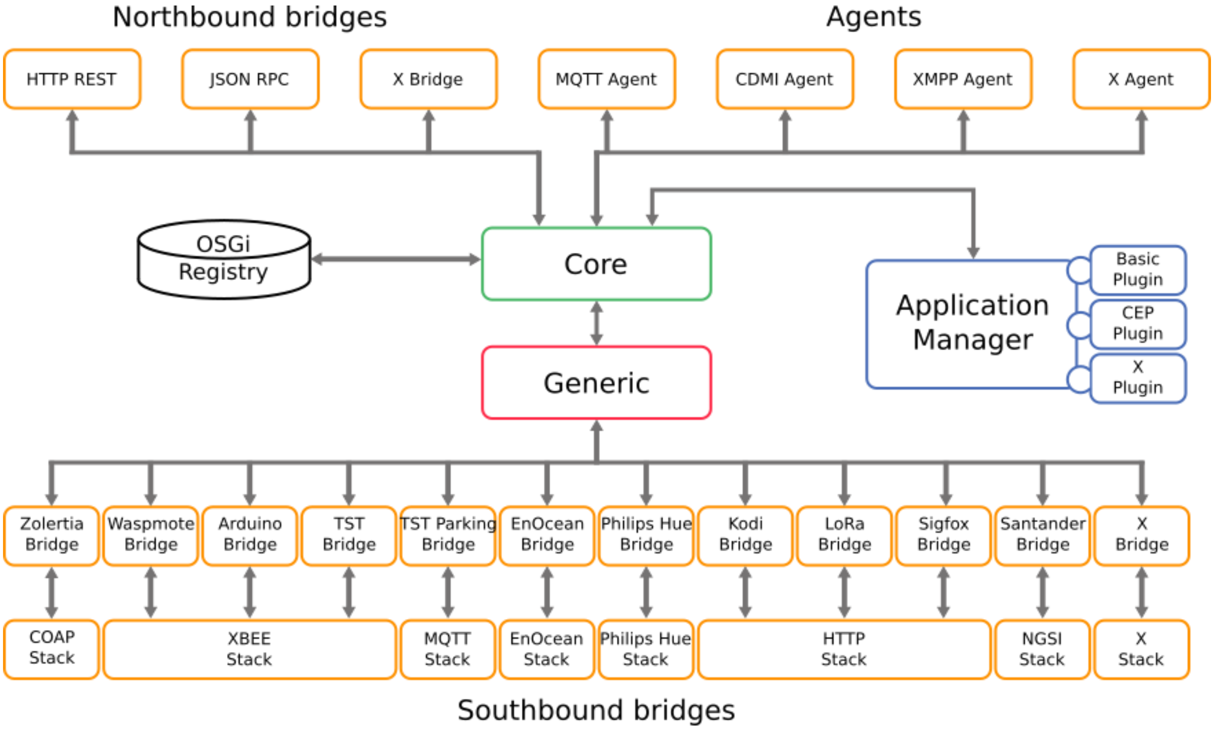

Figure 2: Overview of the sensiNact southbound and northbound bridges

The interactions between the gateway and other entities are performed through an extensible set of northbound and southbound bridges.

Southbound bridges are specialized into the interaction with devices, which can be sensors or actuators. Each bridge is in charge of communicating with a specific kind of device, using a given protocol. Out of the box, sensiNact ships with southbound bridges such as Zigbee (motion sensors, force sensor, etc.), EnOcean (remote controls, windows opener detectors, etc.), CoAP (sliders, buttons, etc.). It also provides a bridge for retrieving context information using NGSi 9/10 protocol. Thanks to an OSGi based architecture, it is possible to add bridges “on the fly”, while the gateway is running, to allow the communication with new devices. Of course, the creation of bridges relies on an API which delegates most of the integration work to the gateway, letting the programmer focus on the communication protocol and the data model of the device to be integrated.

Symmetrically to the southbound bridges, northbound bridges are in charge of publishing information to remote systems. It can be using common protocols, for example MQTT, XMPP, NGSi 9/10. The set of northbound bridges is also extensible, for tailoring special needs or singular systems. The REST API, which is a northbound bridge, is a key part in our architecture. It is designed for the administration of the gateway, thanks to a well-documented API.

A tutorial for southbound bridge implementation inside the sensiNact platform is provided in the public wiki, see [30].

The next sections give details about some of the currently available or planned sensiNact bridges.

3.1.6.1 Android IMU sensiNact bridge

Most Android-powered devices have built-in sensors that measure motion, orientation, and various environmental conditions. These sensors are capable of providing raw data with high precision and accuracy, and are useful if you want to monitor three-dimensional device movement or positioning, or if someone wants to monitor changes in the ambient environment near a device.

The Android platform supports three broad categories of sensors:

- Motion sensors:
  These sensors measure acceleration forces and rotational forces along three axes. This category includes accelerometers, gravity sensors, gyroscopes, and rotational vector sensors.
- Environmental sensors:
  These sensors measure various environmental parameters, such as ambient air temperature and pressure, illumination, and humidity. This category includes barometers, photometers, and thermometers.
- Position sensors:
  These sensors measure the physical position of a device. This category includes orientation sensors and magnetometers.

Android sensor framework provides the following capacities:

- Determine which sensors are available on a device.
- Determine an individual sensor's capabilities, such as its maximum range, manufacturer, power requirements, and resolution.
- Acquire raw sensor data and define the minimum rate at which you acquire sensor data.
- Register and unregister sensor event listeners that monitor sensor changes.

Personal Connected Health alliance [20] provides an overview of the sensors that are available on the Android platform. It also provides an introduction to the sensor framework.

This Android IMU sensiNact bridge allows sensiNact to receive Inertial Measurements from an Android device.

- Operating Requirements
  - The android-imu bridge must be activated at sensiNact configuration (via sensinact -c)
  - The Android device must run the Chrome browser
- How to use it
  - Point the Chrome (from your device) to the URL http://sensinact.address.com/imu/.
  - From that point on you can check the position of your device checking the service called android that is available in sensiNact
- Limitations
  - Multiple android devices are not supported yet.

3.1.6.2 CoAP Generic sensiNact bridge

The Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained networks in the Internet of Things.

The protocol is designed for machine-to-machine (M2M) applications such as smart energy and building automation.

Like HTTP, CoAP is based on the wildly successful REST model: Servers make resources available under a URL, and clients access these resources using methods such as GET, PUT, POST, and DELETE.

More details about the CoAP protocol, specifications, implementation and tools, can be found in [26].

The future CoAP sensiNact bridge (planned) will handle this IoT protocol. The implementation of this new generic bridge could reuse previous implementation done in a previous sensiNact version.

3.1.6.3 EchoNet sensiNact bridge

ECHONET [24] is a Japanese communication protocol designed to create the “smart houses” of the future. Today, with Wi-Fi and other wireless networks readily available in ordinary homes, there is a growing demand for air-conditioning, lighting and other equipment inside the home to be controlled using smartphones or controllers, or for electricity usage to be monitored in order to avoid wasting energy.

To achieve this kind of low-energy, comfortable, safe and reassuring lifestyle, we first need to create a system of rules or a “communication protocol” that can be read by any manufacturer’s equipment. This is why ECHONET was chosen and integrated.

The ECHONET Lite specification [25], in particular, is a communication protocol compatible with the now ubiquitous Internet. It is designed for ease of use and is simpler than the ECHONET specification.

The ECHONET Lite specification is already compatible with more than 100 types of device, and is also being adopted by the smart electric energy meters that will be installed in all households in future.

3.1.6.4 e-Lio app sensiNact bridge

e-lio is a user friendly platform dedicated to residential care markets (retirement homes, assisted-living residences, hospitals ...) that enhances the relationships between the caregivers, the residents and their families.

- For professionals, from any computer or telephone:
  - Animation of the establishment (activities, menus, pictures)
  - Communication with the families (news, pictures, messages)
  - Video Calls
  - Nurse call coupling
  - Adaptation to the resident's autonomy
- For families, from any computer, tablet or connected smartphone:
  - Audio and video Calls
  - On-line information news

- Communication with the resident (pictures, messages, diary)
- Communication with the establishment (messages, pictures,..)
- Advises and information

More details can be found in [21]

The future e-lio sensiNact bridge (planned) will make possible to get data coming from the e-lio application (information about the occurrences of audio and video calls, tv interactions through the e-lio remote control,…) into the sensiNact service layer.

3.1.6.5 EnOcean sensiNact bridge

The EnOcean technology is an energy harvesting wireless technology used primarily in building automation systems, and is also applied to other applications in industry, transportation, logistics and smart homes. Modules based on EnOcean technology combine micro energy converters with ultra low power electronics, and enable wireless communications between batteryless wireless sensors, switches, controllers and gateways.

The sensiNact platform includes an EnOcean bridges that manages 17 profiles for switches (F6-02-01, F6-02-02, F6-02-03, F6-02-04, D5-00-01), occupancy sensors (A5-07-01), temperature and humidity sensors (A5-02-01, A5-02-02, A5-02-03, A5-02-04, A5-02-05, A5-04-01, A5-04-02, A5-04-03), and energy measurement sensors and actuators (D2-01-06, D2-01-08, D2-01-0A). This sensiNact EnOcean bridges is constantly upgraded with frequent additions of new profiles.

3.1.6.6 Free Mobile sensiNact bridge

The French telephone company Free mobile provides a "Notifications by SMS" feature, making it possible to send SMS on his mobile phone from any device with an Internet connection through a simple RESTfull API [40]. [41] gives documentation and proposes a tutorial using this API.

The sensiNact Free Mobile bridge implementation is made on the top of this RESTFull API.

3.1.6.7 LoRaWAN sensiNact bridge

The LoRaWAN sensiNact bridge makes possible to connect with the Orange LoRa platform (France). Some technical details can be found in [29].

3.1.6.8 MQTT sensiNact bridge

This bridge allows sensiNact to subscribe to a MQTT topic, and materializes a sensiNact device based on the messages. Details about configuration and activation of this MQTT bridge are available in the sensiNact wiki [19].

3.1.6.9 Netatmo sensiNact bridge

This bridge allows sensiNact to connect to a netatmo home security account, using the netatmo connect platform [22]

- Operating Requirements
  - Netatmo bridge activated (via sensinact -c)
  - Having an account in netatmo https://dev.netatmo.com/dev/myaccount
- How to use it

- Create a file netatmo-X.cfg where X is a name that will stand as OSGi service PID (for more details check felix Configuration Manager). The fields in this file are define as in a regular property file. The id will become the provider into sensinact (the name of the sensor), the other fields are explicity since the naming convention is the same used by netatmo.
- One example of file (named netatmo-myhome.cfg) content is shown below:

```
id=SENSINACT_PROVIDER_ID
username=YOUR_NETATMO_ACCOUNT_EMAIL
password=YOUR_PASSWORD
clientId=APP_CLIENT_ID
clientSecret=APP_CLIENT_SECRET
```

- In order to be taken into account, this file MUST be deployed in the directory $SENSINACT_HOME/cfgs.

- Limitations
  - Netatmo Connect limits are a security against malicious uses and have been designed to fit all other needs and project. These limits are detailed in [23].

3.1.6.10 openHAB sensiNact bridge

The open Home Automation Bus (openHAB) [27] is an open source, technology agnostic home automation platform which runs as the center of your smart home.

The openHAB sensiNact bridge is capable of finding the openHAB instances running on the local network and instantiate its switch devices into sensinact without further configuration.

The switch type devices on openHAB are devices that respond to ON/OFF command.

No configuration is necessary, as long as the UDP packages between openHAB and the sensiNact machines are not been blocked, sensiNact will be able to find the openHAB instance without issues. Be aware that some routers are configured to block this kind of package in multi-hop, and VPN tunneling as well. In this situation you can manually specify the openHAB address. Details about the configuration of the openHAB sensiNact bridge can be found in [28].

3.1.6.11 Philips hue sensiNact bridge

Philips hue is a line of color changing LED lamps and white bulbs created by Philips. It was introduced in October 2012. The hue system uses the Zigbee lighting protocol to communicate, and can be controlled over Wi-Fi via a Zigbee–WiFi bridge. The sensiNact hue bridge is implemented on the top of the provided Philips hue RESTFull API.

More details about the hue developer program and documentation on the API can be found in [31].

3.1.6.12 Summary of sensiNact IoT bridges

The Table 11 below summarizes the IoT protocol bridges (already supported and planned) for the sensiNact platform.

IoT Protocol bridges for sensiNact					
Protocol	**Domain**	**Layer**	**Connection edges**	**Status**	**Comment**
Android IMU	accelerometer gyroscope and magnetometer	Sensor	Device to WAN	Implemented	Mobile phone as a accelerometer gyroscope and magnetometer
BLE	Personal Area Network	Sensor	Device to Gateway	todo	RF; Profiles: TI_cc2650
COAP	Generic	Sensor	Device to Gateway	todo (done in a previous version)	RF
EchoNet	Smart Building	Sensor	Device to Gateway	Implemented	Wired
e-Lio app	TV activity, social networking, sound	Gateway	gateway	todo	API
EnOcean	Smart Building	Sensor	Device to Gateway	Implemented (17 profiles)	RF; Profiles: F60201, F60202, F60203, F60204, D50001, A50701, A50201, A50202, A50203, A50204, A50205, A50401, A50402, A50403, D20106, D20108, D2010A
Free Mobile	sms	Service	Gateway to WAN	Implemented	sms sender
HTTP REST	Generic			Implemented	Abstract bridge; useful for further HTTP bridges
KNX-RF	Generic	Sensor	Device to Gateway	To be implemented	RF
KODI	TV	Device	Device tyo Gateway	Implemented	Based on HTTP

Protocol	Domain	Layer	Connection edges	Status	Comment
LoRa	Outdoor	Sensor	Device to Gateway	Implemented	RF ; Development of ad-hoc profiles
LoRaWAN	Outdoor	Sensor	Device to WAN	Implemented	Connection to Orange platform
MQTT	Generic	Service	Gateway to WAN	Implemented	Northbound and southbound
Netatmo	Weather station	Sensor	Device to LAN	Implemented	wifi or wired ethernet
NFC ACR12	RFID card reader	Sensor	Device to Gateway	To be implemented	USB
NGSI 9 & 10 v1	Generic	Interoperability	Gateway to Gateway	Implemented	Related to FiWare
NGSI 9 & 10 v2	Generic	Interoperability	Gateway to Gateway	To be implemented	The v2 specification is not yet ready
OpenHab	Generic	Gateway	Gateway	Implemented	API
Philips Hue	Domotic lighting	Sensor	Device to LAN	Implemented	wired ethernet
Sigfox	Generic	Sensor	Device to WAN	To be implemented	RF
Tikitag	RFID card reader	Sensor	Device to Gateway	Implemented	RFID ; deprecated
X3D	Generic	Sensor	Device to Gateway	To be implemented	RF, Delta Dore proprietary protocol
Xbee	Generic	Sensor	Device to Gateway	To be implemented	RF
XMPP	Generic	Sensor	Device to Gateway	Implemented	
Z-Wave	Generic	Sensor	Device to Gateway	To be implemented	RF

Table 11: sensiNact supported and planned IoT protocol bridges

IoT platform bridges for sensiNact						
	UNIVERSAAL	**SOFIA2**	**OPENIOT**	**FIWARE**	**SENIORSOME**	**IoTivity**
sensiNact	no	no	no	available (NGSI/HTTP)	no	Through MQTT

Table 12: sensiNact supported and planned bridges with other ACTIVAGE IoT platforms

3.1.7 SENIORSOME bridges status

The SeniorSome platform enables the use of different IoT protocols, sensors and means to connect and build a large scale IoT network. Interoperability is achieved in two main layers: 1) physical device layer whereas there exists an API for connecting devices through different bridges (with APIs). 2) Further a layer is provided at the backend where full restful API resources are available. SeniorSome provides (aims to provide) open interoperability for the other IoT platforms.

IoT Protocol bridges for SENIORSome					
Protocol	**Domain**	**Layer**	**Connection edges**	**Status**	**Comment**
Bluetooth	Generic	Sensor	Device to Gateway	Implemented	
Wifi-direct	Generic	Sensor	Device to Gateway	Implemented	
COAP	Generic	Sensor	Device to Gateway	To be implemented	
HTTP REST	Generic	Sensor	Device to Gateway	Implemented	
HTTP/REST	Generic	Service	Gateway to WAN (2way)	Implemented	
MQTT	Generic	Service	Gateway to WAN	Implemented	
NGSI 9 & 10 v1	Generic	Interoperability	Gateway to Gateway	To be implemented	
NGSI 9 & 10 v2	Generic	Interoperability	Gateway to Gateway	To be implemented	

Table 13: SENIORSome supported and planned IoT protocol bridges

	UNIVERSAAL	**SOFIA2**	**OPENIOT**	**FIWARE**	**sensiNact**	**IoTivity**
SENIORSome	planned	under test	To be implemented	To be implemented	To be implemented	Tbc (Through MQTT)

Table 14: SENIORSome supported and planned bridges with other ACTIVAGE IoT platforms

3.1.8 IoTivity bridges status

The IoTivity stack has core and service layers 43[32]. The **core layer** provides the required infrastructure for handling connectivity (connectivity abstraction) and resource representation (OCResource). The **service layer** provides higher level abstractions for the resources like, groups of "things", virtual and logical sensor management and protocol plugin manager for interacting with non-OIC devices.

The IoTivity core layer uses the interfaces provided by the operating system to access and control standard radio technologies like Bluetooth, WiFi, Zigbee. It can provide the required connectivity layer abstraction for applications to talk to devices irrespective of the connectivity type. The set of protocol bridges supported by IoTivity are presented in Table 15.

IoT Protocol bridges for IoTivity					
Protocol	**Domain**	**Layer**	**Connection edges**	**Status**	**Comment**
Constrained Application Protocol (CoAP)	Generic	Sensor	Device to Gateway	Implemented	
MQTT	Generic	Service	Gateway to WAN	Implemented	MQTT is supported as a protocol plug-in
Wi-Fi Direct	Generic	Sensor	Device to Gateway	Implemented	
Bluetooth low energy	Generic	Sensor	Device to Gateway	Implemented	
Bluetooth	Generic	Sensor	Device to Gateway	Implemented	
ANT+	Generic	Sensor	Device to Gateway	Implemented	
Zigbee	Generic	Sensor	Device to Gateway	Implemented	
Z-Wave	Generic	Sensor	Device to Gateway	Implemented	
UPnP Device Control Protocol (DCP)	Generic	Sensor	Device to Gateway	Implemented	https://openconnectivity.org/developer/specifications/upnp-resources/upnp

Table 15: IoTivity supported and planned IoT protocol bridges

In an IoTivity network, things are represented as resources along with a set of REST style getters and setters (based on CoAP PUT, POST, GET, etc.). Services in IoTivity are built using resource representations of the devices/things.

Applications can make use of the IoTivity API via the SDK to access the "things" (OCResource) in an IoTivity network and perform operations on them.

Moreover, the **Web Service Interface (WSI)** [33] offered also by of IoTivity aims to simplify representation of web based services in a network of IoTivity devices. It can act as a conceptual bridging layer between the IoTivity network and web based services. Figure 3 presents the functional blocks of the Web Services Interface.

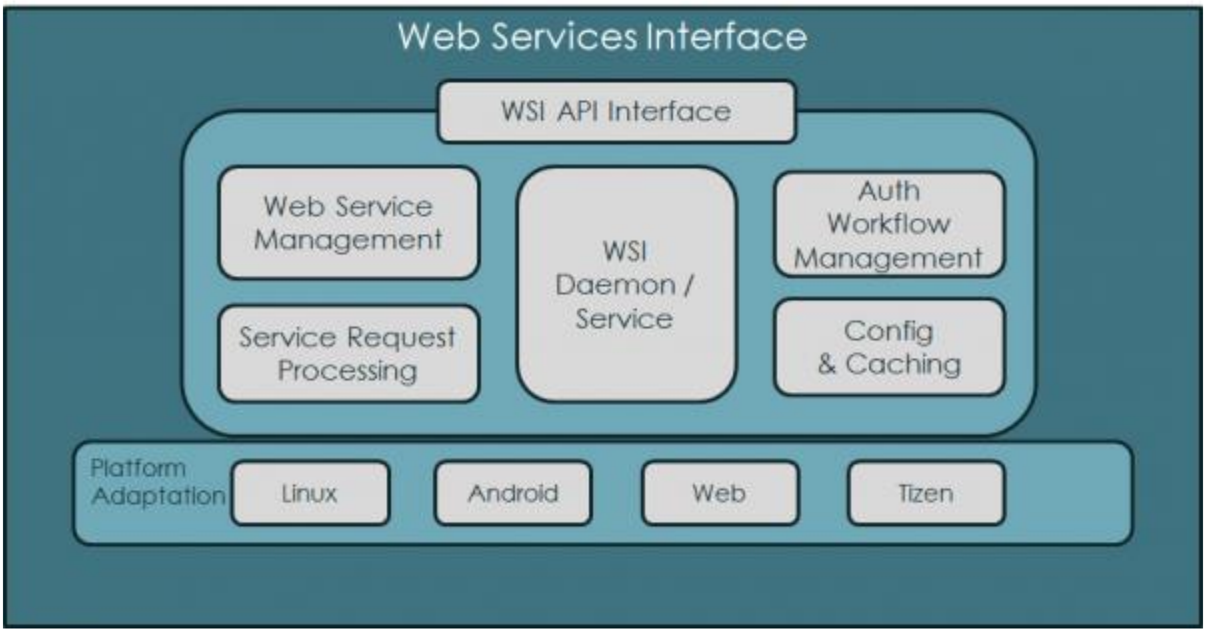

Figure 3: Functional blocks of the *Web Services Interface* of IoTivity

Except from the WSI, there are also 3 additional frameworks acting as bridges:

- The **Bridging Project** [34] – As IoTivity is compliant with only the OCF [35] ecosystem (with relatively low to no native devices on the market) the community needs a way to bridge into other ecosystems to leverage ongoing support for the ever-growing OCF ecosystem. It is beneficial for developers and product manufacturers to support the OCF ecosystem as early as possible. The Bridging Project aims at providing the proper infrastructure and corresponding plugins for supporting developers and product manufacturers.
- **UPnP-Bridge** [36] – UpnP-Bridge is an open source software framework for discovering existing UPnP devices/services. Then, it can translate UPnP devices/services/actions into IoTivity devices/services and register them as IoTivity devices through the Resource Container [37]. This allows existing UPnP devices/services/action to be managed and used through IoTivity like any other IoTivity device.
- **AllJoyn-Bridge** [38] – AllJoyn Bridge is an open source software framework that implements generic "on-the-fly" translation between AllJoyn [39] and IoTivity devices (both AllJoyn consumers/producers and IoTivity clients/servers).
-

Table 16 below presents the bridges between IoTivity and the other ACTIVAGE IoT platforms.

IoT platform bridges for IoTivity						
	UNIVERSAAL	**SOFIA2**	**OPENIOT**	**FIWARE**	**sensiNact**	**SENIOR SOME**
IoTivity	No	Through MQTT	Through CoAP/MQTT	Through MQTT	Through MQTT	Through MQTT

Table 16: IoTivity supported and planned bridges with other ACTIVAGE IoT platforms

3.2 The IoT protocols deployed by DSs

This section presents a table mapping edge protocols (at device layer) required by DS as drafted in the D9.1 deliverable. It is important to note that IoT protocols are not entirely stated for every deployment sites, and so the Table 17 is incomplete. The next version of this report on ACTIVAGE bridges for European platforms will complete this table including updates on actually deployed IoT protocols by DS as they will be fully described in the future D3.6 deliverable.

Deployed IoT protocols in Deployment Sites									
IoT protocol	**DS1 GAL**	**DS2 VLC**	**DS3 MAD**	**DS4 RER**	**DS5 GRC**	**DS6 ISE**	**DS7 WOQ**	**DS8 UK**	**DS9 FIN**
BLE	need				need	need			
BlueTooth	need	need	need		need			need	need
Ethernet REST API like	*need*							*need*	
Proprietary alarm sensor RF protocol (social alarm)	need								
Proprietary bed sensor RF protocol						need	need		need
SIP (phone ip)							need		TBC
Wifi	*need*	*need*		*need*	*need*			*need*	*need*
Zigbee	need				TBC				TBC
Z-wave			need		TBC	need	need		

Table 17: IoT protocol preliminary requests by deployment sites

The list of deployed IoT platforms by deployment sites is stated, and the following Table 18 can be considered as final.

DS \ Platform		universAAL	SOFIA2	OPENIOT	FIWARE	sensiNact	SENIOR SOME	IoTivity
DS1	GAL		deployed					
DS2	VLC	deployed			deployed			
DS3	MAD	deployed						
DS4	RER				deployed			
DS5	GRC	deployed			deployed			deployed
DS6	GNB					deployed		
DS7	WOQ	deployed						
DS8	UK							deployed
DS9	FIN						deployed	

Table 18: ACTIVAGE IoT platforms as they are deployed among deployment sites

3.3 Mapping between specified IoT protocols and IoT platforms

This section describes what are the IoT protocols edges/services supported/planned by each IoT platform. The table below allows to check that all needed protocols by deployment sites are supported by at least one platform.

Protocol \ Platform	UNIVERS AAL	SOFIA2	OPENIOT	FIWARE	sensiNact	SENIOR SOME	IoTivity
Android IMU					done		
ANT+							done
BLE	done for Continua certified devices		Todo		todo		done
Bluetooth						done	done
COAP			Done	todo (done in a previous version)	todo (done in a previous version)	todo	done
EchoNet					done		
e-Lio app					todo		
EnOcean					done (17 profiles)		
Free Mobile					done		
Fs20	done						
HTTP REST	done	done	done	done	done	done	

Protocol \ Platform	UNIVERS AAL	SOFIA2	OPENIOT	FIWARE	sensiNact	SENIOR SOME	IoTivity
HTTP SOAP	done	done	done	done			
KNX(-RF)	done				todo		
KODI					done		
LoRa					done		
LoRaWAN					done		
MQTT		done	Done	done	done	done	done (MQTT as protocol plug-in)
Netatmo					done		
NFC ACR12					todo		
NGSI 9 & 10 v1				done	done	todo	
NGSI 9 & 10 v2				done	todo	todo	
OpenHab	done				done		
Philips Hue	todo				done		done
RSS			Done				
Sigfox				done	todo		
Tikitag					done		
UDP			Done				
UPnP DCP							done

Protocol \ Platform	UNIVERS AAL	SOFIA2	OPENIOT	FIWARE	sensiNact	SENIOR SOME	IoTivity
Wi-fi Direct						done	done
X3D					todo		
Xbee					todo		
XMPP			Done		done		done
Zigbee	done		Done	todo			done
Z-Wave	done			todo	todo		done

Table 19: Edge protocol bridges for each IoT platform in ACTIVAGE project

4 Bridges among platforms

Previous works have been done to make interoperate the ACTIVAGE IoT platforms (see InterIoT project [42]). The table below presents the status of the available cross platform bridges.

IoT platform bridges							
	UNIVERS AAL	**SOFIA2**	**OPENIOT**	**FIWARE**	**sensiNact**	**SENIOR SOME**	**IoTivity**
UNIVERS AAL		Trough InterIOT	Trough InterIOT	Trough InterIOT	no	planned	no
SOFIA2	Trough InterIOT		no	Through InterIOT	no	under test	Through MQTT
OPENIOT	Trough InterIOT	no		Through InterIOT	no	todo	Through CoAP/ MQTT
FIWARE	Trough InterIOT	Through InterIOT	Through InterIOT		available NGSI/HT TP	todo	Through MQTT
sensiNact	no	no	no	available NGSI/HT TP		todo	Through MQTT
SENIORS OME	planned	under test	todo	todo	todo		tbc (Through MQTT)
IoTivity	no	Through MQTT	Through CoAP/ MQTT	Through MQTT	Through MQTT	tbc (Through MQTT)	

Table 20: Table of available cross platform bridges

5 Platform interoperability layer bridges

The platform interoperability layer bridges are the bridges between the seven ACTIVAGE IoT platforms and the future ACTIVAGE interoperability layer as specified in D3.2 deliverable:

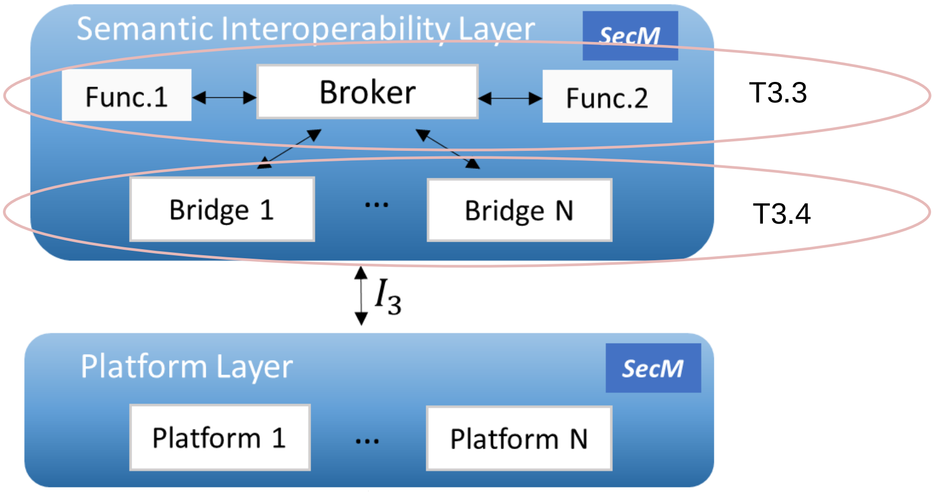

Figure 4: ACTIVAGE interoperability layer bridges

Future works in the tasks T3.3 and T3.4 will address the concrete interface specifications as well as the implementation of ACTIVAGE bridges between the seven platforms and the ACTIVAGE semantic interoperability layer.

The next version of this deliverable of the Task 3.4 (D3.10) will the details of the implementation of these seven ACTIVAGE platform interoperability bridges at month 24.

6 Summary and Conclusions

This section checks that deployment sites requirements in terms of IoT protocols are fulfilled either by the planned and existing IoT platform bridges implementation or through interoperation inside AIOTES. First listing the gaps between IoT protocols needed by DS and provided yet by platforms, this conclusion shows that solution can be handled with the ACTIVAGE interoperation approach.

6.1 Coverage of requested protocols among deployment sites

We can summarize the coverage of required edge protocols among the ACTIVAGE deployment sites using the table below (Table 21), showing the availability of requested edge protocols by DS (see Table 17) through their installed IoT platform(s) (see Table 18).

Deployed IoT protocols in Deployment Sites

IoT protocol	DS1 GAL	DS2 VLC	DS3 MAD	DS4 RER	DS5 GRC	DS6 ISE	DS7 WOQ	DS8 UK	DS9 FIN
BLE	need not supported by SOFIA2 **PLANNED 2018**	UNIVERSAAL (continua certified devices)			IoTivity and UNIVERSAAL	sensiNact			
BlueTooth	need not supported by SOFIA2 **PLANNED 2018**		need not supported by UNIVERSAAL **PLANNED 01/2018**		IoTivity			IoTivity	SENIORsome
proprietary bed sensor RF protocol						need not supported by sensiNact **PLANNED 01/2018**	UNIVERSAAL		need not supported by SENIORsome **PLANNED TBD**
SIP (phone ip)							UNIVERSAAL		need not supported by SENIORsome **PLANNED TBD**
zigbee	SOFIA2				UNIVERSAAL FIWARE and IoTivity				SENIORsome
zwave			UNIVERSAAL		UNIVERSAAL FIWARE and IoTivity	sensiNact	UNIVERSAAL		

Table 21: Check table between request and availability of edge protocol for each deployment site

6.2 List of edge protocol gaps by deployment site and action plan

From the Table 21: Check table between request and availability of edge protocol for each deployment site above, we can extract the gaps, i.e. required edge protocols that are not managed by the installed IoT platforms.

This table is not a complete and final description of the required IoT protocols, nevertheless this first analysis makes it possible to highlight the 4 pools of potential gaps among IoT protocol requirements in the deployment sites:

1. In the DS1 GAL deployment site, the SOFIA2 deployed IoT platform does not support needed Bluetooth, BLE and wifi protocols at device layer
2. In the DS3 MAD deployment site, it appears that the UNIVERSAAL platform does not support the needed Bluetooth protocol
3. In the DS6 ISE deployment site, the sensiNact platform does not support the proprietary bed sensor RF protocol.
4. In the DS9 FIN deployment site, the SENIORsome deployed IoT platform does not support needed SIP and proprietary bed sensor protocol at device layer

The next section proposes solutions to overcome these gaps through the ACTIVAGE interoperability layer.

6.3 Proposed solutions through the ACTIVAGE interoperability layer

It is up to the deployment site leader to manage the eventual edge protocol gaps through the ACTIVAGE interoperability layer, by installing among the ACTIVAGE IoT platforms one that offers connectivity to the vacant edge protocol(s).

Regarding the observed potential protocol gaps at DS level (Table 18), knowing the ACTIVAGE IoT platforms able to deal with the listed IoT protocol (Table 19) and getting existing interoperability bridge between the ACTIVAGE IoT platforms from Table 20, it is possible to provide advices to use interoperability layer as fix. The 4 following recommendations are the conclusion of this report on bridges for ACTIVAGE platforms.

- Recommendation 1: for DS1 GAT (missing Bluetooth and BLE handling) deployment site, one solution may be to deploy the IoTivity platform beside SOFIA2, using the future ACTIVAGE interoperability bridges, following the Use case 2 – Interoperability within Deployment Sites as described in D3.2.
  For their first deployments, DS1 uses a proprietary IoT platform developed by TELEVES, hosted in gateway. They actually plan to switch to ACTIVAGE platform in 2018 following the ACTIVAGE interoperability layer implementation plan.
- Recommendation 2: for DS3 MAD deployment site (missing also Bluetooth handling), one solution may be to deploy also the IoTivity platform beside UNIVERSAAL following the same Use case 2.

- Recommendation 3: for DS6 ISE deployment site, one solution may be to deploy UNIVERSAAL beside the sensiNact platform so as to manage the missing bed sensor communication protocol. For this interoperability use case, the Use case 3 – Interoperability within Deployment Site, with a single gateway hosting may be tested.
- Recommendation 4: for DS9 FIN deployment site, we can propose the same solution than DS6: deploy UNIVERSAAL beside the SENIORSome platform so as to manage the missing SIP and bed sensor communication protocol.

The Table 22 (next page) presents the four recommendations for the involved sites and the current status of their reflexion about these recommendations.

Regarding the Task3.3, the next version of the deliverable for the Task 3.3, scheduled on M24, related to bridges for European platforms, will give a more precise status of the interoperations among ACTIVAGE IoT platforms deployed in the DS. At month 24, required IoT protocols will be defined more accurately by deployment sites and interoperability layer will be available for first ACTIVAGE interoperations between European platforms.

Overview of recommendations to DS for ACTIVAGE interoperability					
DS		**IoT protocol gap**	**Recommendation**	**Status of reflexion**	**Comment**
DS1	GAT	Bluetooth + BLE	interoperate SOFIA2 with IoTivity	OK planned 2018	For their first deployments, DS1 uses a proprietary IoT platform developed by TELEVES, hosted in gateway. OK to switch to ACTIVAGE platform in 2018, following the progress of the AIOTES interoperability layer implementation
DS3	MAD	Bluetooth + BLE	interoperate UNIVERSAAL with IoTivity	checking if relevant	DS3 developed a module dedicated to their bluetooth lamp: specific bridge for the brand, modularized in such a way that it could become an official universAAL bridge. IoTivity bridges may not be appropriate for the DS3 bluetooth devices that most are not standard profiles (device manufacturers typically create a proprietary or even adhoc protocol over the serial profile: this is one of such cases for DS3 devices).
DS6	ISE	bed sensor proprietary protocol	interoperate sensiNact with UNIVERSAAL	OK planned 01/2018	For this interoperability use case, the Use case 3 – Interoperability within Deployment Site, with a single gateway hosting may be tested.
DS9	FIN	bed sensor and SIP proprietary protocol	interoperate SENIORSOME with UNIVERSAAL	proposition submitted	The needed bed sensor protocol might be handled by a proprietary solution

Table 22: Overview and status of the recommendations to DS for ACTIVAGE interoperability

References

[1] UNIVERSAAL IOT, SEMANTIC INTEROPERABILITY FOR RAPID INTEGRATION & DEPLOYMENT: HTTPS://GITHUB.COM/UNIVERSAAL

[2] SOFIA2 WEB SITE: HTTP://SOFIA2.COM/HOME_EN.HTML

[3] OPENIOT – OPEN SOURCE CLOUD SOLUTION FOR THE INTERNET OF THINGS: HTTP://WWW.OPENIOT.EU

[4] SENSINACT WEB SITE: HTTPS://PROJECTS.ECLIPSE.ORG/PROJECTS/TECHNOLOGY.SENSINACT

[5] FIWARE WEB SITE: HTTPS://WWW.FIWARE.ORG/

[6] IOTIVITY WEB SITE: HTTPS://WWW.IOTIVITY.ORG/

[7] SENIORSOME WEB SITE: HTTP://WWW.SENIORSOME.COM

[8] ONEM2M WEBSITE: HTTP://WWW.ONEM2M.ORG

[9] THE FUTURE INTERNET PUBLIC-PRIVATE PARTNERSHIP (FI-PPP) HTTPS://WWW.FI-PPP.EU/

[10] SOFIA2 WEB SITE: HTTP://SOFIA2.COM/HOME_EN.HTML

[11] UNIVERSAAL IOT, SEMANTIC INTEROPERABILITY FOR RAPID INTEGRATION & DEPLOYMENT: HTTPS://GITHUB.COM/UNIVERSAAL

[12] ZIGBEE ALLIANCE HTTP://WWW.ZIGBEE.ORG/

[13] ZB4O - ZIGBEE™ API FOR OSGI™ SERVICE PLATFORMS: HTTP://ZB4OSGI.AALOA.ORG/

[14] PERSONAL CONNECTED HEALTH ALLIANCE: HTTP://WWW.PCHALLIANCE.ORG/

[15] FS20 PROTOCOL: HTTP://FHZ4LINUX.INFO/TIKI-INDEX.PHP?PAGE=FS20%20PROTOCOL

[16] ECLIPSE SMARTHOME, A FLEXIBLE FRAMEWORK FOR THE SMART HOME: HTTPS://WWW.ECLIPSE.ORG/SMARTHOME/INDEX.HTML

[17] OPENIOT – OPEN SOURCE CLOUD SOLUTION FOR THE INTERNET OF THINGS: HTTP://WWW.OPENIOT.EU

[18] OPENIOT GITHUB: HTTPS://GITHUB.COM/OPENIOTORG/OPENIOT

[19] SENSINACT MQTT BRIDGE WIKI: HTTP://WIKI.ECLIPSE.ORG/SENSINACT/BRIDGE_MQTT

[20] DEVELOP > API GUIDES > LOCATION AND SENSORS > SENSORS OVERVIEW : HTTPS://GITHUB.COM/LSIR/GSNANDROID SENSOR OVERVIEW: HTTPS://DEVELOPER.ANDROID.COM/GUIDE/TOPICS/SENSORS/SENSORS_OVERVIEW.HTML

[21] TECHNOSENS WEB SITE: HTTP://WWW.TECHNOSENS.FR/

[22] NETATMOCONNECT WEB PAGE: HTTPS://DEV.NETATMO.COM/EN-US/DEV

[23] NETATMOCONNECT RATE LIMITS: HTTPS://DEV.NETATMO.COM/EN-US/RESOURCES/TECHNICAL/GUIDES/RATELIMITS

[24] ECHONET WEB PAGE: HTTPS://ECHONET.JP/ABOUT_EN/

[25] ECHONET OPEN SPECIFICATIONS: HTTPS://ECHONET.JP/SPEC-EN/#STANDARD-01

[26] COAP TECHNOLOGY: HTTP://COAP.TECHNOLOGY/

[27] OPENHAB WEB SITE: HTTPS://WWW.OPENHAB.ORG/

[28] SENSINACT OPENHAB BRIDGE WIKI: HTTP://WIKI.ECLIPSE.ORG/SENSINACT/BRIDGE_OPENHAB

[29] Discover the LoRa® Explorer Kit: https://partner.orange.com/discover-orange-explorer-lora-kit/

[30] A sensiNact bridge tutorial - the weather station example: http://wiki.eclipse.org/SensiNact/Tutorial_Bridge

[31] Philips hue developer program: https://www.developers.meethue.com

[32] IoTivity Web Service Interface (WSI): https://wiki.iotivity.org/web_service_interface

[33] IoTivity Web Service Interface (WSI): https://wiki.iotivity.org/wsi

[34] IoTivity bridging project: https://wiki.iotivity.org/bridging_project

[35] Open connectivity fundation: https://openconnectivity.org/developer/reference-implementation/iotivity

[36] IoTivity UPnP Bridge: https://wiki.iotivity.org/iotivity_upnp_bridge

[37] IoTivity wiki: https://wiki.iotivity.org/resource_container

[38] IoTivity github: https://github.com/iotivity/iotivity-alljoyn-bridge

[39] Allseen Alliance: https://allseenalliance.org/framework/documentation/develop/building

[40] Nouvelle option « Notifications par SMS » chez Free Mobile: https://www.freenews.fr/freenews-edition-nationale-299/free-mobile-170/nouvelle-option-notifications-par-sms-chez-free-mobile-14817

[41] http://www.domotique-info.fr/2014/06/nouvelle-api-sms-chez-free/

[42] INTER-IOT PROJECT: http://www.inter-iot-project.eu/