Grant Agreement No.: 723174
Call: H2020-ICT-2016-2017
Topic: ICT-38-2016 - MEXICO: Collaboration on ICT
Type of action: RIA

D5.4: SmartSDK Roadmap v2

Revision: v.1.0

Work package	WP 5
Task	Task 5.1
Due date	31/05/2017
Submission date	31/05/2017
Deliverable lead	INFOTEC
Version	1.0
Authors	Hugo Estrada (INFOTEC), Blanca Vázquez (INFOTEC), Francisco Monsanto (UBIWHERE), Germán Molina (HOPU), Miguel González (ITESM), Alicia Martínez (CENIDET), Hugo Estrada (INFOTEC), Blanca Vázquez (INFOTEC), Tomas Aliaga (MARTEL) Antonio Macías (CICESE), Jesús Favela (CICESE), Enrique Succar (INAOE).
Reviewers	Federico M. Facca (MARTEL), Ricardo Vitorino (UBIWHERE)

Abstract	This document provides an overview of the planned R&D achievements focusing on the period from June 2017 to November 2017
Keywords	Technical Roadmap, Agile, Epics, FIWARE

Document Revision History

Version	Date	Description of change	List of contributor(s)
V1.0	31/05/2017	Final version ready for submission	Blanca Vázquez (INFOTEC), Federico M. Facca (MARTEL)
V0.4	19/05/2017	Add clarification on releases and sprints.	Federico M. Facca (MARTEL), Ricardo Vitorino (UBIWHERE), Hugo Estrada (INFOTEC), Tomas Aliaga (MARTEL)
V0.3	15/05/217	Integration of contributions from partners plus minor adjustments.	Francisco Monsanto (UBIWHERE), Germán Molina (HOPU), Miguel González (ITESM), Alicia Martínez (CENIDET), Hugo Estrada (INFOTEC), Blanca Vázquez (INFOTEC), Antonio Macías (CICESE), Jesús Favela (CICESE), Enrique Succar (INAOE).
V0.2	26/04/2017	Version ready for comments and integration by partners.	Federico M. Facca (MARTEL)
V0.1	15/04/2017	Table of Contents and initial version.	Hugo Estrada (INFOTEC)

Disclaimer

Copyright notice

Project co-funded by the European Commission in the H2020 Programme		
Nature of the deliverable*:	R	
Dissemination Level		
PU	Public, fully open, e.g. web	✓
CI	Classified, information as referred to in Commission Decision 2001/844/EC	
CO	Confidential to SmartSDK project and Commission Services	

** R: Document, report (excluding the periodic and final reports)*

DEM: Demonstrator, pilot, prototype, plan designs

DEC: Websites, patents filing, press & media actions, videos, etc.

OTHER: Software, technical diagram, etc.

EXECUTIVE SUMMARY

SmartSDK is the FIWARE's "cookbook" for developing smart applications in the Smart City, Smart Healthcare, and Smart Security domains. Concretely this means that SmartSDK refines, combines and develops new FIWARE Generic Enablers (GEs) and FIWARE Data Models into a set of well-codified and ready-to-use solutions. This is very important to facilitate the take up of FIWARE by new developers and its transition from proof-of-concept environments to production ones.

This document updates the first version of the Roadmap (D5.3) discussing covered developments during the period from September 2016 to May 2017, and providing an overview of planned Epics in the period from June 2017 to November 2017. Full documentation of stories and features is left in the Agile management tool (JIRA) for the interested readers.

This document describes an overview of the components defined and developed in the following three domains:

- Smart city for monitoring pollution and traffic;
- Smart security for intelligent video surveillance;
- Smart healthcare for mobile sensing.

In addition, activities for the SmartSDK platform and its contribution to the FIWARE Community will be focused on:

- Providing a first integrated release of the SmartSDK Platform Manager through of setup of Rancher Server on the FIWARE Lab;
- Developing additional generic components for enablers (hardware and software);
- Updating Cloudino to NGSI version 2.0.

The next release of this document (November 2017) will refine the components of the smart scenarios according to the evolution on the project and requirements in the applications.

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

This document updates the previous version of SmartSDK R&D roadmap (November 2016 - May 2017) with the aim to highlight the planned contributions to the FIWARE Open Source Community planned in the period from June 2017 to November 2017.

The roadmap is not set in stone and it will evolve throughout the lifespan of the project according to the different technical and non-technical aspects that influence the project: e.g. needs of the application scenarios, community relevance of the outcomes, available resources, and so on.

This first release of the SmartSDK Roadmap presented in this document is based on the initial activities defined through the Agile R&D methodology[1]. The document is not meant to be a detailed discussion of the identified epics, features and user stories. Those are detailed in JIRA following FIWARE Community best practises[2] and will evolve at each sprint (particularly user stories).

This deliverable focuses instead on providing a high-level picture of R&D plans of SmartSDK, including priorities for the upcoming months, in order to:

- Guide the SmartSDK developers in the R&D activities by keeping in mind main outcomes expected by November 2017;
- Provide hints to the wider FIWARE Community of SmartSDK plans, so as to favour the establishment of collaborations.

1.1. Concept

SmartSDK is the FIWARE's "cookbook" for developing smart applications in the Smart City, Smart Healthcare, and Smart Security domains. SmartSDK is looking into applications developed so far within Europe and Mexico (using FIWARE or alternative Open Source technologies), analysing them and making a cookbook for developing applications in the Smart City, Smart Healthcare, and Smart Security domains.

The "cookbook" is based on: a set of architecture patterns (i.e. the basic cooking processes), a set of Generic Enablers (i.e. the basic ingredients) and a set of data models (i.e. the spices and flavours binding the ingredients through the cooking process).

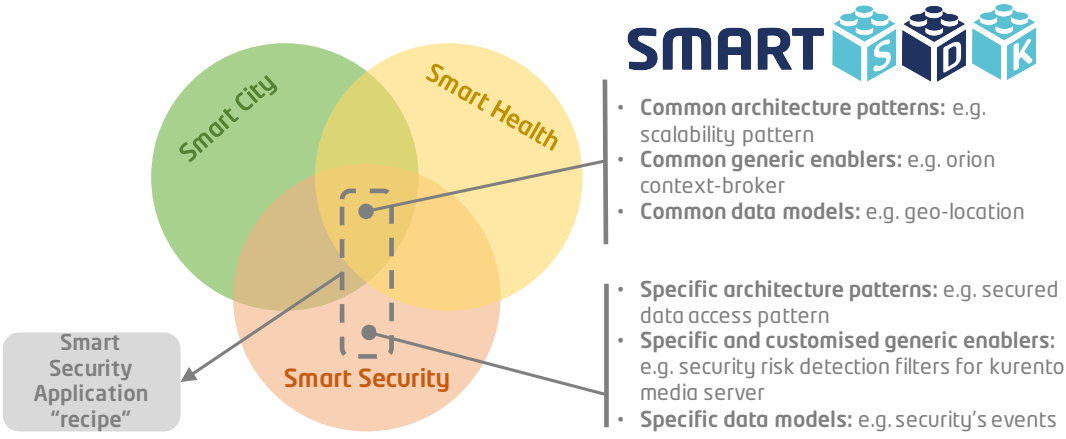

Figure 1. SmartSDK's cookbook concept.

[1] The methodology and tools are documented in D5.1: Quality Assurance Guidelines [2]. Appendix A of this document includes a summary of the methodology for the sake of the reader, given that D5.1 is a confidential document.
[2] https://jira.fiware.org/projects/SMAR

Cookbooks often include a set of basic recipes that act as a common basis for different dishes. Similarly, SmartSDK will include as well basic patterns, Enablers and data models that can be used for the development of data-intensive and IoT-based Smart applications and extend and complement them with specific patterns, Enablers and data models that are useful in the context of Smart City, Smart Healthcare, and Smart Security domains.

1.2. Overall architecture

SmartSDK will base its architecture on the FIWARE Reference Architecture and extend it to its wider scope. The core of the system supporting the deployment, management and monitoring of Smart applications is the SmartSDK Platform Manager.

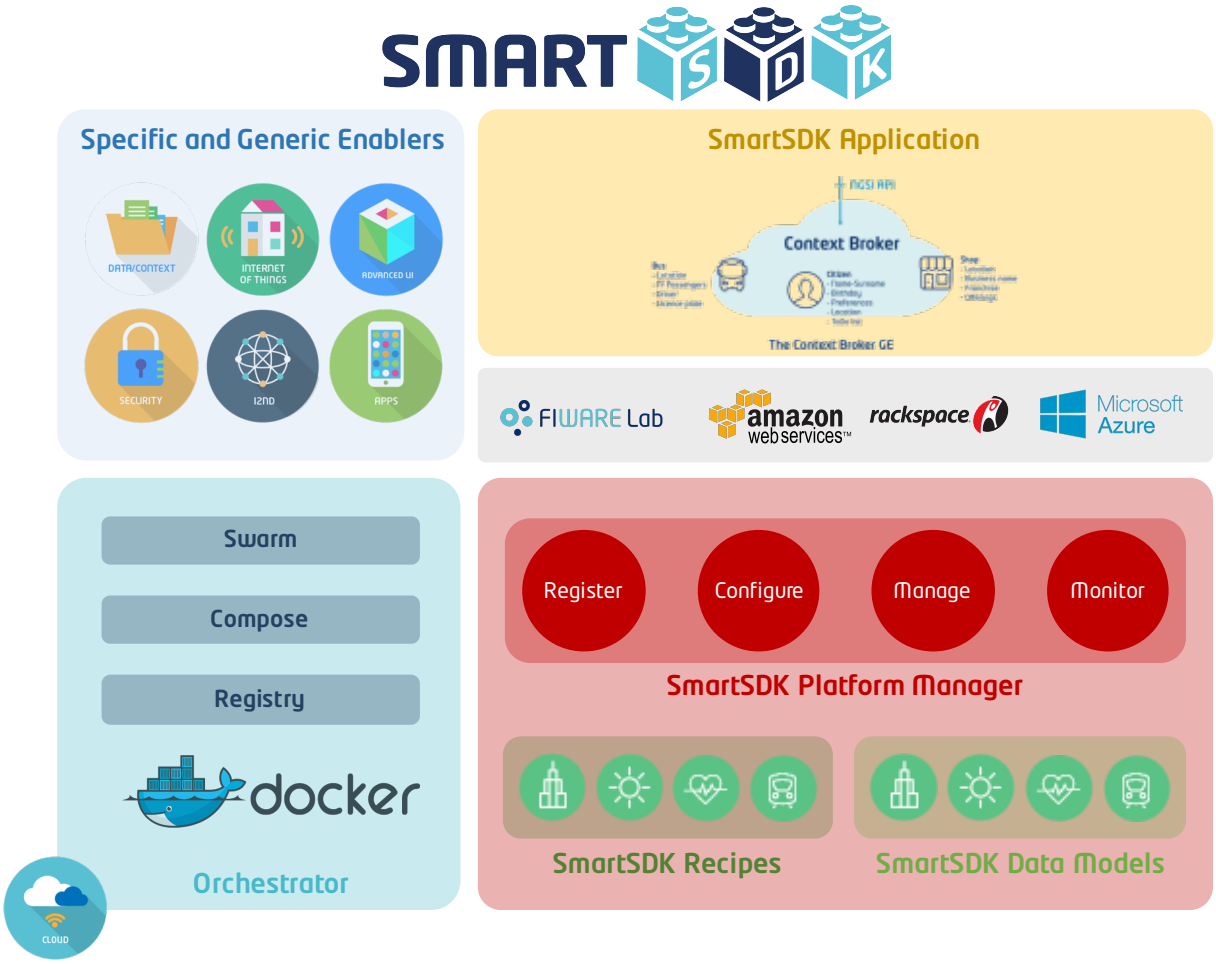

Figure 2. SmartSDK Architecture.

The SmartSDK Platform Manager will allow registering and maintaining the SmartSDK Recipes and SmartSDK Data Models:

- SmartSDK Recipes describe reference architectures for the application domains covered in the project and link them with Data Models. The recipes include relations among the Enablers and leverage on cloud patterns and typical FIWARE patterns for data-intensive and IoT-based applications.
- SmartSDK Data Models are NGSI-based information models defined to support the exchange of data in data-intensive and IoT-based applications in the domains covered in the project.

Recipes will be described using Docker Compose file descriptors or similar solutions (e.g. Kubernetes PODs descriptions) following the orientation that the FIWARE Cloud Hosting Chapter will take on containers orchestration.

Developers will be able to select a SmartSDK Recipe from a dashboard and provide configuration parameters for the deployment and dynamic self-configuration of their Smart application.

The SmartSDK Platform Manager will deploy and manage configured Smart applications through the FIWARE Cloud Hosting Generic Enablers, for example, leveraging on the Docker GE.

Through the SmartSDK Platform Manager users will be able to control the status of their Smart applications, and re-configure them and eventually re-deploy them (e.g. on different hosting solution from FIWARE Lab, including commercial FIWARE providers or alternative commercial cloud providers).

Beyond the core of the system, SmartSDK Recipes and SmartSDK Data Models, SmartSDK will provide new Enablers and enhanced ones. These contributions will be derived also from the experience developing the trial applications, as presented in the following section.

1.3. Enhanced IoT and Data Management FIWARE Architecture

As mentioned above and detailed in the following sections, SmartSDK contributes to the FIWARE Community and number of new components dealing with IoT enablement and Data Management. In this section, we shortly present how this component integrates with the current FIWARE Reference Architecture for IoT and Data Management. Figure 3 shows the interaction between the different components (dark blue for the novel ones introduced by SmartSDK and light blue for the original FIWARE components).

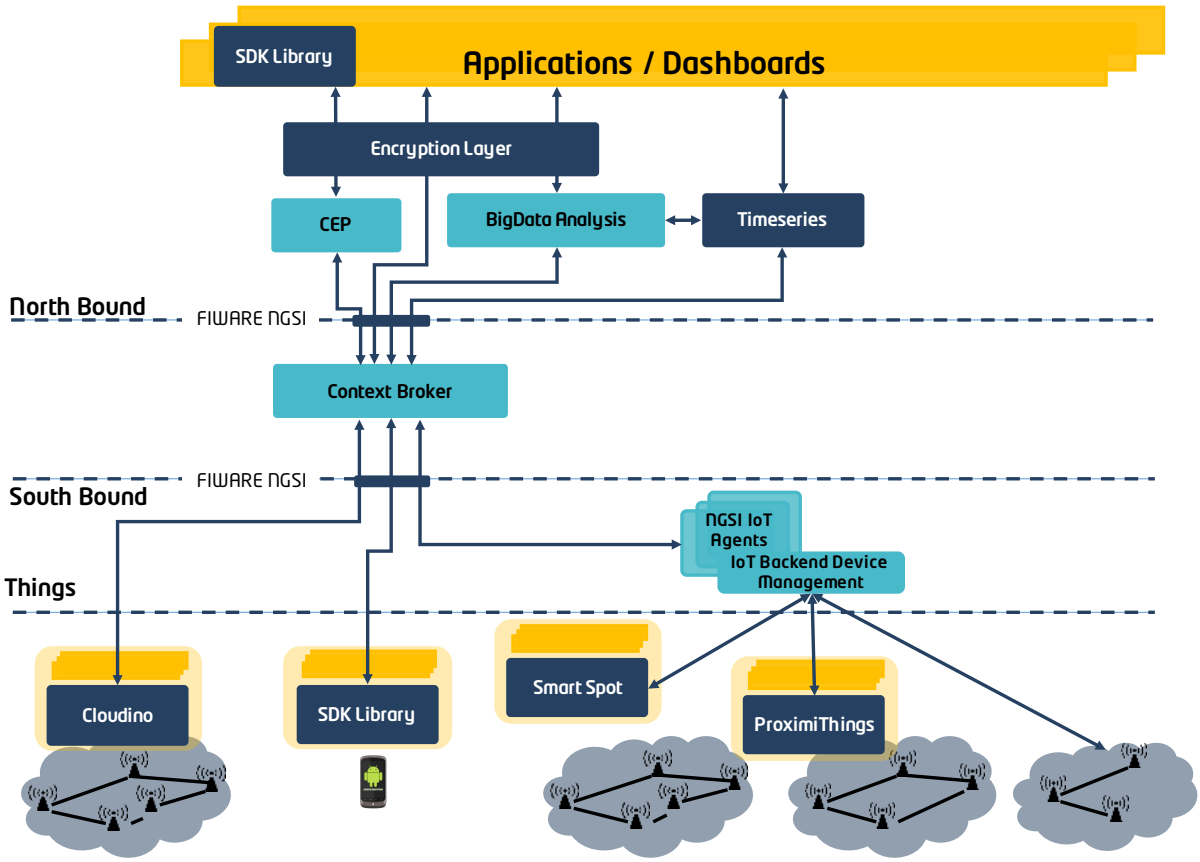

Figure 3: SmartSDK revised reference architecture for IoT and Data Intensive applications.

At the Internet of Things enablement layer, SmartSDK introduces:

- Cloudino a NGSI native hardware module that integrates with Arduino-like boards and allows to manage sensor data collection and publication to Context Broker, without the need of intermediary IoT agents.

- Smart Spot, a Physical Web solution integrated into FIWARE through the LwM2M IoT Agent.
- ProximiThings, a FIWARE-enabled framework for the incorporation of proxemic interaction capabilities in IoT systems, integrated into FIWARE through the MQTT IoT Agent.

At the Data Management layer, SmartSDK introduces:

- A set of NGSI SDK Libraries that allows (mobile) applications to consume and produce NGSI data (and accordingly the SDK Library is presented in the figure in both roles).
- A Timeseries database implementing NGSIv2 APIs, that replaces STH Comet, providing native support for Timeseries database in FIWARE and native integration with timeseries data visualization tools such as Grafana.
- An Encryption Layer that support the fine-grained encryption and decryption of NGSI attributes, thus ensuring data protection and preservation for NGSI data sets.

2. APPLICATIONS

During the period November 2016 - May 2017, different components and proofs of concept have been developed for the three scenarios: smart city, smart security and smart health. This section presents the progress of those applications domains and their planned achievements for the period June 2017-November 2017.

2.1. Smart city

The application developed in the Smart City domain (called Green Route) will focus on supporting the citizen mobility in high-polluted cities, like Mexico City, with the aim of improving the life quality of citizens and fostering environmental friendly behaviours by citizens. The end-user perspective is shortly summarised below.

The objective of application is to help users finding the best route to reach a destination, considering the user profile (such as health conditions), and the user preferences, such as transport type. The application proposes the ideal route for the user, avoiding streets with high levels of pollution, traffic jams or pollens, etc., allowing for instance, to obtain the preferred routes for people with respiratory diseases.

2.1.1. Covered Epics (November 2016 - May 2017)

To support the mentioned smart city scenario, a set of components have been developed from Sprint 0, September 2016 to Sprint 2.1, May 2017. Those components developed are mentioned in each epic defined in the roadmap 1.0.

➔ User profile management

- New data models have been developed to contribute to FIWARE Community: user and disease profile. These data models contain sensible information about app users, therefore this information cannot be sent to the Orion Context Broker by security and privacy issues. Therefore, the data for users and diseases are only stored in regular, local and protected databases. However, in order to facilitate its reuse, the data models have been defined according with best practices by FIWARE Data models.
- New data model for alerts has been also developed to indicate the alerts generated by users in a given location. This model could be used to send alerts related to traffic jam, accidents, weather conditions, high level of pollutants and so on. This information needs to be send to Orion Context Broker because it represents context data useful to app users. In order to consume the consume this data, applications need to be subscribed to the Context Broker to have real time information about the alerts, and, as a consequence, generate user notifications, or trigger other actions.
- A module has been designed and developed to register the user profile. The module contains all the validations to create a new register, such as when the email does not have a valid structure, the user is already registered, the user is already registered, and when a token has been sent to the user's email, and so on.
- A module has been designed and developed to register a respiratory disease. The module allows to a user define his asthma level condition in a friendly way. The user can select if his condition is mild intermittent, mild persistent, moderate persistent, or severe persistent of asthma. In addition, a user can indicate his allergy symptoms, such as nasal congestion, cough, itchy and so on.
- A module has been designed and developed that allows to manager to create new groups, such as "Bikers", "Runner", "Diseases", "Public Transport" and "Private

vehicles". A manager can create, edit, update and delete groups. Also, the manager can select what types of notifications will receive a group. For instance, the manager can create the group of Bikers and he indicates that this group will receive notifications of weather conditions, pollutants and road occurrences.

- A module has been designed and developed that allows the definition of rules for each type of users and roles. This is because different roles (user, admin, super admin, public transport admin, etc.) have different functionalities and permissions for accessing different type of data.

➔ Vehicle profile management

- The team designed and implemented a module to register the private vehicles of a user. The module allows users to register the main features of their vehicles, for instance, they can enter the data of his car, or motorcycle, such as plate number, brand name, model name, fuel consumption, fuel type, also the user can indicate if this vehicle is his favourite or default car. Furthermore, this module considers all the validations to user, such as when a data is required, when is duplicated the register or when you should confirm that you want to delete a type of transport.
- From the data models developed in the "Route Management Epic", a module was designed and deployment to register public transport vehicles. The module allow to public transport manager to register the schedules, stops, frequencies, agencies, route name, departure times, arrival times, and so on attributes related to share static public transport data. Moreover, these modules allow editing, updating, and deleting a type of public transport. Furthermore, this module considers all the validations to user, such as when a data is required, when is duplicated the register or when you should confirm that you want to delete a type of transport.
- The vehicle model from FIWARE has been reused to deploy the features of vehicles.

➔ Private transport information management

- A process for generating the KML (Keyhole Markup Language) format files of the private transport geographical information has been defined. These files were generated for the 22 routes from the ITESM private transport, where information related to the bus stops as well as the departure and arrival schedules was also incorporated.

➔ Maps management

- A module has been developed to represent the result from route engine on a map. The user can see graphically their waypoints and the route to arrive at each point.
- Module to represent Mexico City air quality data on a map. Using a web app, the user can access all air quality data available in Mexico City and all the information that they collect in real time. This means that the information that user has access is continually updated.
- Regarding each air quality sensor available, the web app calculates and represents the Air Quality Index (AQI) for each pollutant that the sensor has information about. Regarding the sensors representation on the map, each one is represented by a marker with the correspondent AQI color regarding the pollutant with the highest AQI for that sensor.

➔ Route management

- Route engine module that allows calculating the best route between two points using public transportation. To do that the route engine allows specifying several parameters regarding the route, like departure and arrival times, maximum walking distance and

others.

- Regarding the calculated route, the routing engine returns the possible itineraries found and all the steps that the user needs to follow in each one. These steps refer to transportation changes (the calculated route can involve various transport types and services), line changes and others. It also returns the route's polygon in a polyline format and the total distance travelled.
- Another routing engine was integrated to obtain the best route between two points. This engine provides ordered waypoints that the route must pass through (for instance, with Start: A; End: B and Waypoints E, C, D, it can compute the best route through the points in the following order: A - E - C - D - B). Moreover, the routing engine can use other types of data, such as traffic or pollution, to calculate the fastest way to visit all the way points. The engine returns the route's polygon in a polyline format, the instructions to follow the route and the total distance travelled.
- New data models were developed based on the the General Transit Feed Specification (GTFS) that defines a common format for public transportation schedules and associated geographic information. The models defined are agency, route, stop and trip. All these models allow us to model a public transport agency, route, and stop. Furthermore, the trip model can be used to find the users' favourite public transport trip and all the data related to it.
- Three data models were developed related to public transportation: Service Networks, Service Alerts and Vehicle Position. The first one is used to map a public transportation service network based on the General Transit Feed Specification (GTFS) that defines the common format for public transportation schedules and associated information. The second one, is used to map alerts related to all the parts that make the public transportation service network. The last one, is used map the public transportation vehicles position.
- A module has been developed to show the public buses from ITESM on map. The module process KML files to obtain the coordinates on the road.

➔ Data context management

- A crawler was developed to obtain the air quality data from Air Quality Mexico City web site (http://www.aire.cdmx.gob.mx). The data are send to Orion Context Broker. Each hour it is possible to obtain the pollutants data of the previous hour.
- The air quality model from FIWARE have been reused to deploy the pollutants data.

➔ IoT Management

- Developed a service for study the user's interactions with the device. This module will provide to the Smart Spot admin interesting information about the users that are interacting with the device.
- Smart Poi Data Model: this data model represents the area of interaction issued by a Smart Poi. This area can be formed for one or mode Smart Spots.
- Smart Spot Data Model: this data model represents the devices that make up a Smart Poi. This device can be managed through the IoTAgent.

2.1.2. Planned Epics (June 2017 - November 2017)

To support the mentioned smart city scenario, a set of components need to be developed from June 2017 to November 2017.

➔ User profile management

- Rules management: this component allow us to define dynamic rules, according the role and the profile of the users. In this way, a manager will create or delete a rule directly in the front-end of the application.
- Configure health notifications: this component allows to user configure health notifications according his groups and the user could decide the periods to save his notifications.
- Fix minor bugs: the current manager view allows to create groups, however there are some bugs that must be fixed to

➔ Vehicle profile management

- Schedule public transport module: this component allows to register the information of public transport of the Mexico City.

➔ Maps management

- Public transport location: integrate the real time public transport location in the maps.
- Pollens maps: display information about pollen concentrations.
- Fix minor issues: validate data sources for some pollutants that seem to be displaying incorrect values.

➔ Route management

- Congestion and traffic information: display real-time traffic information on a map.
- Best routes: integrate traffic and pollution data in the best routes calculation.

➔ Data context management

- Public transport management: this component allow to obtain the position from a public transport vehicle and this information could be available in the smart city application for all the users.
- Traffic information: this component allow to obtain real traffic data from public transport. This information could be shown for all the users.

➔ IoT Management

- Fix LwM2M protocol minor bugs: this enabler contains minor protocol bug that are fixed in the official repository.
- Do single connection LwM2M to multiple entity NGSI data model: this improve will allow representing and managing devices in multiple NGSI entities through Orion Context Broker.
- Improve communications for saving data and communication cost: with the finality of saving cost, the communications between the IoT Agent and devices will be optimized.

➔ Alerts management

- Following, the components to be developed in next 6 months are listed below.
- Define and implement the alerts database schema: build the schema that will allow developers to save a log about the alerts generated by users registered in the alert application.
- Define the “observed event” catalog: establish the types of events that will be incorporated into each of the types of alerts for the alert application.
- Design the alert application. Activity related to the design of mockups for the alert

application.

- Develop the alert application: task associated with the development of the alert application according to the mockups defined.

2.1.3. Key used FIWARE Enablers

- Identity manager - KeyRock
- Orion Context Broker
- Backend Device Management – IDAS
- Cygnus
- Cosmos Big-Data Analytics

2.1.4. Contributions to the FIWARE Community

The main contributions of the FIWARE Community from these activities are:

- Creation of a new data model for alerts generated by a user in a given location. This model could be useful to send alerts to Orion Context Broker related many scenarios: traffic, pollution, suspicious activities, foods, and others. A web service could be subscribed to OCB to obtain data generated by users and generated notifications or trigger other actions.
- Creation a news data models for trips. These models including all properties which can be used to find the user's favourite public transport trip and all data related to it. The data models are based on the General Transit Feed Specification (GTFS) that defines a common format for public transportation schedules and associated geographic information.
- Publication of real air quality data from Mexico City through of Orion Context Broker. The data contains the measures of pollutants of carbon monoxide, ozone, nitrogen dioxide, sulfur dioxide and particles pollution. The measures are updated each hour. Any SMEs, start-ups and entrepreneurs could obtain the air quality data to develop highly innovative services and applications.

2.2. Smart Security

The Smart Security application will focus on detecting and analyzing security risk such as, theft, access controls, people detection, fights, crowd analysis, etc., through the combination of video cameras and mobile sensors, in both, indoor and outdoor scenarios, for instance, parking lots and buildings. The goal is to support the end-user (i.e. security guard) to prevent risk situation and consequently improve the quality of life of citizens.

2.2.1. Covered Epics (November 2016 - May 2017)

To support the mentioned smart security scenario, a set of components have been development from Sprint 0, September 2016 to Sprint 2.1, May 2017. Those components developed are mentioned in each epic defined in the roadmap 1.0.

- Data management
  - Implementation of one video streaming view module from a single camera. The objective of this implementation was to define the video information, resolution, format, etc.
  - Implementation of multiple video streaming views from multiple cameras. The objective of this implementation was to evaluate the behavior of the visualization in terms of delay, bandwidth, number of cameras, etc.

- Design and implementation of a storage module from one video streaming. This module will let us retrieve the video to consult the events detected. Besides, this module is necessary for later stages of the application.
- New data model for video surveillance scenarios have been developed. This model could be used to send alerts related to security risk involving people and vehicles. The objective of this data model is to define all the data that will send to Orion Context Broker. Then, other modules of the application could consult this information to generate notifications for a user or even to trigger other actions.
- A GUI based on Web2Py.

➔ Pattern recognition

- First wrapping approach to implement algorithms based on OpenCV and Kurento.

2.2.2. Planned Epics (June 2017 - November 2017)

To support the mentioned smart security scenario, a set of components need to be developed from June 2017 to November 2017.

➔ Data management

- Database implementation required to store both, video alerts and smartphone data.
- Selection of the map format to be used in order to visualize the different areas to be monitored in the campus.
- Design of a software module to make online queries about whether a registered smartphone is in the campus or not.
- Design of a software module to make historical queries about whether a registered smartphone was on campus or not in a specific period.
- Design and implementation of notifications to alert the user when a vehicle is parked incorrectly.
- Development of a software module, based on the data from smartphone, capable of identify persons driving at an unauthorized speed on campus.
- Design and implementation of notifications to alert the user when a vehicle goes in the wrong direction.
- Design and implementation of notifications to alert the user when a vehicle goes in the wrong direction.

➔ Pattern recognition

- Test of modules integration to evaluate the communication between them, the data transmission to the cloud as well as the implemented algorithms performance.
- Development of a software module (Kurento filter) capable of detect vehicles on a video stream.
- Development of a software module (Kurento filter) capable of verify the identity of a person in a video stream in indoor environments.
- Creation of a database of people in outdoor environments to later test people detection/recognition algorithms.
- Creation of a database of vehicles to later test vehicles detection/classification algorithms.

- Development of a software module (Kurento filter) capable of detect persons in a video stream in outdoor environments.
- Development of a software module (Kurento filter) capable of detect vehicles parked incorrectly.
- Development of a software module, based on the data from smartphone, capable of detect vehicles going on the wrong direction within the parking lot.
- Development of a software module (Kurento filter) capable of recognize persons in a video stream in outdoor environments.
- Creation of a database of people behaviours to later test the detection/recognition algorithms of activities.

➔ Design and implementation of databases for the application

- Design of the databases based on the current data models
- Database implementation
- Design and implementation of a component to determine when a registered smartphone is inside a specific Campus.

2.2.3. Key used FIWARE Enablers

➔ Orion Context Broker

➔ Kurento

➔ Cygnus

2.2.4. Contributions to the FIWARE Community

The main contributions to the FIWARE Community are:

➔ Creation of a new data model for video surveillance. This model would be useful to send information (i.e. alerts) to the Orion Context Broker related with security risk.

➔ Visualization and notification of events in areas of interest to detect vandalism or even to prevent incidents that may trigger tragic events.

2.3. Smart Health

Sensors in mobile, wearable, and environmental devices are becoming ubiquitous, allowing the inference of activities and behaviors associated with health by facilitating the collection of daily-life data. The Smart Health task focuses on creating a specific set of components to support the development of healthcare mobile applications based on analyzing data gathered from sensors in mobile and wearable devices. In particularly, we will be presenting an application developed in the context of mobility assessment to address the harmonization and management of mobile sensing data using and extending the FIWARE platform.

2.3.1. Covered Epics (November 2016 - May 2017)

A set of components have been under development from Sprint 0.1 (November 2016) and will continue until Sprint 2.1 (May 2017). Next, we present a brief description of the components developed and planned for this period:

- Upload Information
  - A data model focused on representing sensor data gathered from smartphones, wearable devices, and health data inferred from the sensor data.
  - A mobile phone application implemented on Android O.S. to collect sensor data when the individual performs certain controlled activities. The data gathered during the physical tests is stored in the mobile device and then transferred to the cloud to make it persistent by storing it using the Orion Context Broker and Cosmos GE.
- Dashboard monitor
  - A widget development, called "oHealthCxt-Browser" that aims to facilitate the setup of a file-manager to seek physical-test performances, parameter of interest, among other relevant information in the context of health applications.
  - A software component implemented to retrieve historical data from the Cosmos GE (by reading and formatting HDFS documents into NGSI schemas), which can be used over Wirecloud widgets.
- Data analyses
  - Walking speed. An initial approach for detecting events using accelerometer data recorded while performing a controlled physical tests (e.g., TUG or Strength).
  - Overall Stability Index (SI). A first version of a mechanism to calculate the SI which represents the variance of the platform displacement in degrees, from level, in all motions during a test.

2.3.2. Planned Epics (June 2017 - November 2017)

To support the mentioned smart health scenario, a set of components need to be developed from June 2017 to November 2017.

- Data analyses
- Subject study. A first subject study is conducted to test both: FIWARE's integrated component and application development.
- Control Access
- The health data sensing management allows access to data only to authorized users. In addition, private data is stored independently in the public Cosmos GE.
- Data integration
- The mechanisms for analysing data are integrated on the the application architecture, thus, results can be retrieved from the Wirecloud GE.
- Wearable devices integration
- Development of a project's proof of concept for practical testing the flexibility of the data-model, over wearable sensing devices such as smartwatch.

2.3.3. Key used FIWARE Enablers

- Cygnus
- Identity manager - KeyRock
- Orion Context Broker
- Wirecloud

2.3.4. Contributions to the FIWARE Community

The contributions focus on three components:

- Data model. It consists of a data structures for mobile sensing applications designed to bring a meaningful interpretation of data by reducing its complexity through a standardized set of schemas. It aims to facilitate writing applications that can address mobile sensor data for health. The data model can be extended to incorporate new sensors and health parameters of interest.
- Bifrost. A software implementation developed on python using the Flink framework to connect widgets with historical data components. Brifost serves as an intermediary that transports query parameters from the Wirecloud GE to the Cosmos GE through an interpreter (analogically illustrated as a bridge). Once data is retrieved from the Cosmos GE, it is sent to the bridge, and then the data is parsed into NGSI format and sent back to the Wirecloud GE; so it can be displayed over respective widgets.
- oHealth-Browser. A software widget developed to handle three different modules: Participants, Physical test, and Parameters of Interest. Altogether, functions are built into a single component which can be extended in order to cover a wider range of services. The GUI is personalized based on basic information, such as the IP address of the remote service (e.g., Cosmos or a third party server), Module of service (i.e., Participants, PhysicalTest, or ParameterOfInteres), and Attributes to retrieve, among others.

3. PLATFORM

With the aim of contributing to FIWARE OS Community and supporting the development of the scenarios depicted in the previous section, SmartSDK will work on the development of new functionalities for existing GEs and novel ones. The functionalities are grouped around three main groups:

- The platform: the glue that help the provisioning and automation of smart applications.
- The data management: the enhanced set of functionalities that increase the capacity of FIWARE Data Management platform.
- The IoT management: the enhanced set of functionalities that increase the capacity of FIWARE IoT platform.

3.1. Platform

3.1.1. Covered Epics (November 2016 - May 2017)

To support the SmartSDK platform, a set of components have been development from Sprint 0, September 2016 to Sprint 2.1, May 2017. Those components developed are mentioned in each epic defined in the roadmap 1.0. The SmartSDK platform Manager is based on rancher, an Open Source project released under the Apache2 License.

- Orchestrator for SmartSDK services.
  - The orchestrator activities were focused on running a set of Docker Swarm nodes to compose a swarm cluster. In particular, the orchestrator has a few restriction on the type of network connection. In order to fully use the capabilities of the SmartSDK manager, a networking guide documenting different possibilities for a working setup was added. Additional care was used for the correct networking setup involving the FIWARE Lab.
- Application catalogue for SmartSDK services.
  - The Platform Manager offer a catalogue of deployable services based on Docker Compose files version 2. The baseline for the SmartSDK recipes was based on Docker Compose version 3, that is not backward compatible. It is however possible to launch services described by Docker Compose version 3 while waiting for the full upstream support. An extensive documentation was added to explain an alternative way to deploy the SmartSDK recipes.
- Generic Enablers Recipes
  - Development of recipes made of shell scripts and Docker Compose files (version 3) to support the deployment of Orion, Comet and Cygnus Generic Enablers, enabling the application of the Scalability, High Availability and Co-location patterns using Swarmkit.
- User management integrated with FIWARE Lab for SmartSDK platform.
  - By using a custom build of rancher is possible to use the Oauth authentication supplied by the FIWARE Lab. The use of the FIWARE Lab Oauth endpoint simplifies the user management on the Platform and offers an integrated user experience.
  - We tested also an user interface extension made by ICCLAB of the Zurich University of Applied Sciences in order to ease the selection of the FIWARE Lab parameters

needed in order to create a Docker Swarm cluster on the user project hosted by the FIWARE Lab.

- Platform manager running on FIWARE Lab.
  - The Platform manager can be deployed on a FIWARE Lab node, in a custom project. It can be run autonomously by each user or used as a service. When used as a service the user is required to create is Docker Swarm nodes on a proper Cloud provider. The Cloud provider can be also the FIWARE Lab. Some care is needed in the networking setup, considering the fact that in some node (notably the Spain2 node, that hosts the keystone authentication and authorization service) has a network with the default MTU of 1400 bytes instead of the usual 1500. An extensive documentation was made in order to offer a guidance for the users.

3.1.2. Planned Epics (June 2017 - November 2017)

- Generic Enablers Recipes
  - Development of recipes (in the form of shell scripts and Docker Compose version 3 files) to enable the applicability of the Scalability, High Availability and Co-location patterns to the deployment of the CKAN and IDAS Generic Enablers.
- User Interface for Docker Compose v3
  - The current user interface does not support the advanced features offered by Docker Compose format v3. It only support a basic view of the running Docker instances. The http://portainer.io/ project should support this in the near future.
- User management integrated with FIWARE Lab for SmartSDK platform
  - We plan to automate the build and the documentation of the custom Docker image used for the FIWARE Lab Oauth integration.
- Overlay network
  - To overcome the limitation of rancher networking we plan to explore and document other VPN alternatives: for example Freelan, VDE2 and GVPE.
- Persistent storage configuration
  - There are also few discrepancies regarding the support of shared storage in Docker Composev2 and v3 to be further investigated. The optional persistent storage configuration can be enhanced, for example by using a separate storage network.
- SmartSDK Platform Manager in HA
  - Good production practices for cloud distributed application encourage the setup of highly available services. Currently our testing environment do not have the resources to offer HA, but we plan to overcome this issue in the near future.

3.1.3. Contributions to the FIWARE Community

- A container management solution integrated into FIWARE Lab
- A set of recipes allowing production grade deployment of FIWARE GE using Docker

3.2. Data management

The Data Management activities will focus on contributions that will enrich the FIWARE Data Management chapter and improve interoperability of NGSI (w.r.t. devices and open data models). Activities have been defined and prioritized taking into account SmartSDK applications' requirements.

3.2.1. Covered Epics (November 2016 - May 2017)

To support the SmartSDK platform, a set of components have been development from Sprint 0, September 2016 to Sprint 2.1, May 2017. Those components developed are mentioned in each epic defined in the roadmap 1.0.

- Time Series for NGSI
  - Perform a SOTA analysis of the modern available timeseries databases.
  - Develop a flexible benchmarking testbed for timeseries databases (tsdb)
  - Develop and validate first version of custom translators (NGSI to custom TSDB and vice versa)
- NGSI Encryption Layer
  - Develop the initial part of a SOTA analysis of the available data encryption algorithms, specific for attribute encryption.
- SDK Library for NGSI
  - A component has been developed to read the smartphone data in real time. The data to be used depends on the application domain.
  - Classes were developed to create data types for each one of the data obtained from smartphone.
  - Classes were created to configure the connection of the mobile device to the Context Broker.
  - Classes were developed to send the data updates to the Context Broker via NGSI specification.
  - Classes were defined and implemented to obtain response about the update from the Context Broker.

3.2.2. Planned Epics (June 2017 - November 2017)

- Time Series for NGSI
  - Develop and validate first version of custom translators (NGSI to custom TSDB and vice versa)
  - Enable visualization of historical data using tools like Grafana.
- NGSI Encryption Layer
  - Improve the SOTA analysis of the available attribute encryption algorithms.
  - Compare performance of different attribute encryption algorithms.
  - Develop a first version of the NGSI-AE.
  - Test the encryption / decryption to guarantee NGSI data integrity.
- SDK Library for NGSI
  - Develop a first version of the JavaScripy Library based on current Android mobile application.

3.2.3. Contributions to the FIWARE Community

The main contributions of the FIWARE Community from these activities are:

- A native NGSIv2 Timeseris database, including support for data visualization.
- A facility to support data encryption on top of NGSI providers.
- A SDK to support mobile devices interaction with NGSIv2 providers and consumers.

3.3. IoT Management

The IoT Management activities will focus on contributions that will enrich the FIWARE-ready IoT ecosystem, in particular with the aim of improving Cloudino framework. Activities have been defined and prioritised taking into account SmartSDK applications' requirements.

3.3.1. Covered Epics (November 2016 - May 2017)

To support the SmartSDK platform, a set of components have been development from Sprint 0, September 2016 to Sprint 2.1, May 2017. Those components developed are mentioned in each epic defined in the roadmap 1.0.

- CLOUDINO
  - A Cloudino WIFI Connector has been developed that enable developers connect the Cloudino to the FIWARE Context Broker without an IoT-Agent, using the simple Cloudino Configuration Web Interface.
  - A Cloudino WiFi Connector has been developed to permit the connection of Cloudino to the FIWARE using MQTT IoT-Agent, using the simple Cloudino Configuration Web Interface.
  - Currently, a Cloudino Connector is being developed to permit the connection of Cloudino to FIWARE using Cloudino Cloud Service, using the simple Cloudino Configuration Web Interface.
- SMART SPOT
  - BLE Physical Web capability in Smart Spots. The device is able to broadcast the desired URL via Bluetooth using the google protocol eddystone URL and giving to the device de capability of send physical webs to the users.
  - Device URL Manager. This service provide to the user the capability of admin the URL broadcasted by the devices seamless, it also provide some statistics like the number of interactions. Now days this service is used by an API REST but we are working in a user interface to facilitate the user's interactions.
  - This tool is used for manage the physical web URL of any device by software.
  - A smartphone will detect a eddystone URL advertisement with a fixed device URL that point to the Device URL Manager, then the Device URL Manager will redirect the request to the real URL.
  - GSM Module. Thanks to this module, the Smart Spot has the capability of sending data using a micro sim with data. This module also has manage features like connection handle disconnections, use the cheaper connection available, reconnect if is possible and improve the power saving. This module also provide to the Smart Spot the capability of by localized via GPS, this feature is interesting when we are trying to improve mobile entities.

3.3.2. Planned Epics (June 2017 - November 2017)

- CLOUDINO

- Cloudino Cloud Service. For this project, we need to be develop a specific cloud for Cloudino (Cloudino.io Cloud Service) to enable developers connect with different open/private clouds. In this context, the Cloudino Cloud will be an intermediate among the data capture by the electronic device in the specific cloud platforms.
- Create a version of Cloud as a Generic Enabler. The current version of Cloudino need to be properly packaged as Docker container to be aligned with Generic Enabler best practices

➔ SMART SPOT

- Sensor integration. For this project will be integrated several sensors, like temperature, humidity, air quality and accelerometer, some of this sensors have to be defined in the OMA protocol and conveniently parsed to NGSI entities.
- Improve IoT Agent. Single connection-multiple entity LwM2M IoTAgent we have to improve this enabler for saving data and communications cost.

3.3.3. Contributions to the FIWARE Community

➔ Smart Spot Data Model. For the correct integration of the Smart Spot in the FIWARE ecosystem, we have been working in a NGSI data models approved and certificated by FIWARE. https://github.com/Fiware/dataModels/tree/master/PointOfInteraction

➔ IoT Agent Fixes. For the correct behavior of the device, some corrections were made in this FIWARE enabler, this correction were about the way of managing the LwM2M protocol. Corrections were merged in the master branch by the Generic Enabler owner.

4. CONCLUSIONS

This document presents the SmartSDK R&D roadmap covering the period from June 2017 to November 2017 with the aim to highlight the planned contributions to the FIWARE Open Source Community. The next release of this document (November 2017) will refine the Epics and priorities according to the evolution on the project and requirements in the application scenarios.

In this deliverable, we quickly overviewed the following items for each of the application scenarios covered in Work Package 2:

- The epics covered in the R&D phase to support such a user perspective.
- The usage of FIWARE Enablers.
- The contributions to FIWARE.
- The priority by November 2017

As regards the Platform (Work Package 3), the deliverable highlights:

- The epics covered in the R&D phase to implement the new Enablers
- The planned contributions to FIWARE.
- The priority by November 2017 of such contributions.

REFERENCES

[1] SmartSDK Consortium. Description of Action. July 2016.

[2] SmartSDK Consortium. D5.1: Quality Assurance Guidelines. November 2016.

[3] Dean Leffingwell. 2011. Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise (1st ed.). Addison-Wesley Professional

APPENDIX A SMARTSDK AGILE METHODOLOGY

To drive the creation of tangible project outcomes, SmartSDK adopts an agile iterative methodology. The methodology is driven by the applications in the Smart City, Smart Healthcare and Smart Security domains and the applications' validation into trials.

The analysis of the state-of-the-art and of the applications steers the Research and Development of planned outcomes (SmartSDK Platform and the Applications to be trialled). The R&D outcomes are then validated through the trialling of the Applications with real-users. Validated outcomes are then adopted into Technology transfer activities (i.e. trainings and contributions to standardization bodies). Feedbacks and outputs of the process feed the next cycle of activities.

The whole project foresees 2 major cycles, each of which will include two minor iterations. The first set of outcomes will be trialled in Mexico at the end of the first cycle (M12), while second stage trials will leverage on the European FIWARE ecosystem (M24).

The methodology will be implemented using the Lean Agile approach by Leffingwell [3] adopted by FIWARE Agile Development Methodology[3]; this will allow a continuous monitoring of project progress in term of outcomes and exploitable results. The process is depicted in Figure 4.

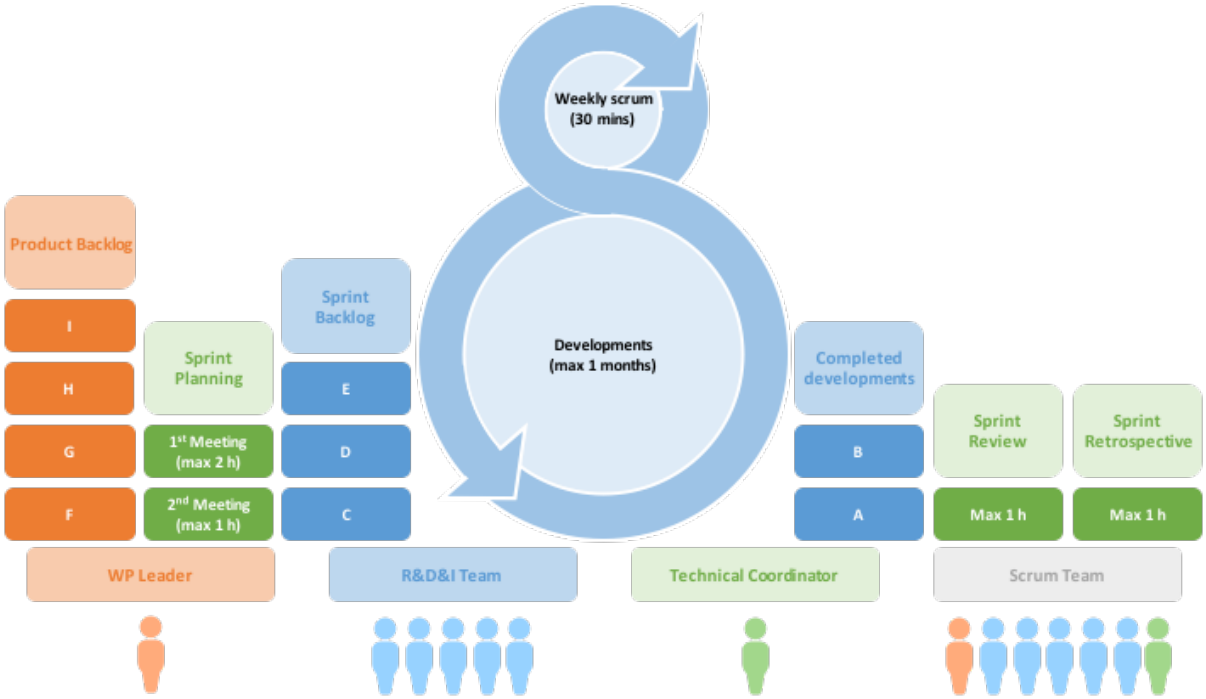

Figure 4. SmartSDK Agile Development Process.

The Agile meeting are planned as follows:

- Sprint planning 1st Monday of the month 16:30 – 17:30
- If needed also 1st Tuesday of the month 16:30 – 17:30
- Weekly scrum all Mondays 17:00-17:30
- Sprint review: last Thursday of the Month 16:30 – 17:30

[3] http://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/FIWARE_Agile_Development_Methodology

The first Sprint is only for design purposes and last longer (from the beginning of September to the end of October), then the Sprint are monthly. Every third Sprint includes a Release.

A.1 The Process

1. The application scenarios' requirements will be described in a plain-text document. From this description, Application owners will derive main Epics.
2. Epics will be assigned to a Component (an Application, a Data Model, or an Enabler; part of WP2 or WP3).
3. Component Owners will define a set of Features for each Epic.
4. Features will be decomposed in User Stories.
5. User stories will be then planned for specific Sprints during the Sprint Planning, and validated within Sprint Reviews that will include demonstration of achieved results.
6. Each User Story should be refined and detailed before being assigned to a Sprint.

The methodology will be implemented leveraging on the FIWARE JIRA management tool.

A.1.1 Roles and responsibilities

- Product Owner
  - Who: All the task leaders
  - What:
    - Define user stories
    - Define features
    - Define epics
    - Prioritize the backlog
- Developer
  - Who: All the people active in the "Development" / "Delivery"
  - What:
    - Refine and Implement user stories
    - Document progresses
- Scrum Master
  - Who: Federico/Tomas (Martel)
  - What:
    - Help to prepare the sprints planning and sprints review sessions
    - Check status of activities
    - Facilitate collaboration
    - Remove obstacles
- Scrum Team
  - Who: Developers, Product Owners and Scrum Master
  - What:

- Assign/Re-Assign stories to components (in collaboration with other product owners)
- Sprint planning
- Sprint review