AAL PROGRAMME	**PETAL** www.aal-petal.eu	

Deliverable 3.1

Architecture Specification

Responsible Unit: CNR
Contributors: Ideable

Document Technical Details:

Document Number	D3.1
Document Title	Architecture Specification
Version	1.0
Status	Final
Work Package	WP3
Deliverable Type	Report
Contractual Date of delivery	30.09.2018
Actual Date of Delivery	09.10.2018
Responsible Unit	CNR
Contributors	Ideable
Keywords List	Personalization, Web Applications, Architecture, Context-dependent systems
Dissemination Level	Public

Document Change Log:

Version	Date	Status	Author	Description
0.1	18-09-2018	Draft	Silvia Tulli (CNR)	First Draft
0.2	24-09-2018	Draft	Iñaki Bartolome (Ideable)	Ideable Contribution
0.3	02-10-2018	Draft	Marco Manca, Carmen Santoro, Fabio Paternò (CNR)	CNR Contribution

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

Contents

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

1 INTRODUCTION

The PETAL project aims to offer a personalizable lighting system and an ambient assisted living ecosystem for older adults with Mild Cognitive Impairments. The objective of the PETAL project is to extend the time older people can live in their home environment by increasing their autonomy and assisting them in carrying out activities of daily living. Therefore, the solution proposed should consider the Mild Cognitive Impairments physical and cognitive issues (e.g., reduced sight, balance disorder, irregular sleep-wake rhythm, memory loss).
In the PETAL project we plan to achieve this goal by developing a technological platform able to gather relevant information about the users' behaviour (e.g., cognition status changes), and then providing personalized support to them (e.g., lighting system, reminder for cognitive stimulation activities). The platform should allow non-technical users (formal and informal caregiver or older adults users themselves) to personalize the functionalities of the relevant applications to tune the behaviour of the home appliances (in particular the lighting system), and its connected services when relevant events occur or when the specified condition are verified (Figure 1).
Moreover, the developed platform should send reminders or alarm messages to caregivers, relatives and even to the older adults in order to remind them to take medications or to report dangerous situations.
We aim to **improve the MCI people's quality of life**, **decrease caregiver burden**, **decrease the need for institutionalization**, and **reduce overall healthcare costs** by monitoring the environment and user behaviour and allowing the user to control applications and devices.
In order to be effective for the wide diversity of users, applications should have **flexible and highly personalized context-dependent behaviour**.

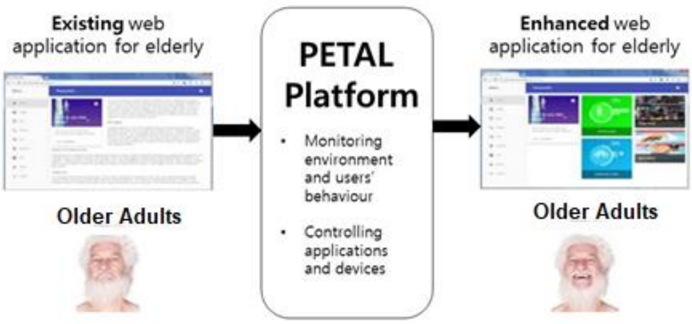

Figure 1: Abstract view of the PETAL platform

Thanks to the PETAL platform, the lighting system and the existing application aimed at helping elderly people can be personalized to provide alarms and reminders or support the sleep-wake rhythms. According to user needs and the surrounding environment the platform adapts its functionalities and user interface to the specific abilities of the MCI older adults (e.g., cognitive and vision impairments).

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

2 HIGH LEVEL DESIGN OF THE ARCHITECTURE

In this section, we will explain the High-Level Design (HLD) of the architecture that will be used for developing the PETAL platform. The HLD architecture provides an overview of the entire system identifying the main components that will be developed. The system architecture describes the modules and their objective/goals, as well as the basic logical/functional relationships occurring between such introduced modules. The main module of the platform is the **Personalization module,** which will provide support for monitoring the actual context of use and adapt the considered application according to rules indicated by caregivers or elderly people. For example, assume that the application for lighting already provides lighting controls based on user motion. The application could be instructed to turn off the lights if the user moves to another room or to change the colour or intensity of the lights if the older adults have visual deficits.

In particular, since we expect that most older adults suffering from MCI will not be able to personalize their applications according to their specific needs on their own, we suppose that formal or informal caregivers will contribute to personalizing the applications of the MCI users, on behalf of them. In order to enable the caregiver to define the adaptive behaviour of the platform, we will provide a Personalization Environment for defining the relevant events/conditions and actions (personalization rules). According to the "personalization rules" specified by this tool, the applications used by the older adults should be adapted to various contexts.

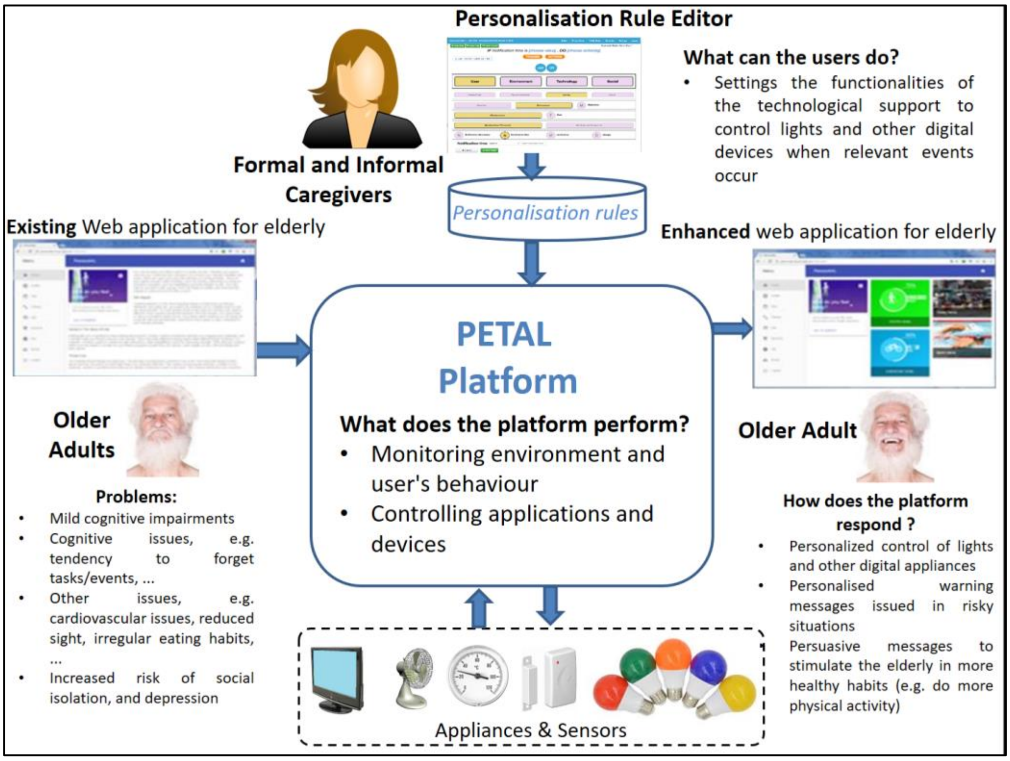

Figure 2: Main components of the PETAL Platform

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

2.1 Context-Dependent Behavior

The context-dependent behaviour of the platform is organised according to the context model's four main dimensions: user, environment, technology and social relationship. Each dimension is composed by various entities. The user is described in terms of personal data (age, gender, education), position (relative or absolute) and cognition (self-assessment value, emotional state, cognitive state, training results, training time, …). The environment is defined in terms of date and time and ambient attributes (light level, noise level, temperature, humidity, gas level …). All dimensions will be refined in the future depending on the actual features of the target contexts of use (Figure 3).

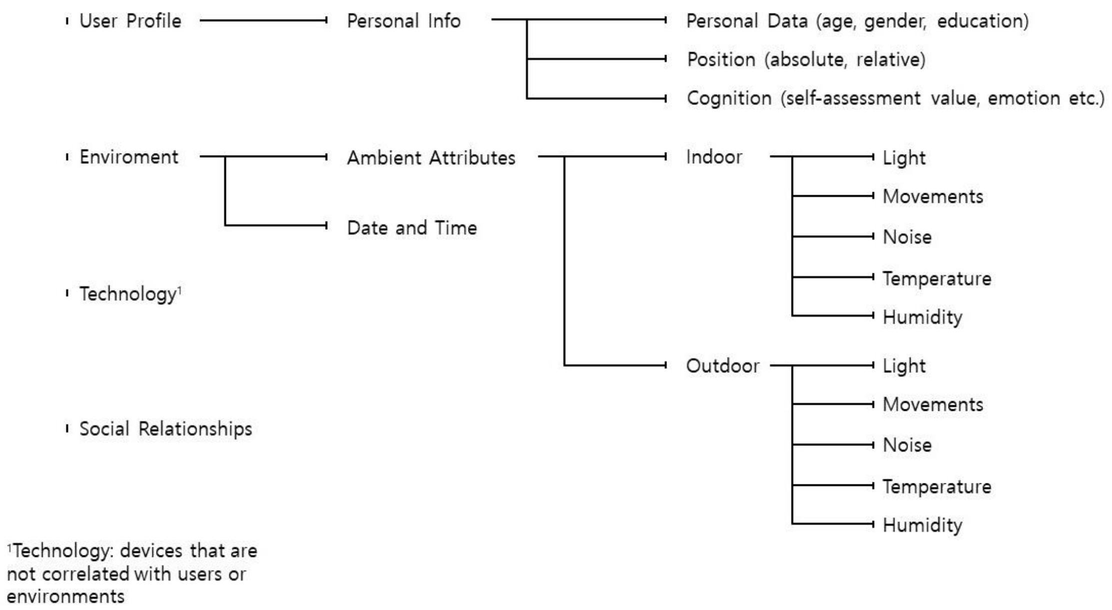

Figure 3 – Initial context entities of the PETAL Platform

2.2 Platform Components

The platform components for monitoring the environment and user behaviour and controlling applications and devices are:

- The **Context Manager** is composed of two different modules: Context Server and Context Delegates. The first one acts as a bridge between the information detected by the sensors and the Rule Manager. It provides a hierarchical organization of the

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

information received from the sensors and informs the Rule Manager when an event occurs or when a condition is verified. The Context Delegates are software which are able to communicate with the sensors or the appliances in order to get their state and then send those values to the Context Server.

- The **Rule Manager** has a twofold functionality: it receives and stores the rules from the Personalization Rule Editor and acts as an in-between module between the Context Manager and the Applications. The applications subscribe to the Rule Manager in order to receive the actions which describe how to change their behaviour when a rule is triggered. Then the Rule Manager retrieves the rules associated to the application and subscribes to the Context Manager in order to be informed when a rule is triggered.
- The **Personalization Rule Editor** allows formal and informal caregivers to define trigger-action rules for the target users and the context of use (Figure 2).
- The **Behaviour Analysis** analyses the Context Manager's historical information. The goal is to identify behavioural patterns and suggest new possible rules.
- The **existing Applications** subscribe for updates by sending a request to the Rule Manager and should be able to interpret and apply the received actions in order to change the state of the appliances accordingly.

2.2.1 Context Manager

Context awareness plays a crucial role in smart home applications, it improves the productivity of smart home as well as pervasive and mobile application (Non-Alisavath et al. 2017). The Context Manager manages the context awareness of the platform by collecting and analysing the contextual information coming from the environment/user/external sensors. The Context manager is a middleware Web-based application (Java Servlets, RESTful Web services) able to detect and communicate the data in a logical structure to the other modules (Rule Manager, Authoring Tool). The Context Manager is divided in Context Server and Context Delegates. The Context Delegates are software connected to the appliances and devices that update the content (data, information) of the Context Server. The Context Manager communicates to the Rule Manager only when the triggers specified in a rule cause the activation of it (i.e. when an **event** occurs or a **condition** is fulfilled).

2.2.2 Rule Manager

The Rule Manager is an in-between module between the context-aware applications and the Context Manager. It receives the rules defined through the Personalization Rule Editor and it administers the adaptation mechanisms by sending to the subscribed applications the changes to apply. It is a web-based application (Java Servlets) that applies the adaptation rules specified by the Authoring Tool. If multiple rules with opposite and conflictual effects are triggered, the Rule Manager may apply some priority mechanism to decide which rule should be applied.
During the requirement phase we considered multiple type of adaptations, indeed an application may want to change the appearance of its interface; an application may want to show reminders or alarm messages and finally an application may want to update the state of the appliances/devices which it is able to control.

In order to allow all these types of adaptation we implemented different communication channels between the Rule Manager and the Context-Aware Application (e.g. Web-Socket, REST services and MQTT messaging protocol). All these communication channels will be further explained in section 3.

2.2.3 Personalization Rule Editor

The Personalization Rule Editor is the End-User Development tool through which the user specifies the personalization rules. The adding value of EUD is the possibility for adult novices to personalize the platform functionalities combining Triggers and Actions. The usability and flexibility offered by the Trigger-Action paradigm can enable users with little prior experience with computers to personalize complex system behavior (Weintrop et al. 2018).
The language used for the EUD tool has to specify which contextual events should be considered and which consequent changes should be executed on the interactive applications. The rules are declared through an ECA-based (Event, Condition, Action) structure; where events are modifications of a context state. Conditions are Boolean predicates referring to a context state (they are optional) and actions are changes in the appliances state or activation of some functionalities.

- Events are associated to a change in the state of a contextual entity. The “event” part of a rule can be both an elementary event or a complex event (e.g., Boolean, comparison or sequential operators).
- Conditions refers to a persistent state of a contextual entity. The "condition" in a rule can be either elementary (e.g., Boolean predicates) or complex (composition of elementary conditions).
- An action specifies the changes which should be applied by the context-aware target application. Obviously, the actions are application dependent since they are applied by applications. Thus the actions may change depending on the target application and its functionalities. The “action” of a rule can be one or a set of actions which can be applied sequentially or in a parallel way.

Rules can have priorities, which are useful when multiple, conflicting rules occur simultaneously, thus priorities act as a mechanism to identify the rule which is the most likely to be triggered.

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

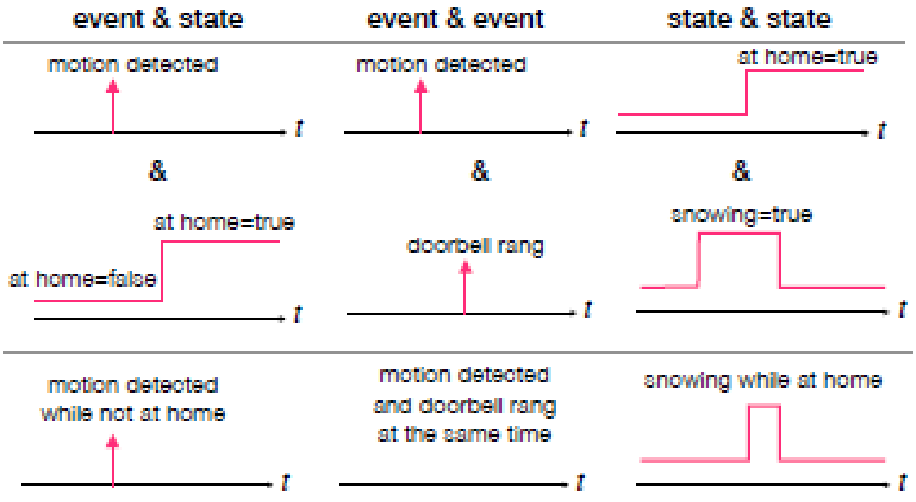

Figure 4: Example of Context-Dependent Triggers

2.2.4 Behaviour Analysis

The goal of the Behaviour Analysis is to understand if there are any deviations to the standard behaviours, rules or routines in the overall environment and among the users (Parvin, 2018). The tools and technology used by the users, the frequency of use of certain appliances and devices, and the "personalization rules" specified can be used to extract recurrence patterns and define a behavioural model. We want to investigate the possibility to suggest new personalization rules to the users or automatically adapts the system behaviour based on the behavioural patterns identified.

2.2.5 Applications

The PETAL platform communicates with existing web applications. In the PETAL project we consider a Cognitive Stimulation Application. The Cognitive Stimulation Application acts as a Context Delegate providing to the Context Server the information about the cognitive user status. The End-User Developer specifies the rules through the Personalization Environment, the Rule Manager applies the personalization rules by sending to the applications the actions to apply. The Applications interprets the actions and applies them by interacting with appliances and devices or by showing reminder or alarm messages. The Context Delegates connected to the appliances, devices and sensors update the content (data, information) of the Context Server. The Context Manager communicates to the Rule Manager that the event/condition defined by the "personalization rules" have been triggered.

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

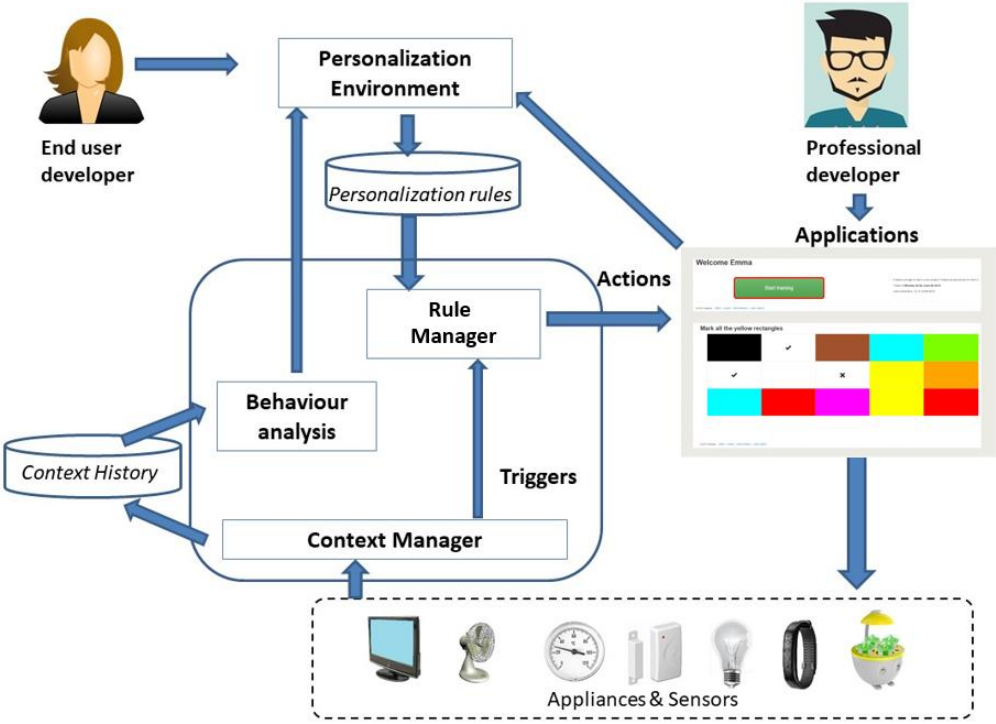

Figure 5: Integration of Applications with the Personalization Platform

2.2.6 Architecture of the Platform integrated with Sensors

A network of sensors and actuators will be installed to collect contextual information about the home environment and the users. Changes in sensors state may trigger personalization rules related to their value. The inclusion criteria for the selection of the sensors are: all sensors must be commercial, all sensors must be easily installable, all sensors must have a communication channel to connect them with the PETAL platform.
Due to this we consider if the sensors are standalone and wireless. Currently, the measurements taken into account are:

Environmental measures:

- Motion (true/false)
- Light detection (lux)
- Temperature (C°)
- Humidity (%)
- Gas (true/false)
- Windows/Doors state (Open/Close)

Obviously, all these metrics will be related to the different environments which compose the actual context of use, i.e. during the trials it will be possible to define rules referring the motion or the temperature of a specific room.

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

Additionally, the following general environmental measures will be used, which are not related to a specific environment/room:

- Weather information and forecast (external humidity, temperature real and feels like, pressure, chance of precipitation, wind speed)
- Date and time

User measures:

- Position/Room (in which room the user is)
- Position/Time (how long a user stays in a room)
- Heart Rate (bpm)
- Sleep (how long did the user sleep tonight? How long did the user spend in light, deep or REM sleep?)
- Physical Activity (steps, distance, calories)
- Going out/in from/to home

Technology measures:

- State of appliances/devices
- Battery level of appliances/devices

Formal and informal caregivers can describe the behaviour of the target appliances/devices through the Rule Editor by defining the triggers and the actions which compose a personalization rule. The Personalization Rule Editor communicates with the Rule Manager (RM) by sending the defined rules and receives the model of the target context from the Context Manager (CM) in order to present the triggers to the End Users which they may compose to define a rule. The application will subscribe to the Rule Manager in order to receive the actions when a rule is triggered. After the Application subscription, the Rule Manager retrieves the Rules associated both to the Application and to the User and extracts the list of triggers. Those triggers are composed of Events and Conditions which take into account the state of the Context Entities. For this reason the Rule Manager subscribes to the Context Manager in order to be informed when an event occurs or a condition is verified (i.e. in order to be informed when a rule is triggered). When a rule is triggered the Context Manager notifies the Rule Manager which sends the corresponding actions to the target application.
The considered application may run on a client browser and in this case will be able to display alarms or reminder to the user. However, if we want to apply actions which update the state of smart appliances or devices it is not possible to rely on an application running on a client browser, indeed if the browser is closed or the tablet/laptop/pc is off the application is no longer able to receive and apply the actions.
In order to cope with such issues, we decided to consider in our architecture the openHab[1] technology. Figure 6 shows the architecture of the PETAL platform including the openHAB module. The open Home Automation Bus (openHAB) is an open source, technology -agnostic home automation platform which integrates different home automation systems, devices and technologies into a single solution. It provides a common approach to smart devices/appliances across the entire system, regardless of the number of manufacturers and sub-systems involved. OpenHab runs on many platforms including Raspberry Pi. It can act i) as an application, whose behaviour can be adapted depending on the context of use, in that it subscribes to the Rule

[1] https://www.openhab.org/

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

Manager, receives the actions and then changes the state of the appliances accordingly (see "setValue" arrow in Figure 6); and ii) as a Context Delegate (CD) which gets the state of the appliances and devices registered on it (see "getValue" arrow) and sends those values to the Context Server.

In particular, the main differences of the architecture shown in Figure 6 compared to the previous one (Ghiani et al., 2017) are that the Context Server and the Rule Manager (previously running at a centralised server at CNR) will be running on the gateway which is expected to be installed in the house of the older adults. In this way we expect that the runtime communications directed to/coming from the Context Server or the Rule Manager with the other architectural modules will not be affected by network latency, as they will be carried out through the wireless router.
The only module that currently remains external to the gateway is the Rule Editor, which is used at design time by the end user to create the rules. The communications involving the Rule Editor are basically the following ones: a) communications needed at configuration time, in order to populate the hierarchies of triggers and actions in the Rule Editor (see the dashed arrows in Figure 6); and b) the communication with the Rule Manager, carried out when the user selects the "Apply Rule" button (to send the concerned rule to the Rule Manager, which is in charge of storing them internally). Regarding a), before users start to use the Rule Editor, its hierarchies of triggers and actions have to be dynamically populated with the actual elements considered in the current context of the user (e.g. if a lamp is moved from a room to another room or if a device is not connected anymore to the platform, this should be properly and dynamically reflected in the Rule Editor). The hierarchy of trigger types relevant in the specific domain considered are modelled in an XSD file that the authoring tool loads when it starts. However, in order to properly support the specific, actual environment at hand, the Rule Editor needs to be 'populated' in terms of the actual triggers and actions available in the setting considered. To this aim, at initialization time, the Rule Editor asks the Context Server for the list of actual objects, sensors and devices available in the current context. According to the information received from the Context server (there is a rest service in the Context server for this purpose), the Rule Editor is able to populate the panel dedicated to triggers. A similar process is repeated for the actions: in order to populate the panel dedicated to actions, the Rule Editor asks the application about the specific actions that are available. Such configuration is done at starting time (when the user starts the Rule Editor), therefore in Figure 6 the associated communications have been rendered through dashed arrows.

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

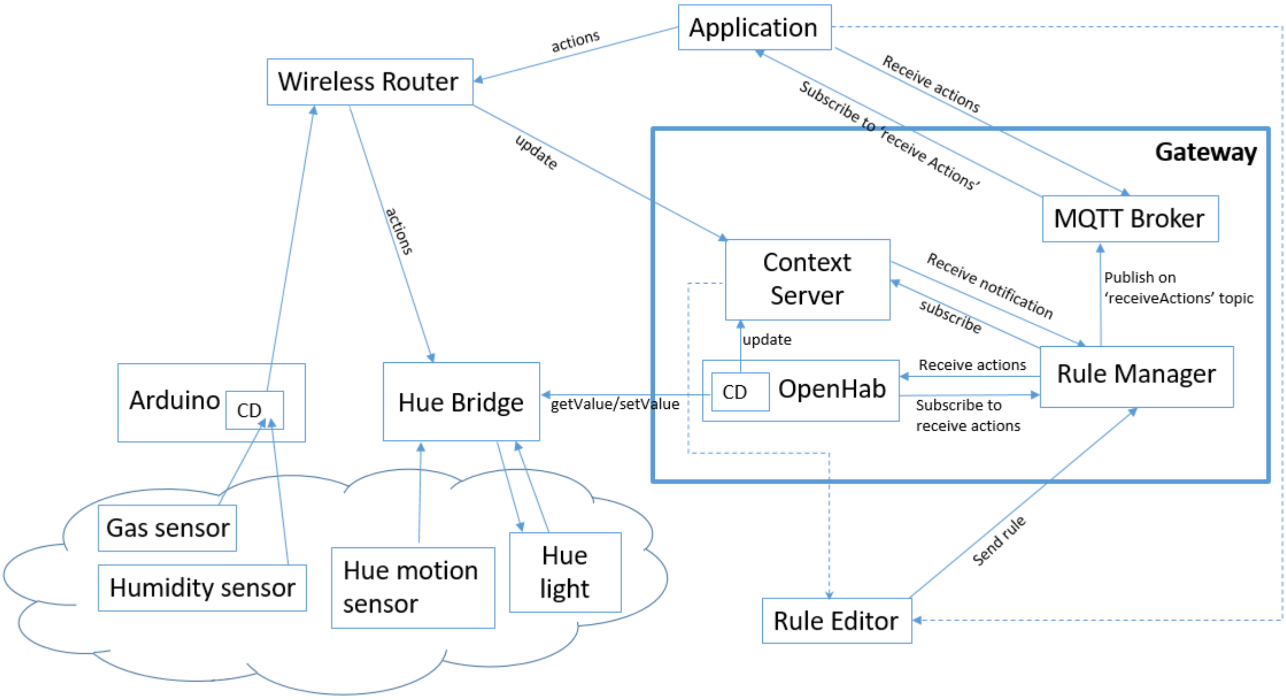

Figure 6: The PETAL architecture

The following list shows the sensors and the accessories we are considering to integrate in the platform.

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

Sensors KIT	Price	Portability	Standalone	Wireless Connectivity	Motion	Temperature	Light level (lux)	RGB Light Sensor	Humidity	Sleep Tracking and Stages	Heart Rate	Steps, Calories, Distance	Floors Climbed	Gas	Body Temperature	Body Position	Proximity
Fitbit Charge 2	€99,99	wearable	yes	yes	yes	no	no	no	no	yes	yes	yes	yes	no	yes	yes	no
Xiaomi Mi Band 3	€22,41	wearable	yes	yes	yes	no	no	no	no	yes	yes	yes	yes	no	yes	yes	no
Hue Motion Sensor	€34,99	portable and fixed	yes	yes	yes	yes	yes	no	no	no	no	no	no	no	no	no	yes
Hue Bridge	€53,52	portable and fixed	yes	yes	no	no	no	no	no	no	no	no	no	no	no	no	no
Estimote Proximity Beacons	£99,00	portable and fixed	yes	yes	yes	yes	yes	no	no	no	no	no	no	no	no	no	yes
Estimote Location Beacons		portable and fixed	yes	yes	yes	yes	yes	yes	no	no	no	no	no	no	no	no	yes
Grove Gas Sensor (MQ5)	£6,90	fixed	no	no	no	no	no	no	no	no	no	no	no	yes	no	no	no
Arduino UNO R3	€26,39	fixed	no	no	no	no	no	no	no	no	no	no	no	no	no	no	no
DFRduino Wireless Ethernet Shield	€5,14	fixed	no	no	no	no	no	no	no	no	no	no	no	no	no	no	no
DH11 Temperature & Relative Humidity	€3,54	fixed	no	no	no	yes	no	no	yes	no	no	no	no	no	no	no	no

Figure 7: Sensors considered in PETAL

Motion, light level and temperature can be measured by HUE Motion Sensors[2]. The HUE Motion Sensors have a range of 5 meters and a detection angle of 100 degrees in both horizontal and vertical direction. Hue motion sensors are connected to the Hue Bridge through ZigBee communication protocol, then the Hue Bridge exposes some REST services, which can be called

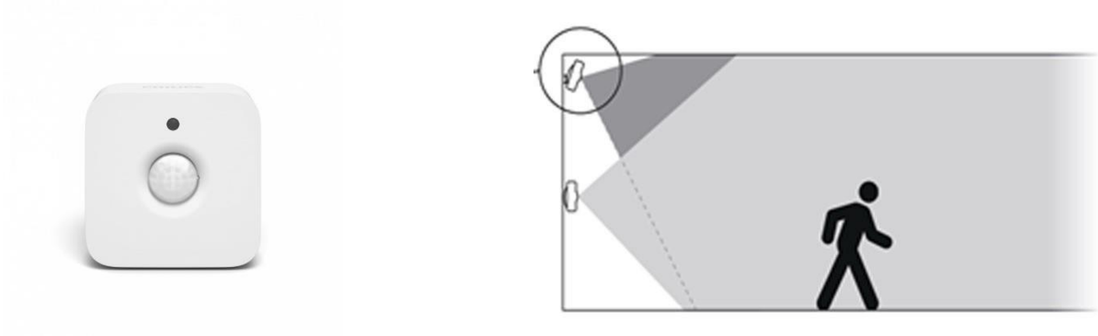

Figure 8: Hue Motion Sensor and Best Angle to Detect Movements

by the Context Delegate running on the OpenHab gateway. For instance, if there is a Hue Motion sensor and a Hue light registered to the OpenHab, the HUE Bridge offers REST services used to get the current state of lights/motion/temperature, and a Rest service used to update the status of registered lights.

[2] www2.meethue.com

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

We considered to use Estimote Beacons[3] for taking data about the location and proximity. While Estimote Proximity Beacons have a range of 70 meters, Estimote Location Beacons can detect the presence in a range of 200 meters.

The beacons send via Bluetooth a signal with an ID for their identification. When a smartphone running the Proximity Context Delegate is nearby, it detects the signal and, since a beacon is associated to a room, it communicates the user position to the context server via a wireless connection. The Proximity Context Delegate we developed exploits the Estimote SDK which informs when the smartphone enters/exits to/from a region identified by the beacon; when a user enters in a region we can start a timer and stop it when the user exits from the region; in this way, we can update the context manager sending the amount of time the user spent in a specific room.
During the Field Trials carried out for another AAL project we tested this solution which pointed out an issues: to identify the user position, the users need to have always a device with them. This situation, especially at home, is not always realistic. A possible solution may consider a wearable device that can be reasonable worn most time and is able to support both Bluetooth and Wi-Fi connection at the same time. We looked at smartwatches and most of them are not able to do this, some have Bluetooth and Wi-Fi but do not support open communication with both at the same time. This is an important feature because it enables the possibility to detect the current position (via Bluetooth) and immediately communicate it to the PETAL platform (via WiFi) so that if there are rules that depend on it (e.g. when the user enters in the kitchen then turn on the light) then they can be immediately triggered. There are two smartwatches that seem able to do this: 1-LEMFO LEM7 Android 7 Smart Watch 2- Zeblaze Thor 4 Smartwatch Android 7

Figure 9: Estimote Proximity Beacons

The user physical parameters can be detected by the Fitbit Charge 2. The Fitbit bracelet has: optical heart rate tracker, 3-axis accelerometer, altimeter and vibration motor. Heart rate is influenced by many different factors (e.g. physical exercises, sleep, illness) and it is used as an indicator of physiological adaptation and intensity of effort. Because of this, the heart rate sensor fit the objective of our project and can give relevant information about the user status and inform the system to react properly when it changes.

[3] https://estimote.com/

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

Table 1: List of Devices and Sensors integrated

Devices	Sensors
Hue Motion Sensor - connected to the Hue Bridge (zigbee protocols)[4] (motion, ambient light and temperature)	Motion sensor, Temperature sensor, Light sensor **Range** 5/12 meters and a detection angle of 100 degrees in both horizontal and vertical direction **Number of sensors** 12 different devices simultaneously
Estimote Location Beacons (location)	Motion sensor (3-axis), Temperature sensor, Ambient Light sensor, Magnetometer (3-axis), Pressure sensor, EEPROM Memory 1 Mb, RTC clock, GPIO, NFC **Range** 200 meters **Number of sensors** multiple simultaneously
Estimote Proximity Beacons (proximity)	Motion sensor (3-axis), Temperature sensor, Ambient Light sensor, programmable NFC, RGB LED **Range** 70 meters **Number of sensors** multiple simultaneously
Fitbit Charge 2 (position, heart rate, sleep, physical activity)	Optical heart rate tracker, 3-axis accelerometer, Altimeter, Vibration motor

In addition, the PETAL project plans to use a new standing-luminaire designed for another AAL-project (GREAT – Get Ready for Activity – Ambient day scheduling with dementia) that ensures all user requirements are addressed. The GREAT luminaire allows to control of colour temperatures, illuminance levels and light direction. It supports the daily activities guiding the attention and the spatio-temporal orientation with daylight or artificial light signals. The GREAT luminaire will be positioned in the most commonly used part of the home and thanks to the technological platform will be connected to the internet network and will be controlled by the openHub gateway. The GREAT luminaire integration enlarges the possibilities offered by the system and provides health provoking effects.

[4] Wireless RF mode frequency band: 2400-2483.5 MHz, Wireless communication protocol: IEEE 802.15.4, Operating channels: 11-26, Installing the Hue Motion Sensor on or near metal surfaces may lead to reduced range.

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

Figure 10: The GREAT-luminarie

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

3 COMMUNICATION ACROSS THE PLATFORM COMPONENTS AND APPLICATIONS

The communication across the platform components and the applications are managed by subscription. The Applications subscribe to the Rule Manager to be informed about relevant actions indicated by the rules (Call Rule Manager Rest service). They interpret and execute relevant actions received in JSON format.
The Rule Manager manages the rules associated to the application name and user name (specified during the subscription) by extracting events and conditions from the retrieved rules and by sending them to the Context Manager. When an event occurs and/or a condition is satisfied, the Rule Manager sends the actions to the Applications.

Depending on the type of adaptation needed it is possible to obtain different communication between the Rule Manager and the Application: e.g.,

1. an application may want to change the appearance of its interface; in this case the application can open a web socket communication with the Rule Manager to receive the rule using this communication channel
2. an application may want to show reminders or alarm messages; also in this case the application can use the web socket protocol, however if the application is not opened in the browser it will not receive the message
3. an application may want to update the state of the appliances/devices which it is able to control; if in this case we exploit the web socket communication then we have to face a problem: if the device is off or the browser is closed the application will never receive the actions containing the commands to change the state of the appliances.

A solution to this problem has been addressed by allowing communication between the Rule Manager and the Application through REST services: the application exposes a REST service in order to receive the actions and communicates the URL and the parameters of the service to the Rule Manager during the subscription phase.
In a previous AAL project we worked on we tested this solution by implementing a REST service on an Arduino connected to the router but we encountered another problem: the router was provided by the Internet Service Provider company which furnished the ADSL connection and it was completely closed, thus it was not possible to configure it in such a way to open the port (used by the application REST service) and to route all the request received to that port from the router to the Arduino.

For the PETAL Field trials, we developed another solution by allowing the communication between the Rule Manager and the Application by using MQTT protocol.

MQTT stands for MQ Telemetry Transport, it is a publish/subscribe, extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks. The design principles are to minimise network bandwidth and device resource requirements whilst also attempting to ensure reliability and some degree of assurance of delivery. These principles also turn out to make the protocol ideal for the emerging machine-to-machine (M2M) or Internet of Things (IoT) world of connected devices, and for mobile applications where bandwidth and battery power are limited. Since it is the application which subscribes for a topic to the MQTT broker, there is no need to open and route requests into the

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

router. Thus we implemented the possibility to send the actions using MQTT protocol by installing the MQTT broker called Mosquitto[5]. The Rule Manager sends the actions to the broker by publishing them using a specific topic and the application will subscribe to the same topic in order to receive them. The openHab gateway supports the MQTT protocol, thus there is no problem to use such communication protocol for the communication with the Rule Manager.

[5] https://mosquitto.org/

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

3.1 Integration of the Cognitive Stimulation Application

The Cognitive Stimulation Application is integrated with the Platform via endpoints published in Context Manager rest service.

Every 2 hours (configurable), Kwido Mementia gathers the situation of the people using the tool and send it to the Context Manager via REST endpoints. Context Manager analyses data and generate events to respond according the rules defined for each user.

The data sent from Kwido Mementia to Context Manager contains this information:

- Emotional State rank
  - Sad, 0 to 1
  - Discouraged, 1 to 2
  - Satisfied, 2 to 3
  - Pleased, 3 to 4
  - Happy, 4 to 5
- Training result rank
  - Stupendously 80.00 to 100.00
  - Nimbly 60.00 to 79.99
  - Well-Maintained 40 to 59.99
  - Not progressing well 0 to 39.99
- Cognitive State rank
  - Expert, 4
  - AAML, memory losses associated with age, 3
  - MCI, mild cognitive impairment, 2
  - MoCI, moderate cognitive impairment, 1
  - Automatic, 0
- Self-Assessment Value rank
  - Is a numeric value, representing a typical exam scale value from 0 to 10
- Training Time rank
  - Numeric value
- Time since last connection
  - Numeric value

There is an AWS Lambda recurrent service that acts as a bridge between Mementia and Context Manager. To do so, the information from Kwido Mementia is fetched recurrently by a REST API and sent to the platform by the REST API exposed.

In Kwido Mementia, besides exposing the required REST API endpoints for the synchronization, an extra field has been added to the users, containing in this Context Manager integration case, the user identifier from the security system in place, Auth0.

This way, every time a new user is provisioned in the platform via Auth0, that field will be added to the Mementia user.

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

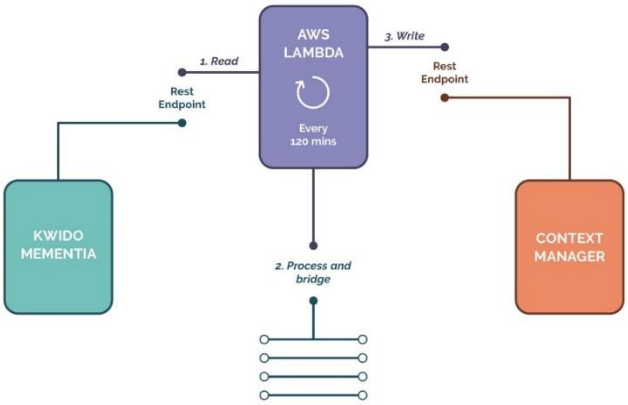

Figure 11: Amazon Web Service (AWS) lambda is a module connecting Kwido Mementia and the Context manager

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

4 CONCLUSIONS

In this deliverable we describe the initial architecture of the PETAL platform, also providing specific indications about how the various modules should communicate. This architecture is also the result of an analysis of drawbacks existing in a baseline version of the platform, which has been used in another AAL Project. In the current version of the platform almost all the modules run in a gateway, which is expected to be available in the house of end users and provide support for more real-time communications between the runtime modules. Thus, this platform seems sufficiently solid for supporting the first round of field trials planned in the project.

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.

5 REFERENCES

Non-Alisavath K., Kanthavong S., Luangxaysana K., Louangvilay X. (2017). *Context-Awareness Application to Control Multiple Sensors for Monitoring Smart Environment,* 14th International Conference on Electrical Engineering/Eletronics, Computer, Telecommunications and Information Technology (ECTI-CON)

Ghiani G., Manca M., Paternò F., and Santoro C. (2017). *Personalization of context-dependent applications through trigger-action rules*. ACM Trans. Comput.-Hum. Interact. 24, 2, Article 14 (April 2017), 33 pages. DOI: http://dx.doi.org/10.1145/3057861

Weintrop D., Afzal A., Salac J., Francis P., Li B., Shepherd D.C., Franklin D. (2018). *Evaluating CoBlox: A Comparative Study of Robotics Programming Environments for Adults Novices*, CHI 2018, April 21-26, 2018, Montrèal, QC, Canada

Parvaneh P., Stefano C., Marco M., and Fabio P. (2018). *Real-Time Anomaly Detection in Elderly Behavior with the Support of Task Models*. Proc. ACM Hum.-Comput. Interact. 2, EICS, Article 15 (June 2018), 18 pages. DOI: https://doi.org/10.1145/3229097

The project PETAL is cofunded by the Active and Assisted Living Programme (AAL-2016) and the following National Authorities and R&D programs in Italy, Spain, Austria and Romania.