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

D2.2: Reference architectures for data-intensive and IoT-based Smart City, Smart Healthcare and Smart Security applications

Revision: v.1.0

Work package	WP 2
Task	Task 2.2
Due date	30/06/2017
Submission date	30/06/2017
Deliverable lead	ITESM
Version	1.0
Authors	Nestor Velasco-Bermeo (ITESM), Joanna Alvarado (ITESM), Miguel González (ITESM), Ariel García (ITESM), Blanca Vázquez (INFOTEC), Hugo Estrada (INFOTEC), Netzahualcóyotl Hernández (CICESE), Luis Valentin (INAOE), Alma Rios (INAOE), Miguel Palacios (INAOE)
Reviewers	Federico M. Facca (MARTEL); Tomas Aliaga (MARTEL)

Abstract	The following report presents the first version of SmartSDK Applications architectures and their derived pattern.
Keywords	Reference Architectures, FIWARE, Smart City, Smart Security, Smart Health

Document Revision History

Version	Date	Description of change	List of contributor(s)
V1.0	19/07/2017	Final version implementing reviewers comments	Nestor Velasco-Bermeo (ITESM), Ariel García (ITESM)
V0.9	15/07/2017	Review	Federico M. Facca (MARTEL); Tomas Aliaga (MARTEL)
V0.8	14/06/2017	SmartCity scenario architecture revised	Blanca Vázquez (INFOTEC)
V0.7	14/06/2017	Healthcare scenario revised	Netzahualcóyotl Hernández (CICESE)
V0.6	13/06/2017	Security scenario revised	Miguel Palacios (INAOE)
V0.5	13/06/2017	Editing of the architecture diagrams	Nestor Velasco-Bermeo (ITESM), Joanna Alvarado (ITESM), Miguel González (ITESM), Ariel García (ITESM)
V0.4	12/06/2017	SmartCity scenario architecture draft	Blanca Vázquez (INFOTEC), Hugo Estrada (INFOTEC)
V0.3	11/06/2017	Healthcare scenario architecture draft	Netzahualcóyotl Hernández (CICESE)
V0.2	10/06/2017	Security scenario architecture draft	Luis Valentin (INAOE), Alma Rios (INAOE), Miguel Palacios (INAOE)
V0.1	01/05/2017	Table of Content	Federico M. Facca (MARTEL)

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 aims to provide a set of ready to use “recipes” to develop smart applications in the Smart City, Smart Healthcare, and Smart Security domains. Such recipes are based on: components (i.e. Generic Enablers and Specific Enablers), data models (i.e. NGSI formalisation of the data exchanged among components) and reference architectures (i.e. the combination of components and data models to support production grade requirements).

SmartSDK already provided a set of Generic Reference Architectures based on the combination of typical FIWARE architecture patterns and cloud architecture patterns (c.f. D3.1 - SmartSDK Reference Models and Recipes). This deliverable takes on from that general work and the architectures developed in the context of the SmartSDK applications to abstract a set of generalized reference architectures (i.e. FIWARE-based architecture solutions) to cope with typical functionalities provided by smart applications in different domains. The deliverable besides covering the status of the SmartSDK applications architectures, summarizes the Road Map of the different modules developed to solve specific Smart Application needs. Finally, for each application domain covered by SmartSDK, the deliverable presents a set of reference architectures. Such architectures are self-contained functional blocks derived from the actual architecture of the Applications developed in SmartSDK and provide re-usable building blocks to facilitate the development of new Smart Applications.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

API	Application Programming Interface
DNS	Domain Name Server
GE	Generic Enabler
GEri	Generic Enabler reference implementation
HA	High Availability
HTTP	Hypertext Transfer Protocol
IoT	Internet of Things
IP	Internet Protocol
REST	REpresentational State Transfer
TCP	Transmission Control Protocol
Poi	Point of Interest
LWM2M	Lightweight Machine to Machine
MQTT	Message Queue Telemetry Transport
GTFS	General Transit Feed Specification
TUG	Timed Up and Go
HDFS	Hadoop Distributed File System

1 INTRODUCTION

SmartSDK has a strong foundation due to its standardized data models, thus providing not just a structured way of collecting, storing and serving data but also key capabilities such as scalability and reusability. The benefits of such capabilities provide potential developers to adopt not only the data models but also refer to the recipes that will be presented in the following document. The recipes described in the upcoming sections are based on a top-bottom approach. They start from the particular solution that's been developed to attend the specific needs of each domain: Smart City, Smart Healthcare and Smart Security. Each domain's Architectures are described accordingly, providing an insightful view of all the components and their interaction. Furthermore, each Scenario is described to provide the reader a clear picture of its main goals and considerations. While each scenario depicts a particular problem, the reader will be presented with generalized Architecture Patterns that will attend different problems within the scope of each scenario. The goal of such generalized Architecture Patterns is to prove the benefits that SmartSDK provides by being flexible enough to generate a new solution by reusing components that weren't originally conceived to solve such new problem. Finally, each scenario provides a Roadmap of the components that will be created to further improve the provided solution and a description of how the new elements will interact.

1.1 Structure of the deliverable

The Deliverable is structured as follows:

- Section 2 presents the different components (Data/Context Management and IoT Services Enablement) in a southbound-northbound macro architecture. The section also introduces SmartSDK's novel enablers in a reference architecture for both the Data/Context Management and IoT Services chapters. It also provides an insight of how SmartSDK extended FIWARE reference Architectures in derived subsets to be adopted in specific smart scenarios.
- Sections 3, 4 and 6 describes the general considerations and aspect of the Smart City, Smart Health and Smart Security scenarios respectively. Each section presents the general Architecture in which it's being built in SmartSDK. The generalized Architecture patterns are presented to implement new solutions formulated upon the scenario's general architecture. Finally an overall Roadmap is outlined to provide a better understanding of the current and upcoming components created.

1.2 Audience

This deliverable is mainly intended for:

- Developers and Operators interested into deploy FIWARE Smart applications in a production context.
- Developers and Knowledge modellers interested into adopting FIWARE data models or contributing to the initiative.

2 SMARTSDK ARCHITECTURE PATTERNS

Figure 1 presents the different components of the Data/Context Management chapter and of the IoT Service Enablement chapter as a southbound-northbound architecture: this is a macro architecture pattern representing how data can be "lifted" in a interoperable format through a standardized protocol (NGSI) and a set of associated standardized data models (FIWARE Data Models[1]). Developers, depending on their needs, can select to adopt a given subset of components in their application and the relevant data models.

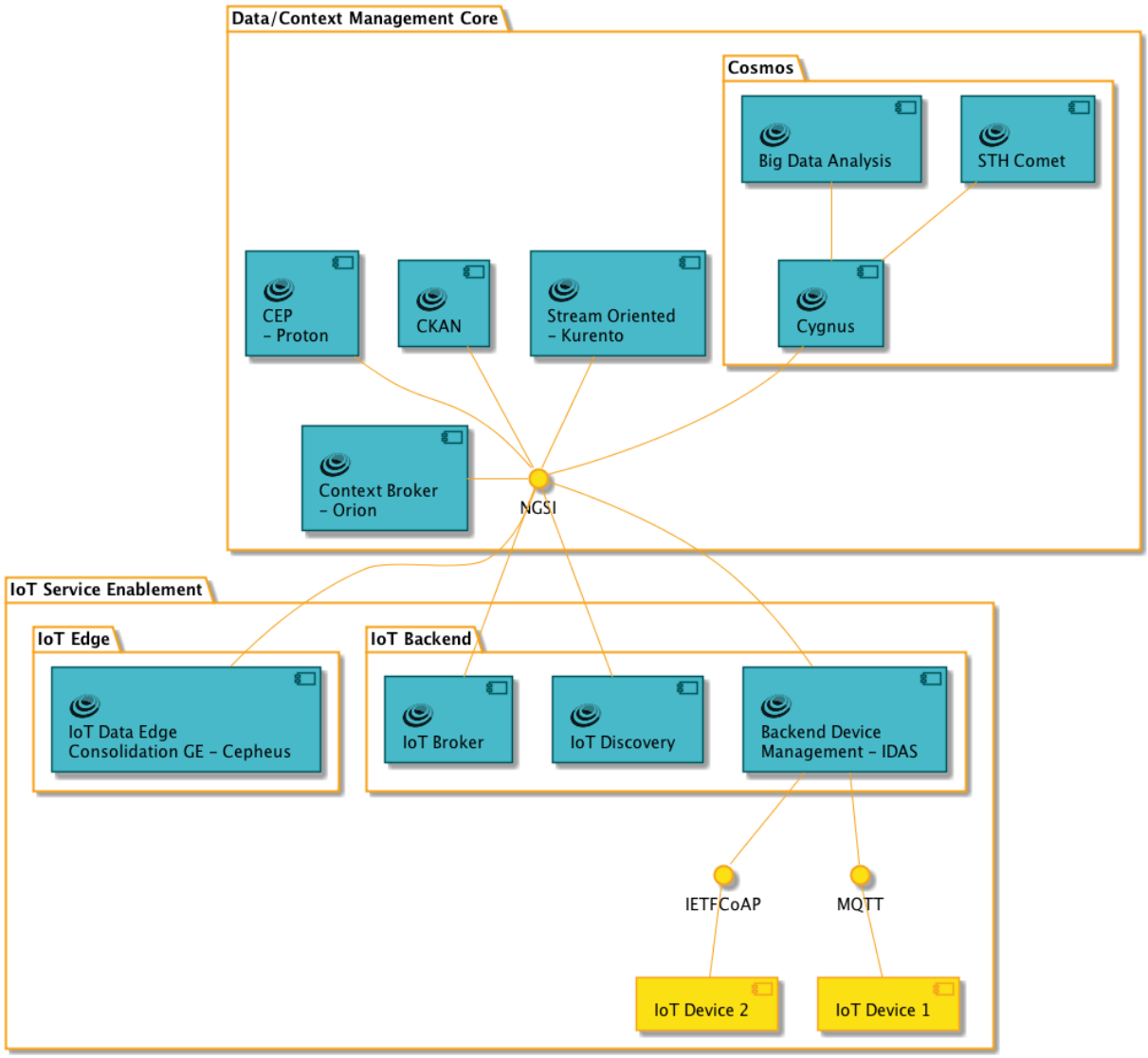

Figure 1. FIWARE Reference Architecture for IoT and Data intensive applications.

Core components of the Data/Context Management[2] chapter include:

- *Context Broker* - Orion, the central element of the Data Context Management architecture, provides a publish/subscribe service for context data.

[1] http://schema.fiware.org
[2] https://catalogue.fiware.org/chapter/datacontext-management

- *Big Data Analysis* - Cosmos, the solution for the analysis of NGSI data sets, is made of different tools including the Short Time Historical data storage (STH Comet) along with the capabilities of integrating different data stores (relational databases, big data filesystems, etc.) thanks to the adaptation capabilities provided by Cygnus.
- *Stream Oriented* - Kurento, an NGSI integrated multimedia processing server.
- CKAN - a repository for Open Data sets.
- *Complex Event Processing* (CEP) - Proton, an event processing tool that identifies patterns over NGSI data and generates response based on identified patterns.

The IoT Services Enablement[3] chapter includes the following core components:

- *Backend Device Management* - IDAS, a set of IoT Agents allow IoT devices to be registered and transforms all collected data into NGSI compliant format.
- *IoT Data Edge Consolidation* GE - Cepheus, a solution for edge processing and aggregation of sensor data NGSI compliant.
- *IoT Broker*, an NGSI middleware that support the aggregation of data from multiple sensors.
- *IoT Discovery*, an NGSI middleware that provides support towards the discovery of context producers by context consumers.

It is worth mentioning that some of these services are meant to run only as Global Instance of FIWARE Lab. This is the case for example of *Big Data Analysis Cosmos*.

SmartSDK, through the development of novel components (c.f. D3.2) extended the above reference architecture to cover additional generic requirements coming from the applications developed in SmartSDK, introducing also Open Hardware-based solutions in the sensing layer.

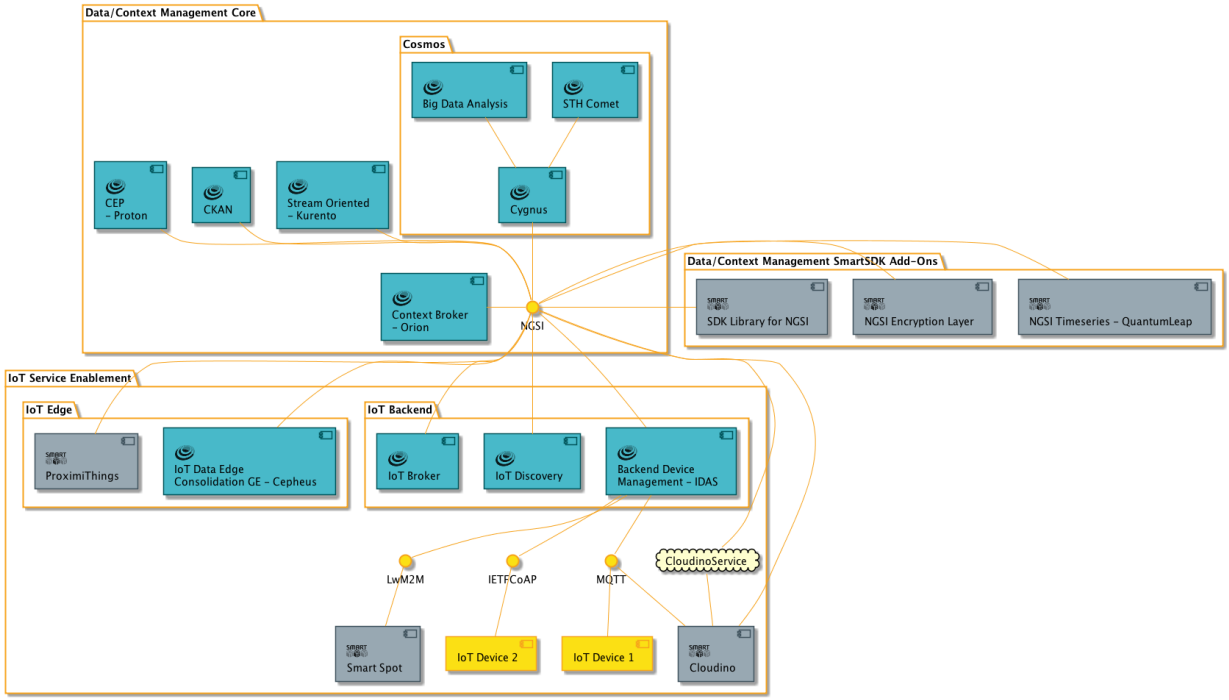

Figure 2. SmartSDK extended FIWARE Reference Architecture for IoT and Data intensive applications.

Figure 2 presents the Northbound-Southbound reference architecture extended to include SmartSDK novel enablers. Also in this case, developers, depending on their needs, can select to adopt a given

[3] https://catalogue.fiware.org/chapter/internet-things-services-enablement

subset of components in their application along with the relevant data models.

Novel components introduced in the Data/Context Management chapter by SmartSDK include:

- *SDK Library for NGSI* - an SDK for mobile devices that allows the developer to update and query context data using NGSI v2.
- *NGSI encryption layer* - a solution that encrypts NGSI data on the client's side to ensure privacy and protection of data stored in any context broker implementation based on the NGSIv2 protocol.
- *NGSI timeseries* - a solution to support natively timeseries processing supporting NGSIv2 protocol (meant to replace STH Comet).

Novel components introduced in the IoT Services Enablement chapter by SmartSDK include:

- *Cloudino* - an Open Hardware IoT solution that natively integrates with FIWARE and allows to natively push sensor data into FIWARE.
- *Smart Spot* - a solution that provides a Physical Web point of interactions natively integrated to FIWARE standards.
- *ProximiThings* - a FIWARE-enabled framework for the incorporation of proxemic interaction capabilities in IoT systems.

To take the most of cloud and microservice technologies, the above reference architectures need to be combined with cloud architecture patterns supporting the deployment in production-like environments of services.

Typical cloud architecture patterns include (cf. D3.1):

- Scalability
- High Availability
- Multi-site pattern
- Co-locate pattern
- Queue-Centric workflow pattern

In D3.1 "*SmartSDK Reference Models and Recipes*" we discussed in detail how such patterns can be combined with FIWARE Reference Architecture. For instance, the Scalability Pattern aims to allow services to easily adapt to a sudden change in the demand of its resources. To apply such pattern to the FIWARE Reference Architecture for IoT and Data intensive applications, we need to understand how the single components of the architecture can scale based on their internals. Considering the Context Broker service, as an example of the process, this boils down to having multiple instances of the context broker service answering in "a coordinated way" to users' requests (cf. Figure 3). This can be attained in the case of web APIs (as the context broker is) using a load balancer (cf. D3.1 for more details). By all means, the picture is actually more complex than that, given that the Context Broker is not a stateless service, and so also data consistency among its back-ends need to be considered as discussed in D3.1.

This version of the deliverable, though, will not yet take into consideration adoption of the cloud patterns and the docker implementation of the recipes. These aspects will be covered in the next version of the deliverable when components will be more mature.

The deliverable will instead focus on deriving subsets (and related extension) of the SmartSDK extended FIWARE Reference Architectures for its adoption in specific smart scenarios, e.g. collecting environmental data, computing transport routing and so on.

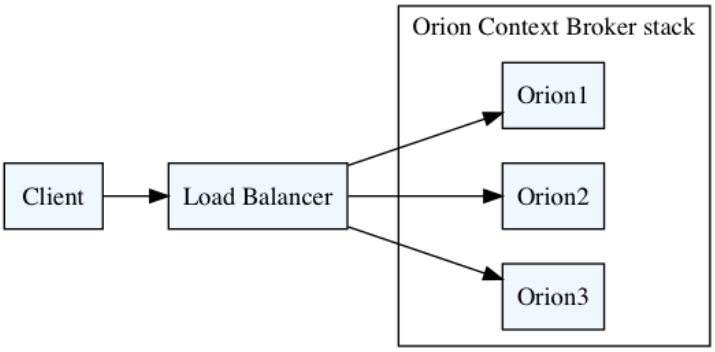

Figure 3. Example of scalability pattern applied to the Context Broker GE.

3 SMART CITY REFERENCE ARCHITECTURE

3.1 Smart City Scenario Description

The application developed in the Smart City domain (named "*Green Route*") aims at providing support to citizens' mobility in high polluted cities like Mexico City. With that in mind, the goal is to both improve the citizen's quality of life and foster environmental friendly behaviours. The end-user perspective is briefly described below.

Green Route's intention is to help the final user determining the best route to follow in order to reach a specific destination considering the user's: profile (such as health conditions), personal preferences (such as transportation type, etc.) Green Route is able to trace the ideal route for the user avoiding routes with: high levels of pollution, traffic jams or pollen saturated areas, etc. With the latter, the suggested route not only complies with the user's preferences but prevents possible health crisis or asthma attacks due to the elevated pollen concentration levels at certain areas and, hence suggesting custom made routes for people with respiratory diseases.

3.2 Smart City Scenario State of the Art

Population is concentrating more and more in cities as years go by. According to the World Bank, 53.4% of the world population resides in a city [3]. With such growth trend, cities are in a great need of becoming a sustainable environment that offers its inhabitants connected and controlled services. Education, security, sanitation, transportation are expected to be responsive to the growing needs of the population, by implementing the integration of sensors along with a robust ICT platform cities can become a living and evolving environment to fulfil such needs. Policy makers can support their decisions not only on a snapshot of what happens in a city with traditional measuring instruments (surveys, questionnaires, etc.) but can be better informed by the integration of all the available data collected by the aforementioned sensors and platforms. Real-time dashboards can display data from a plethora of sources; pollution, traffic, waste energy consumption, light sensors, mobility efficiency, etc. Smart Solutions take a vital role in the transformation of data into valuable information that will allow key stakeholders adjust their goals and course of action according to the current trends and situations happening in their ambiance. Mexico's growing population presents a challenge for all involved decision makers (policy makers, industry leaders, politicians, etc), the needs change abruptly and demand constant adjustments. Based on the United Nations Estimates, the current population is of **130,286,724** people. The challenges that such amount of people represent are reflected on everyday conflicts; insufficient transportation modes, high polluted zones, etc. that translates into discomfort and potential risks to the citizens. In order to be able to provide a solution capable of dealing with such dynamic conditions, the SmarSDK project focuses its efforts in providing a viable tool that will allow its users the possibility of better use the current resources by means of combining IoT enabled sensors and a robust cloud platform to concentrate all the collected data and transform it into a simple to use solution for Mexico City's citizens.

3.3 Smart City Scenario Architecture in SmartSDK

The current Smart City architecture consists of a set of modules that allows the final user to be able to find the best route to follow to reach a specific destination, based on distinctive aspects that describe the user such as its profile (health conditions), preferences, transport type, etc. Figure 4 presents the complete architecture for the Smart City Scenario. This architecture includes some of the components developed as part of the SmartSDK (cf. D3.2), such as: *Smart Spot*, *Cloudino*, and *SDK library* for *NGSI*.

To develop an architecture that allows the creation of an application for both pollution and traffic monitoring, we started by obtaining air quality data from IoT fixed and mobile devices. For instance, Smart Spot and Cloudino are able to record temperature, humidity, location data and are also able to

detect various gas types (NO_2, SO_2, CO, O_3). First, captured data is sent to the *Backend Device Management* - IDAS through LWM2M and MQTT protocols. Then, the data is sent to the *Orion Context Broker* in NGSI format. Furthermore, the Smart City scenario uses the concept of human-as-a-sensor in which a novel mobile alert application is being developed. This mobile application will allow users generate alerts concerning: accidents, traffic jam, pollution, weather conditions, pollen and trigger an asthma attack alert. Additionally, data captured from users is send to the Context Broker. Furthermore, data will be stored using the Cosmos GE in a mongo database. A routing engine was developed to work on data collected from public transportation open data sites, with such information the engine is capable calculating and finding the best route between two locations using public transportation. In order to accomplish that, route engine allows the user to specify several parameters related to its route; departure and arrival times, maximum walking distance and others.

To present the collected data to users in a friendly and novel way, Front and Back ends were constructed. Interfaces and services work together to present the user relevant information such as alerts generated by other users and IoT devices. In the Front-end, the user registers data concerning its health condition, interest groups subscription, its own alerts generated, etc. The Back-end is connected to the *Identity Manager* - Keyrock to authenticate user and services.

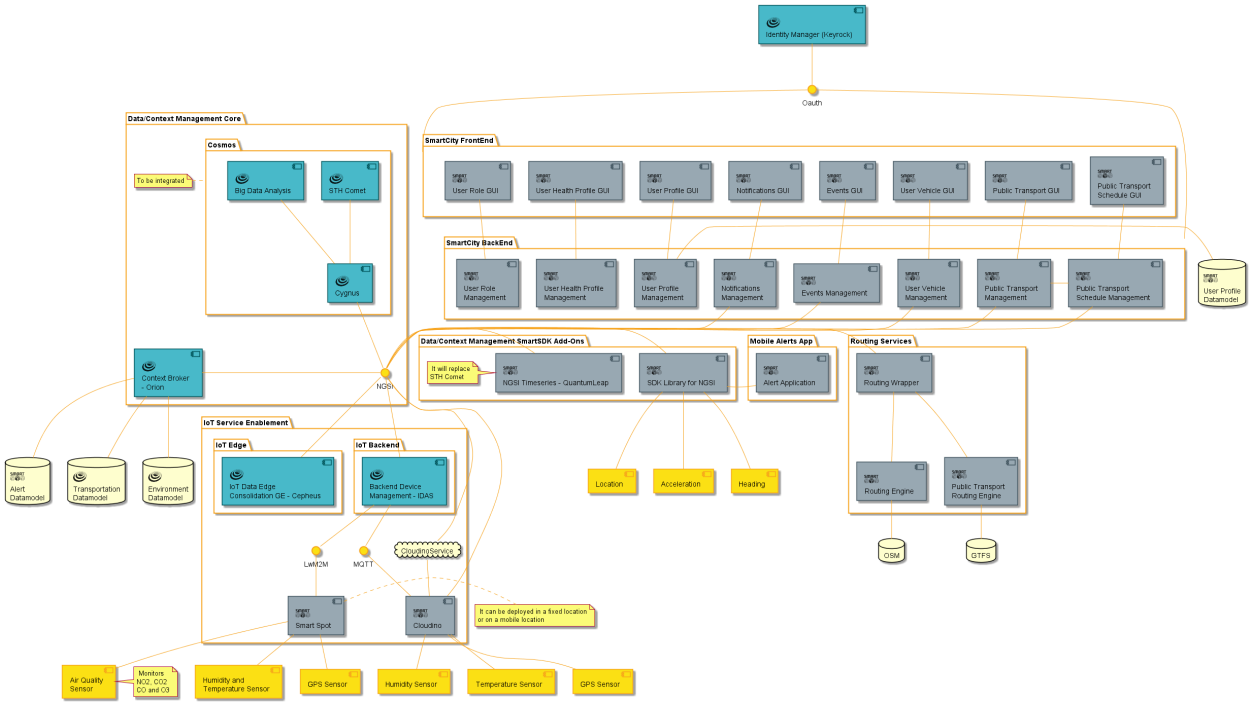

Figure 4. Complete architecture of Smart City Scenario.

3.4 Smart City Generalized Architecture Patterns

Based on the architecture presented in the previous section and considering the domain requirements along with the work presented in D3.1 "*SmartSDK Reference Models and Recipes*", the following reference architectures are formulated based on specific needs to exemplify additional viable scenarios.

3.4.1 Environment monitoring Architecture pattern

In order to be able to monitor the pollutant levels in a city, an environment monitoring pattern has been designed. The Environment Monitoring Architecture pattern distinguishes a collection of components that can be used to develop an application that presents users a selection of alerts based on collected data from both IoT devices and users. Such application is a mobile alerts app that notifies users about different events such as: traffic jams, pollution levels, weather conditions, pollen and

accidents.

The aforementioned IoT devices are a collection of sensors that measure: Temperature, Humidity, Gas Detection Sensor (NO_2, SO_2, CO, 0_3), and a GPS sensor. Collected data is sent using MQTT and LWM2M protocols to an IoT Back-end device management GE. This IoT GE sends data to the Orion Context Broker on NGSI format. In addition, when a user reports an event it generates an alert that's sent using a mobile alert application. Alongside with the report, data is sent to the Context Broker. There, an application that is subscribed to Orion Context Broker and when presented with a new alert it notifies a collection of users based on their preferences and a collection of predefined rules. Alerts are shown in the application's Front-end (such application is called Green Route in the Smart City scenario). Figure 5 depicts the aforementioned Architecture Pattern.

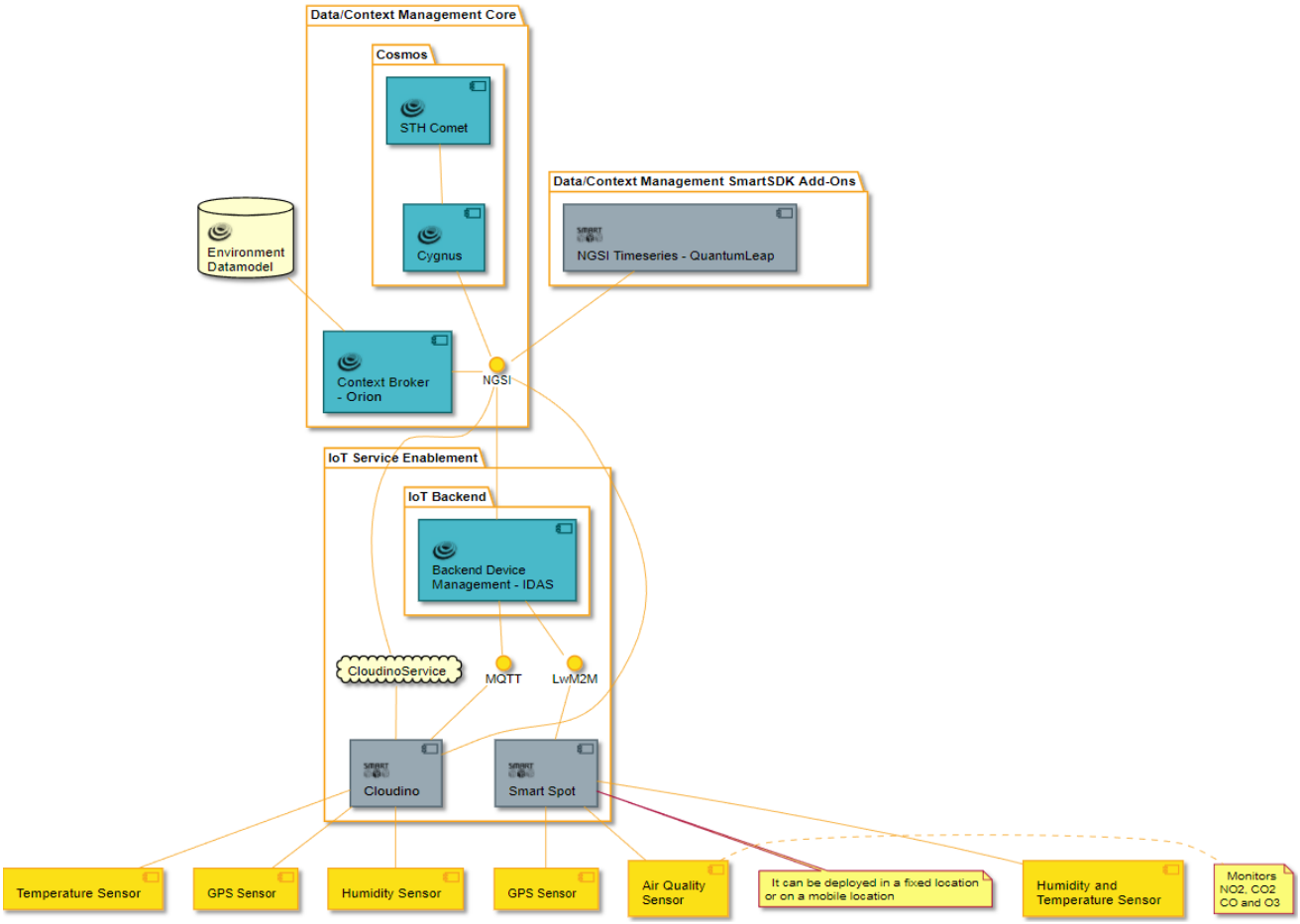

Figure 5. Environment monitoring pattern.

3.4.2 Transport Routing Architecture pattern

With the following Architecture Pattern, users are able to trace the best route between two locations using the public transportation system of a city. The Transport Routing Architecture pattern outlines the usage of IoT devices to retrieve the position of any given public transport in real time, this information will be presented to users in a mobile application.

The aforementioned IoT devices integrate a series of sensors such as a: GPS, accelerometer, and gyroscope. Such equipment can be located on an assortment of public transport modes (buses, subway, rail, etc.). The goal is to monitor key aspects such as: its schedule, stops and location of each transport. The aggregated information is sent to a Back-end Device GE and to the Cosmos GE to be stored. Then, a route service enabler calculates the best route, using GTFS data and a routing engine. The best route is sent to the Back-end Smart City Application with to be presented to the users. Figure 6 depicts the Transport Routing Architecture pattern for Smart City scenarios.

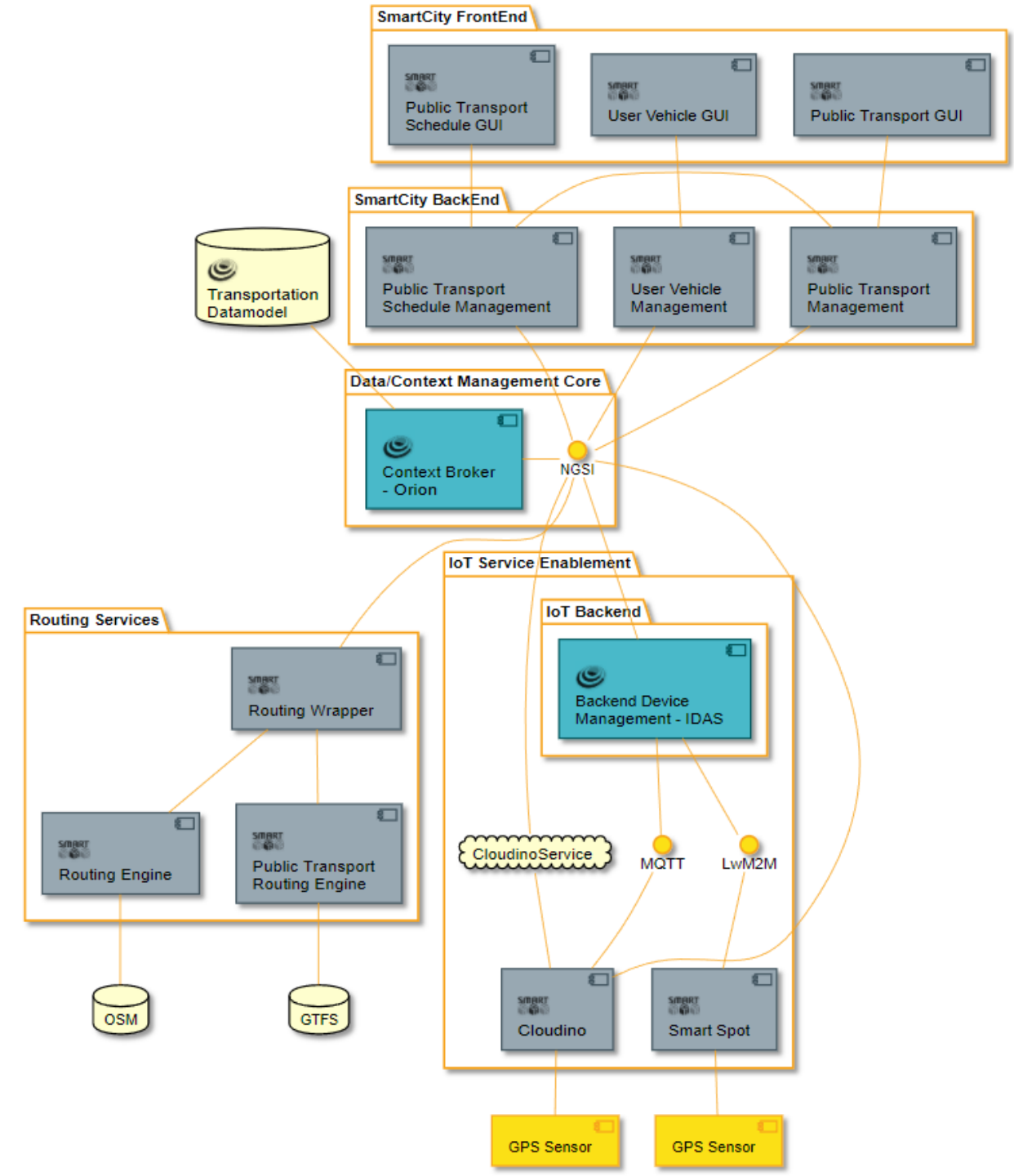

Figure 6. Transport Routing Architecture pattern.

3.4.3 Mobile Alerts Architecture pattern

The revolutionary concept of human-as-a-sensor is a key element of the SmartSDK project. The main idea is that users send an assortment of alerts such as accidents, traffic jams, pollution, etc. In order to be able to carry out such task the following discussion presents a Mobile Alerts Architecture pattern. The pattern comprises of a Library for NGSI that dedicates to collect location, acceleration and heading data. Such data is sent to Orion Context Broker using the NGSI format. From the gathered data a set of notifications are sent to user. Using the same application, a user is able to submit a new alert based on an observed event that needs to be reported. Figure 7 presents the main components

implemented on the aforementioned Architecture pattern.

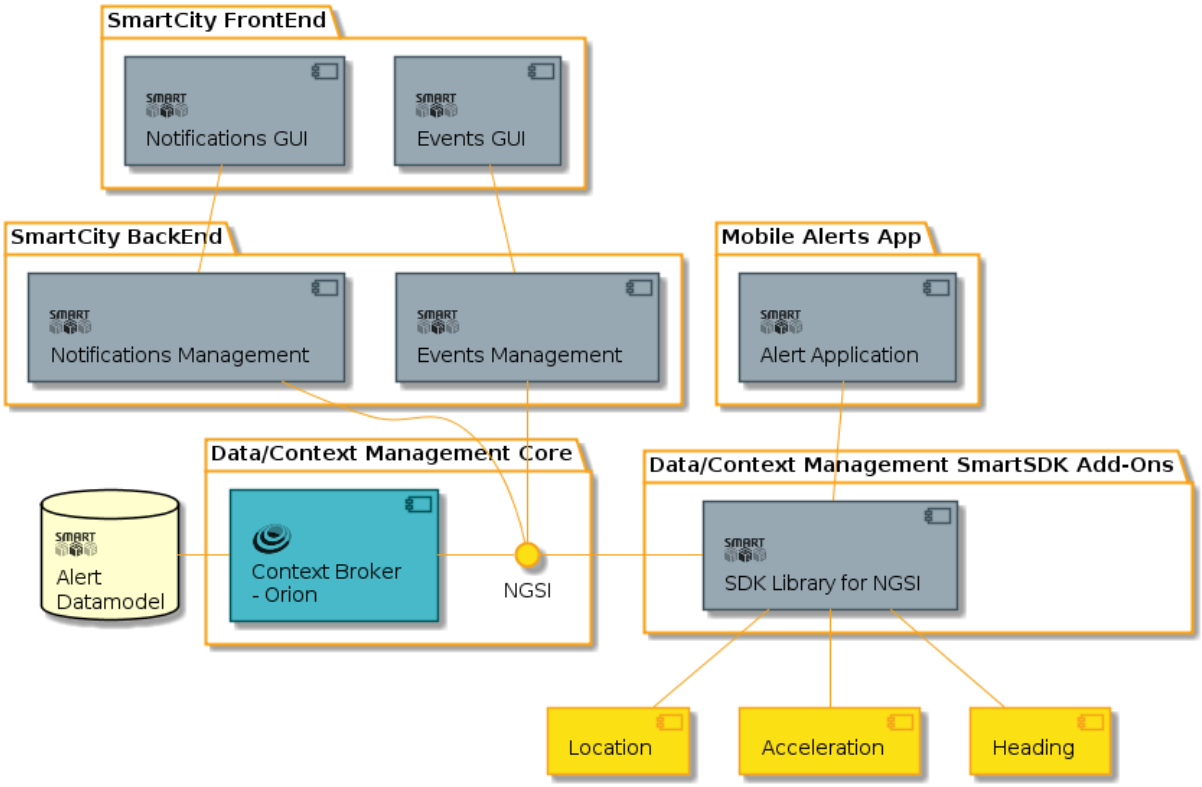

Figure 7. Mobile Alerts Architecture pattern.

3.4.4 User Management Architecture pattern

To guarantee the user the safety of an application the following User Management Architecture pattern is presented. The main goal of this Architecture pattern is securing the authentication process of users and applications. To be able to accomplish this, the *Identity Manager* - keyrock is invoked. In it, domains, roles and permissions are assigned for each type of user. Regarding this, a user only can access from the Front-end applications that are stored in the Back-end and are declared within the Identity Manager, hence, an application can only be used by users that have permissions. The objective is to ensure the security of the applications by avoiding the modification of its content by users without the proper permissions.

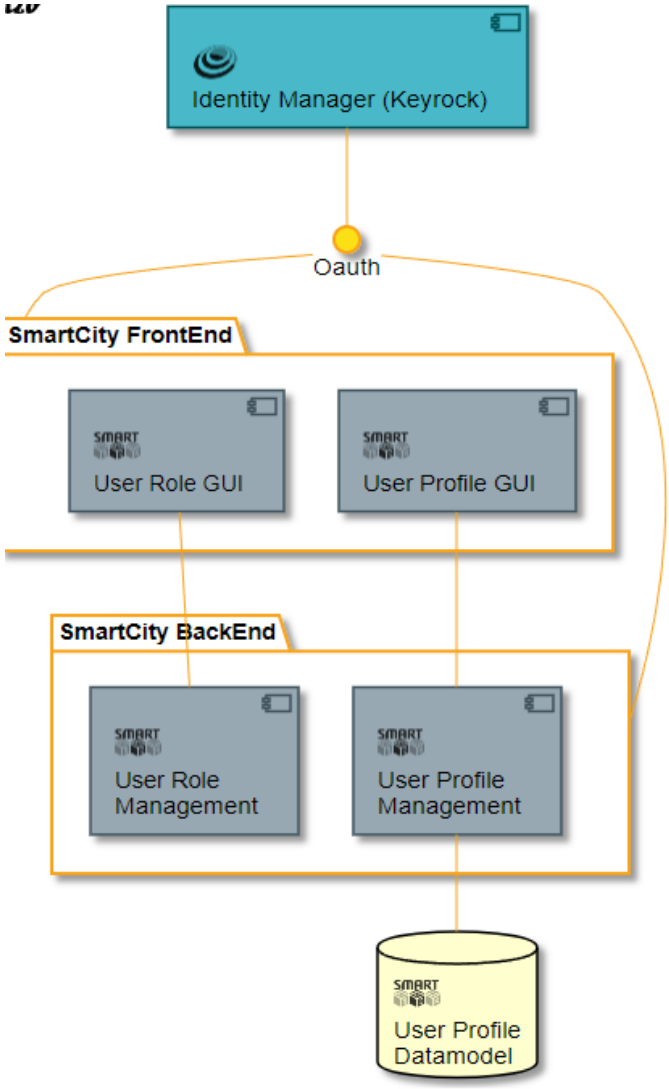

Figure 8. User Management Architecture pattern.

3.5 Roadmap

The previously described architecture is presented as the current solution developed by the project partners that tries to unravel the scenario's specific needs. It's worth noting that anyone reading the following document might implement a different solution but the key point is to serve as an innuendo to work on while solving issues similar to the ones described in this scenario. Likewise, the recipes that are being currently created might suffer slight changes due to the new deployment strategies being employed along with their implementations. Currently, after a year work effort, the following components were developed attending the principles of the Smart City scenario:

- New data models have been developed to contribute to the FIWARE Community. Data models such as: user, disease profile, alerts, public transport, smart spot and smart Poi.
  - The *alerts data model* facilitates the user to generate a given alert in a defined location.
  - The *public transport* models were developed based on the the *General Transit Feed Specification* (GTFS) [5] that defines a common format for public transportation schedules and associated geographic information.

- The *Smart Poi* model represents the area of interaction issued by a Smart Poi.
- Smart Spot describes the collection of devices that integrate a Smart Spoi. This devices can be managed through the IOTAgent.

➔ A number of modules have been designed and deployed to perform tasks such as:

- Record the user's profile.
- Record a respiratory disease.
- Record all the vehicles owned by a user.
- Creation and management of new alert groups such as; "*Bikers*", "*Runner*", "*Diseases*", "*Public Transport*" and "*Private vehicles*".
- Define rules for each type of user and their corresponding roles.
- Registration of public transport vehicles.

➔ A process to generate KML (Keyhole Markup Language) formatted files of the private transport geographical information.

➔ A module has been developed to represent the results from the route tracing engine on a map. The user is now able to graphically see the waypoints and follow the marked route to arrive at each point.

➔ Dedicated module to plot Mexico city's air quality data on a map.

➔ Development of a route tracing engine module that calculates the best route between two points using public transportation.

➔ Another routing engine was integrated to obtain the best route between two points.

➔ A crawler was developed to retrieve air quality data from the Mexico City's Air Quality web site [2]

➔ Developed a service that's able to study the user's interactions with the device.

The next steps of the Smart City scenario consist on the creation of the following elements and components:

➔ The current Architecture pattern will be refined.

➔ A collection of recipes based on the open platform Docker.

➔ A component that is able to record information of the public transportation of Mexico City.

➔ Integrate the real time location of the public transport vehicles in the maps.

➔ Display information related to pollen concentrations.

➔ Display real-time traffic information on a map.

➔ Integrate traffic and pollution data while calculating an optimal route.

➔ Perform a single LWM2M connection to multiple entities of the NGSI data model.

➔ Define and implement the alerts database schema.

➔ Define the "*observed event*" catalog.

Design and develop the alert application

4 SMART HEALTH REFERENCE ARCHITECTURE

4.1 Smart Health Scenario Description

The main purpose of the healthcare application is to expedite the harmonization and sharing of mobile sensing datasets for the healthcare domain. It focuses on mobile devices that collect data from sensors used on physical tests by following clinical protocols to assess the risk of falls.

The generated application has been designed for research purposes, thus, parameters of interest (associated to the risk of falling) are analyzed *a-posteriori* and raw sensor data is kept allowing different stakeholders, such as patients and physicians to better understand how patients' activities and behaviours influence a healthier lifestyle.

As the project's proof of concept, we describe two different scenarios:

1. Estimation of the risk of a fall.

   This application aims to collect sensor data from mobile devices (e.g., smartphones and smartwatches) while a participant performs a specific physical test under supervision and directions of a trained assistant. Controlled tests will focus on three physical performance measurements: Timed Up and Go (TUG), 4-Stage Balance Test, and 30-Second Chair Standard Test.

2. Rehabilitation at home.

   This service has two different front-ends; one for the patient which collects data and one for the clinician which summarizes the data from the patient. In particular, it measures mobility data to estimate differences in limb usage during the day for rehabilitation monitoring purposes. A pilot will be carried out to calibrate the system. First, with healthy phantoms and later with at least 1 patient with motor impairment undergoing rehabilitation therapy. Validation will be carried out using standard motor function assessment scales.

4.2 Smart Health Scenario State of the Art

A major concern in Mexico regarding healthcare is the changing demographics as the population ages. Working we the National Geriatrics Institute in Mexico, we devised a mobile sensing scenario to aid clinicians in assessing the risks of falls by detecting mobility parameters on older adults that conduct standard test that measure walking speed, balance and strength.

4.3 Smart Health Scenario Architecture in SmartSDK

Aforementioned scenarios are being developed under a single architecture, which encloses some of the most common services required in terms of a data sensing application.

Both applications are to be used as tools to collect mobile devices sensor-data (e.g., accelerometer, orientation) while patients perform a set of physical activities. The generated applications are meant to be installed in a smartphone/smartwatch that will be worn by the participant.

In this context, data has been collected on each device and will eventually sent to FIWARE Cloud by calling respective RESTful verbs. Data will connect to Orion Context Broker through the PEP component (Police Enforcement Point), which will invoke a filter mechanism to ensure that only expected data will be analyzed. Once data has been stored into Cosmos Big Data GE and processed by Theseus (a component that analyzes small amounts of data), data is then retrieved using Bifrost; which serves as an broker that transports query parameters from the Wirecloud GE. Finally, data retrieved from the Cosmos GE, it is sent back to Bifrost in order to be displayed on their corresponding widgets.

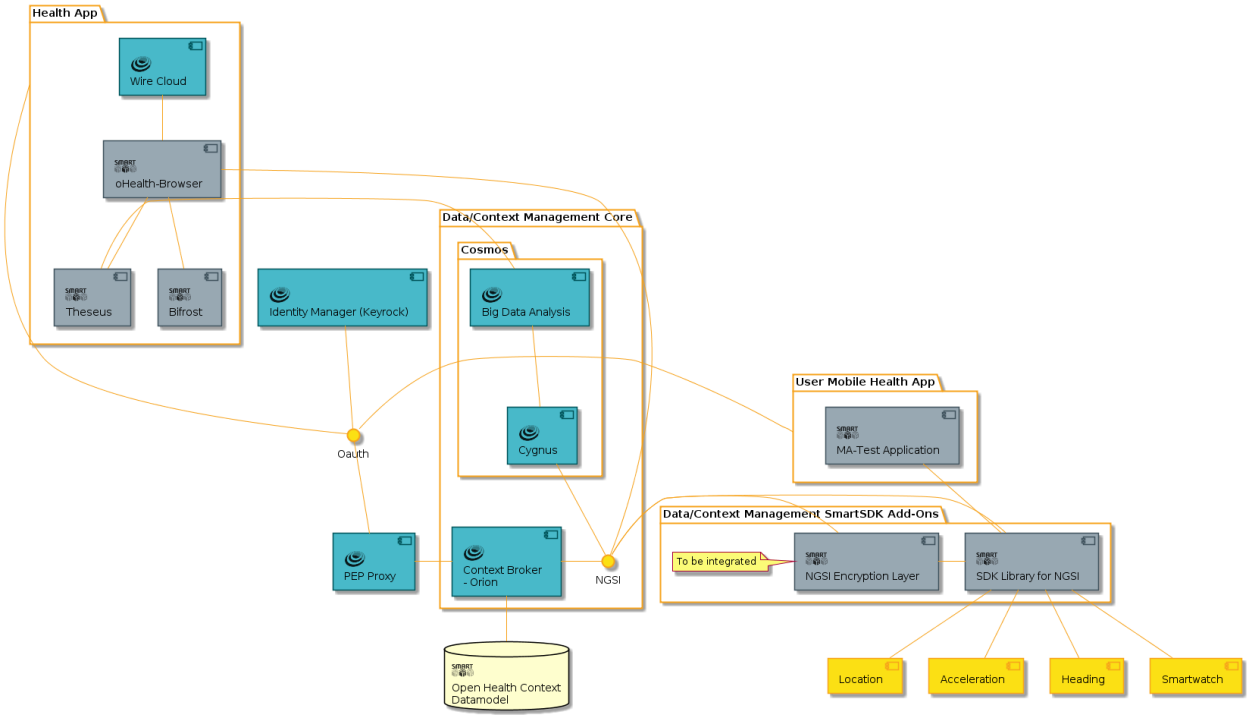

Figure 9. SmartHealth Scenario Architecture.

The Scenario's main contributions focus on three components:

- *Theseus*. A python-based implementation of open-source software to facilitate the analysis of data. It doesn't try to compete with the available Data Analytic GE, but to complement it for quick prototyping purposes by using public libraries, such as SciPy.
- *Bifrost*. A software implementation developed on python using the Flink framework to connect widgets with historical data components. Bifrost serves as a broker 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 corresponding 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.

4.4 Smart Health Generalized Architecture Patterns

Smart Health scenarios rely on a general architecture (Figure 9), that consists of three stages: (1) a software implementation for mobile devices (e.g., smartphones and smartwatches), (2) a set of schemas to wrap health data accordingly into NGSI specifications, and (3) a visualization mechanism to consult physical test results.

As illustrated in Figure 10, persistent sensor data is uploaded to the FIWARE cloud under NGSI standards through the Orion Context Broker GE and observing a set of open health-context schemas (oHealth Contexts). Then (Figure 11), by using the Cygnus connector and the Keyrock authentication mechanism sensor data is stored for further analysis under the Cosmos GE shelter. In this context, data is analyzed by using both Cosmos GE and Theseus algorithms to finally display processed reports in a graphical interface using a set of widgets (Mashup) over Wirecloud GE.

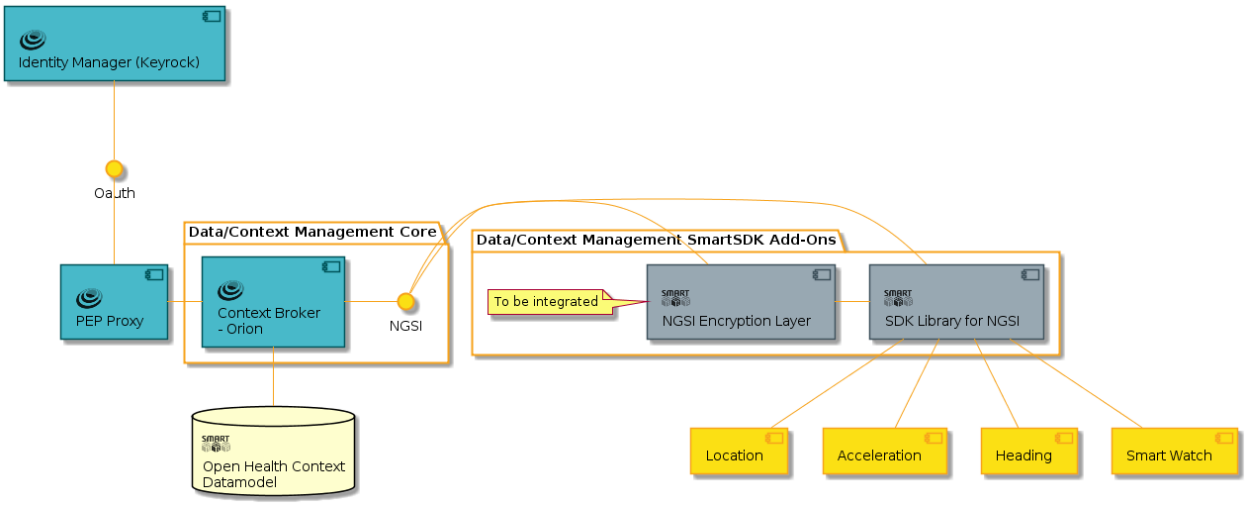

Figure 10. Sensible Data Collection Architecture pattern.

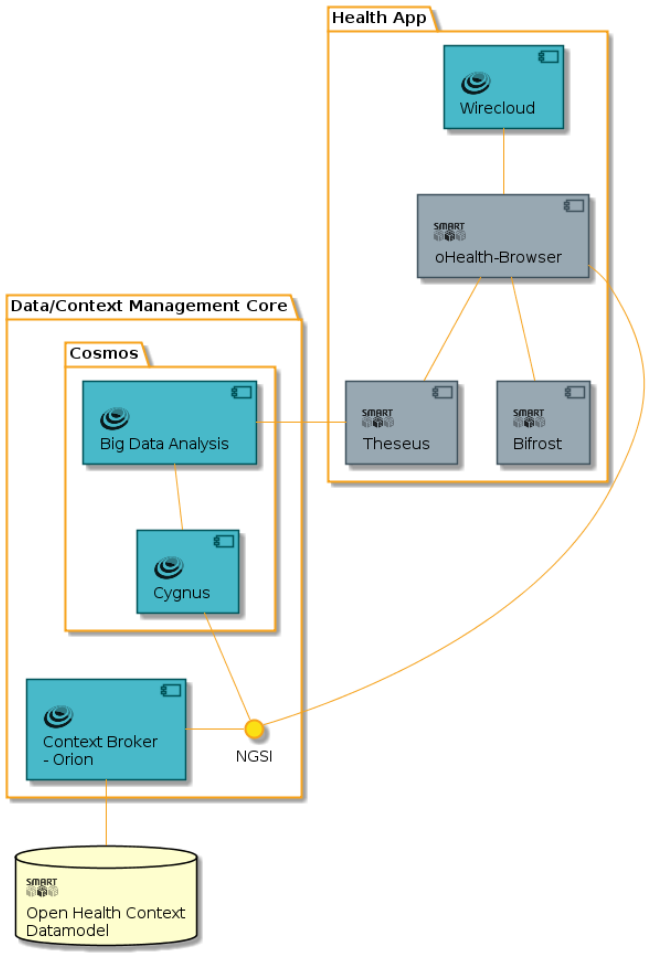

Figure 11. Data Analysis Architecture pattern.

4.5 Roadmap

The components developed within the first year of work are outlined below. Whilst, the architecture

pattern will be refined in the next deliverable, and recipes based on docker compose will be produced.

- Usage protocol
  - For the rehabilitation scenario, define a protocol of device usage, For instance, specifying the smartwatch placement and orientation, identification of arm associated to the smartwatch, non-wear maximum times, etc.
- 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 retrieve sensor data. The collected data gathered during the physical tests is stored in the mobile device and then transferred to the cloud. It becomes persistent by storing it using the Orion Context Broker and Cosmos GE.
  - Dashboard monitor.
  - A widget development, called "*oHealthCxt-Browser*" whose goal is to assist in the setup process of a file-manager to seek; physical-test performance results, parameters of interest, among other relevant information in the context of health applications.
  - A software component to retrieve historical data from the Cosmos GE (by reading and formatting HDFS documents into NGSI schemas), which can be used over Wirecloud widgets.
  - A widget component that can render a statistical summary of mobility data. Data is retrieved from the Cosmos GE and rendered using Wirecloud.
- Data analysis
  - *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.
  - *Subject study*. A first subject study is conducted to test both: FIWARE's integrated component and application development.
  - *Estimate of patient total activity*. Area under the accelerometer curve across all sensed devices (smartwatch x2, and smartphone).
  - *Estimate of patient activity per arm*. Area under the accelerometer curve per device
  - *Differences in bilateral arm usage for patient*. Differences between estimate of activity per arm.
- Access Control
  - Enrich the dashboard monitor of the rehabilitation patient with sign in and sign up capabilities.
  - Enrich the dashboard monitor of the rehabilitation clinician with sign in and sign up capabilities.
  - 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

- Data analysis mechanisms are integrated on the the application architecture, thus, results can be retrieved from the Wirecloud GE.
- Bring together data from different devices (2x smartwatches -one on each arm- and 1 smartphone) into a single packet of data to be uploaded to FIWARE's cloud.

➔ Wearable devices integration

- Creation of the project's proof of concept in order to practically test the flexibility of the data-model, over wearable sensing devices such as smartwatch.
- Test the proposed data-model to be used on smart watches in the rehabilitation scenario.

Overall, future activities will cover:

➔ Refinement of the architecture pattern.

➔ Production of recipes based on Docker compose.

➔ Design and development of an application for rehabilitation purposes to monitor motor function on a coarse scale during a regular week amid daytime (outside the rehabilitation ward).

➔ Design and development of an application that uses wearable devices to detect anxiety in people who interact with a social robot.

5 SMART SECURITY REFERENCE ARCHITECTURE

5.1 Smart Security Scenario Description

The Smart Security scenario focuses on the development of an application capable of providing support to a security guard to prevent risky situations. In order to prevent such situations the risk analysis will be conducted in areas with video surveillance cameras, thus improving the quality of life of the people.

Within the scope of this scenario, the application's core is the video stream analysis from a collection of cameras. The visual information contained in the video stream will be used to detect different events such as: access control alerts, people detection, crowd analysis, etc. Security risks like theft, fights, etc will be identified by the combined use of video cameras and mobile sensors in both indoor and outdoor scenarios (e.g., parking lots and buildings).

The security application has been designed for research purposes, thus, resulting data from the developed algorithms will be kept for further analysis protecting at all time the privacy of recorded people and events.

5.2 Smart Security Scenario State of the Art

Nowadays, with the emergence of Smart Cities and new generation sensors that are capable to connect to a network it is possible to monitor the entire infrastructure of a city including roads, bridges, rail/subways, airports, communications, water, power. Also, it's achievable to optimize its resources, plan its preventive maintenance and monitor security aspects while maximizing services for its citizens.

In particular, security aspects and the safety of people must be a priority of any government. People want to live in secure cities, travel on riskless transports and work in safe places. Likewise, companies want their goods to be securely transported and their factories to be safe. To address this issue, governments have been installing closed-circuit camera systems to prevent risk situations by using this kind of technology.

In Mexico City, there are about 2,100 cameras in operation as part of the "*Bicentennial Safe City*" project. Mexico's capital and gradually many other cities within the country have adopted video surveillance systems as many big cities in the world. However, despite these actions, according to National Institute of Geography and Statistics (INEGI)[4], the percentage of the population of 18 years and older that considers their city unsafe has dramatically jumped from 67% in 2014 to 71.9% in third trimester of 2017.

Although traditional video surveillance systems have proved to be a very useful tool, having smart video surveillance can become highly useful for security guards to detect/prevent risk situations in real time. The major issue on assigning a security guard the task of watching over a set of video streams, is that people's attention, in general, is limited. Due to that fact, there is a high probability that they'll miss some relevant information, that it in the context of security it's considered inadmissible because of the consequences this might have.

[4] As per its spanish acronym (Insituto Nacional de Estadística y Geografía)

With the growth of the Internet's capacity and speed, as well as the advent of ICT technologies, companies, cities and citizens can improve the security of their environments. Nowadays sensor data (e.g., video) can travel live over the internet and inform in real time about any possible hazard. These sources represent potential knowledge that can be exploited for developing preventive mechanisms that can improve security and safety in cities and institutions.

Based on such important need, the SmartSDK project aims at developing a security application. The goal of the Smart Security application is to integrate automatic video analysis techniques to surveillance systems. All, based on computer vision algorithms that are capable of performing tasks as simple as detecting movement in a scene, to more complex ones such as vehicles and even people activities. Currently, FIWAREs cloud architecture components are integrated. Components such as the Orion Context Broker is used to share information about the context where the system is installed along with Kurento, which is used for streaming management.

5.3 Smart Security Scenario Architecture in SmartSDK

The Smart Security architecture is being developed to detect security risk events based on the video stream acquisition from a set of sensors (e.g. video cameras) connected to a network. It is then analysed to detect, label, store and highlight security-relevant events automatically.

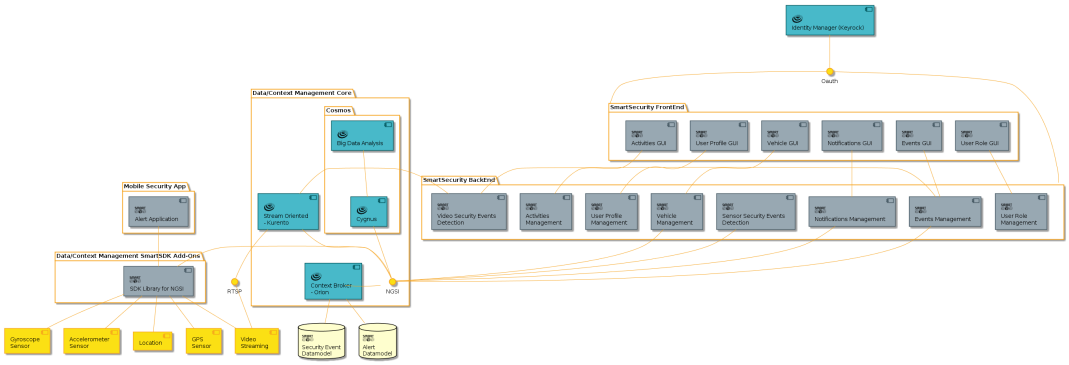

Figure 12. Smart Security Scenario Architecture

To capture the incoming video streaming from IP cameras we are using Kurento GE through the Kurento Media Server (KMS). The KMS is based on Media Elements (ME) and Media Pipelines. The former consists of modules that perform a specific action on a media stream receiving or sending media from other elements, the latter is a chain of media elements. For the security application we have developed a set of filters (ME) capable of subtracting objects (people and vehicles) in motion within a scene, classify such objects and track them. From this information, we are developing algorithms that can detect security events. All the data generated from the stream processing is sent to the Orion Context Broker through its connection with Kurento. Furthermore, the data from the detected security events is stored into Cosmos Big Data GE.

The contributions of the aforementioned architecture consist of the design and implementation of; new Kurento filters to extract relevant information within the scene, a custom made graphical user interface (GUI) with various functionalities. The GUI provides the user with various functionalities; alerts of risky events, retrieval of historical detected events, register new users, customize the displayed visual information, etc.

5.4 Smart Security Generalized Architecture Patterns

Based on the architecture presented in the previous section, we have formulated the following architectures according to the specific needs of the security scenario.

5.4.1 Video Event Detection Architecture pattern

The Smart Security application aims to detect security risks through video stream analysis, thus, a Video Event Detection Architecture pattern was detailed. This Architecture pattern describes the main FIWARE components needed to build the application. In general, this pattern allows us to capture a video stream from a set of cameras, analyze the information through a set of novel Kurento filters that are being developed and to generate notifications/alerts to the user about the monitored area.

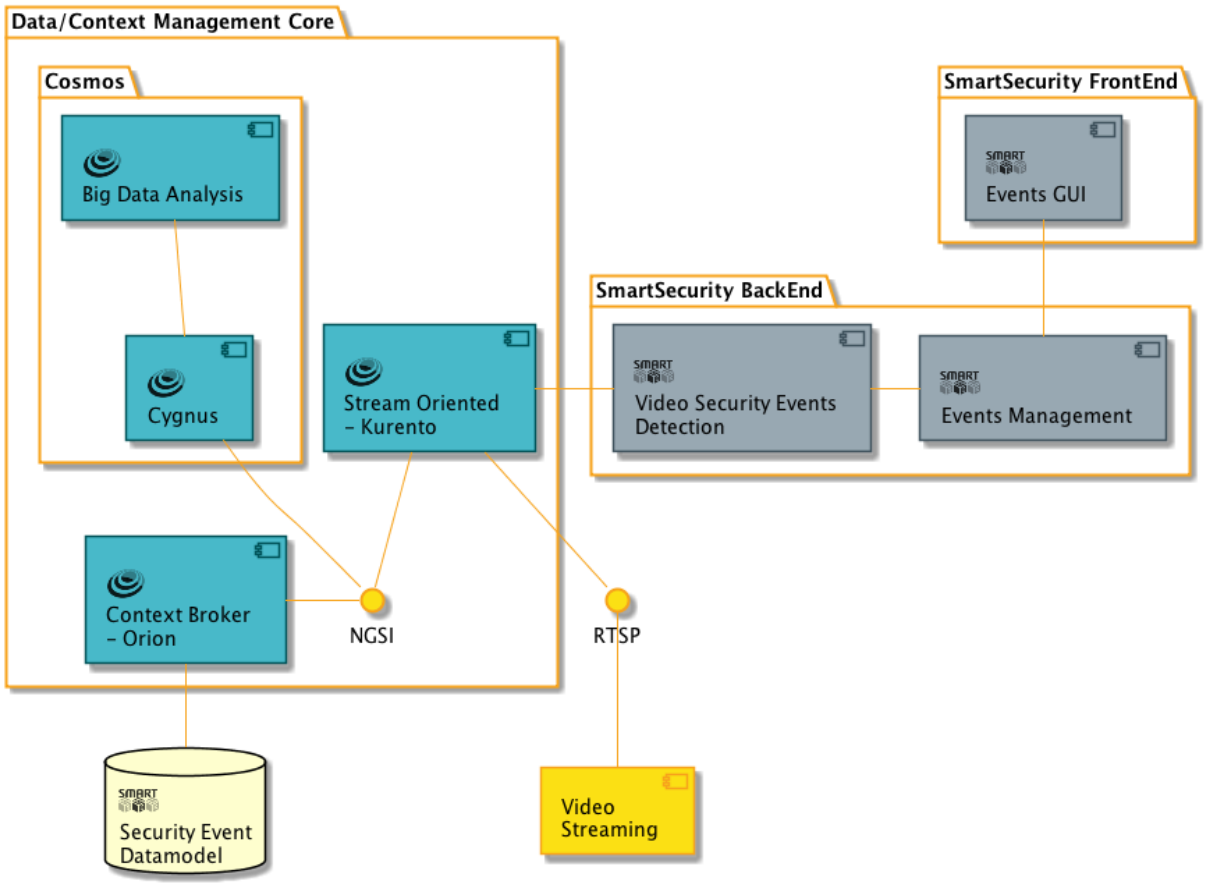

Figure 13. Video Event Detection Architecture pattern.

5.4.2 Event Detection Through Smartphone Architecture pattern

In order to improve the detection of risk situations and also be able to detect them in areas that cannot be monitored by a camera, an Event Detection Through Smartphone Architecture pattern was generated. With this Architecture pattern we are able to collect data from the smartphone. The collected data contains information regarding: velocity, acceleration and position of the user, then the data is sent to the Orion Context Broker using the NGSI format to be analyzed.

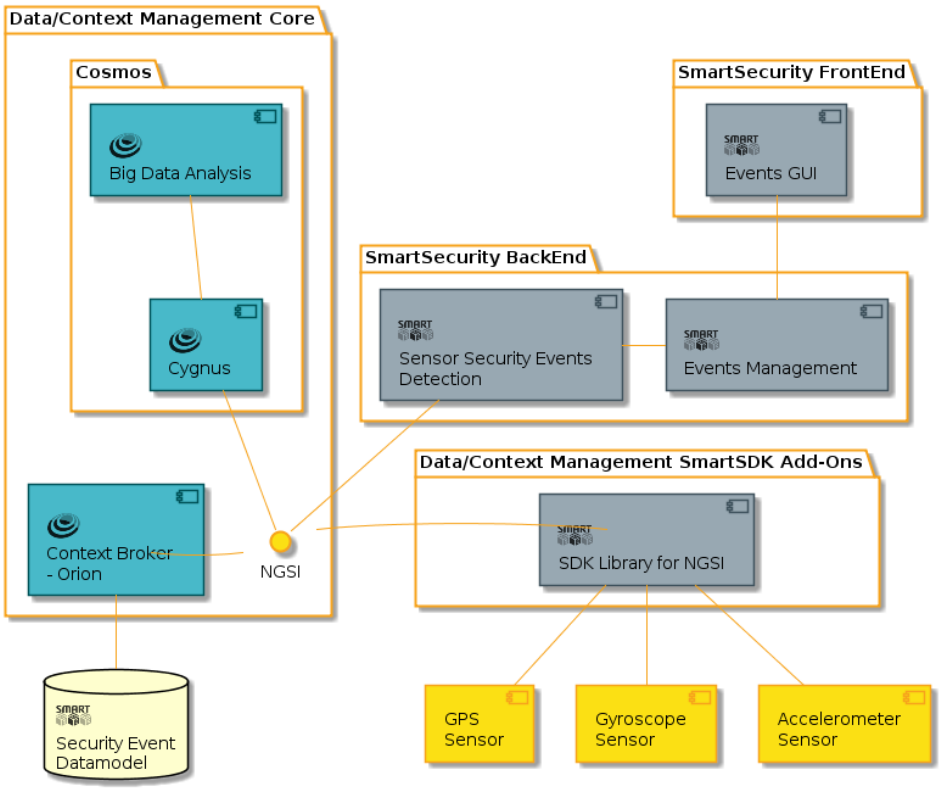

Figure 14. Event Detection Through Smartphone Architecture pattern.

5.4.3 Alerts Architecture pattern

To help the user to quickly respond at any given risky situation the Alerts Architecture patterns has designed. This pattern is being developed using the NGSI format to gather information from the video/smartphone event detection. The data is then sent to the Orion Context Broker and then in the front-end application the user is notified.

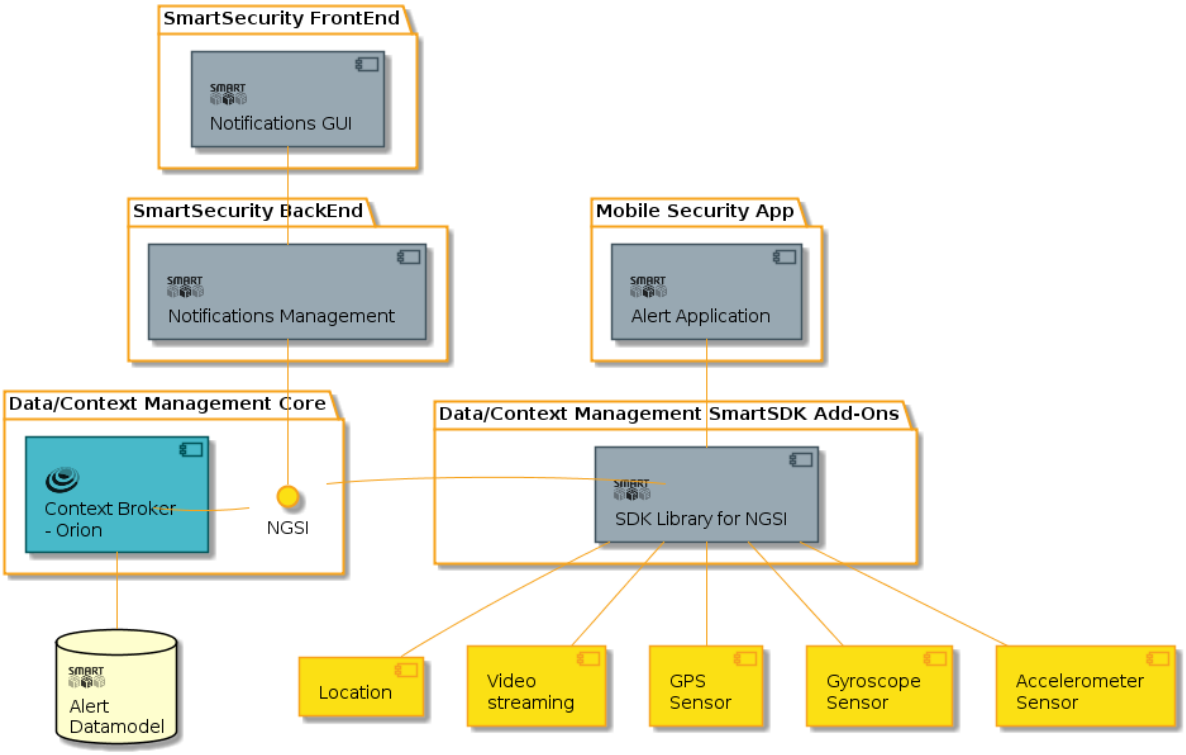

Figure 15. Alerts Architecture Pattern.

5.5 Roadmap

The architecture pattern presented in this document can be seen as an initial version that will be refined along with the next deliverables.

The set of components to support the smart security scenario that will be developed are:

- 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 (test location).
  - 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 time 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 a smartphone that will be capable of identifying people 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 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.

➔ Pattern recognition

- Test of module integration to evaluate the communication between them, data transmission to the cloud as well as the performance of the implemented algorithms.
- Development of a software module (Kurento filter) for vehicle detection 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 test the efficiency of people detection/recognition algorithms.
- Creation of a database of vehicles to test the efficiency of the vehicle detection/classification algorithms.
- Development of a software module (Kurento filter) capable of detecting people in a video stream in outdoor environments.
- Development of a software module (Kurento filter) capable of detecting if a vehicle is parked incorrectly.
- Development of a software module based on the data from a smartphone capable of detecting if vehicles are going on the wrong direction within the parking lot.
- Development of a software module (Kurento filter) capable of recognizing people in a video stream in outdoor environments.
- Creation of a database of people behaviours to later test the detection/recognition algorithms of activities.

6 CONCLUSIONS

This document presents the specific Architecture used to develop smart applications in the Smart City, Smart Healthcare, and Smart Security domains. Such architectures consist of components both Generic and Specific Enablers, data models along with generalized architecture patterns that provide an insight of additional applications derived from the main architectures of each domain. Providing the tools needed to expand and create new applications within each of the three domains.

Domain Specific and generalized architecture patterns are the main contributions of the document the represent FIWARE's main benefits: flexibility and reusability of the components. By providing the building blocks to develop and enhance current solutions within a domain or even integrating components that might belong to a different domain but that might be the key element to solve a specific and fine grain problem pertaining another domain.

REFERENCES

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

[2] SmartSDK Consortium. D3.1 SmartSDK Reference Models and Recipes.

[3] Urban population (% of total). (n.d.). Retrieved July 01, 2017, from http://data.worldbank.org/indicator/SP.URB.TOTL.IN.ZS/countries?display=graph

[4] SmartSDK Consortium. D3.2 SmartSDK IoT and Data Management Enablers.

[5] General Transit Feed Specification. Retrieved July 01, 2017, from http://gtfs.org/

[6] Office of Atmospheric Monitoring. Retrieved July 01, 2017, from http://www.aire.cdmx.gob.mx/default.php