SMARTSANTANDER

SMARTSANTANDER PROJECT

INFSO-ICT-257992 SmartSantander

D1.2

Second Cycle Architecture Specification

Contractual Date of Delivery: 30th May 2012

Actual Date of Delivery: 11th June 2012

Editor(s): CEA

Author(s): See Authors list

Participant(s): ALU-IT, EYU, TTI, UC, UNIS, UZL, CEA, CTI, AI, TID, SACCOM (WUT, CERETETH)

Work package: WP1

Estimated person months (per partner): ALU-IT: 0.2; EYU: 0.5; TTI: 1.99; UC: 2.3; UNIS: 1.7; UZL: 1.5; CEA: 1; CTI: 1.5; AI: 0.05; TID: 0.1

Security: Public

Version: 1.0

Abstract: This deliverable presents the SmartSantander architecture evolution for the second cycle of the project. New use cases have been chosen with a new focus on citizens of Smart Cities. These new use cases along with the winning proposals of the first Open Call have driven the evolution of the architecture definition and its realisation.

Keyword list: *Architecture, components, development phases, interfaces, SENSEI, Telco 2.0, WISEBED.*

Disclaimer: This document reflects the contribution of the participants of the research project SmartSantander. The European Union and its agencies are not liable or otherwise responsible for the contents of this document; its content reflects the view of its authors only. This document is provided without any warranty and does not constitute any commitment by any participant as to its content, and specifically excludes any warranty of correctness or fitness for a particular purpose. The user will use this document at the user's sole risk.

Authors

TID	José Antonio Jimenez	jajh@tid.es
ALU-IT	Monica Russo	monica.russo@alcatel-lucent.com
EYU	Srdjan Krco	srdjan.krco@ercisson.com
TTI	Ramón Bezanilla	rbezanilla@ttinorte.es
UC	Luis Muños	luis@tlmat.unican.es
	José A. Galache	jgalache@tlmat.unican.es
UNIS	Alex Gluhak	a.gluhak@surrey.ac.uk
UZL	Florian Massel	massel@itm.uni-luebeck.de
	Sebastien Ebers	ebers@itm.uni-luebeck.de
CEA	Pierre Roux	pierre.roux@cea.fr
	Vincent Berg	vincent.berg@cea.fr
CTI	Evangelos Theodoridis	theodori@cti.gr
AI	João Fernandes	joao.fernandes@alexandra.dk
SACCOM	Tomasz Tajmajer WUT	t.tajmajer@gmail.com
	Manos Koutsoubelias CERETETH	emkouts@gmail.com

Table of Contents

List of Figures

List of Tables

Acronyms and Abbreviations

AAA	Authentication, Authorisation and Accounting
ACI	Access Control Interface
ASI	Application support interface
AR	Augmented Reality
CAF	Context Awareness Framework
CEP	Complex Event Processing
CM	Configuration Management
DoS	Denial of Service
ESI	Experimental Support Interface
FCAPS	Fault, Configuration, Accounting, Performance, Security
FM	Fault Management
GPS	Global Positioning System
GW	Gateway
HTTP	Hypertext Transfer Protocol
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
KPI	Key Performance Indicator
MSI	Management support interface
NMS	Network Management System
MOTAP	Multihop Over The Air Programming
OLAP	On-Line Analytical Processing

OSA/V	OS Abstraction and Virtualisation
OTA	Over-The-Air
OTAP	Over-The-Air Provisioning
PC	Personal Computer
POBICOS	Platform for Opportunistic Behaviour in Incompletely Specified, Heterogeneous Object Communities
POI	Point of Interest
PM	Performance Management
QR Code	Quick Response Code
REST	Representational State Transfer
RFID	Radio Frequency Identification
SACCOM	Soft Actuation Over Cooperating Objects Middleware
SAN	Sensor Area Network
SM	Session Management
SMS	Short Message Service
SNAA	Sensor Network Authentication and Authorization
TLS	Transport Layer Security
TR	Testbed Run Time
UAS	Universal Alert System
URL	Uniform Resource Locator
USN	Ubiquitous Sensor Network
VTD	Virtual Testbed Device
WAN	Wireless Area Network

WISEBED	Wireless Sensor network testBeds
WoT	Web of Things
WP	Work Package
WSN	Wireless Sensor Network

1. EXECUTIVE SUMMARY

This deliverable introduces the second cycle architecture of the SmartSantander project, which is built incrementally on the basis of the first cycle architecture presented in deliverable [D1.1]. The process justifying this architecture update is thoroughly presented in the document.

First, new use cases that complement the use cases considered during the first cycle architecture design are introduced. While the first cycle design was focused on IoT experimentation for researchers, the current cycle is increasing the focus of the project on the citizens of smart cities. Six new use cases with this objective are considered.

The '*Participatory sensing*' use case expands sensing information to citizens through smart phone sensors, and proposes new applications for their exploitation. The '*SmartSantander augmented reality*' use case will use RFID tags, QR codes and video processing to offer augmented reality applications and enrich visiting experience for tourists. The '*Mobility sensing*' will improve classical sensing coverage by including mobile sensors attached to public transport vehicles. The '*Irrigation*' use case will facilitate the task of the city gardening teams by leveraging sensor networks that will be deployed for monitoring irrigation conditions.

Finally, two middleware based use cases are also introduced. These two use cases are based on the experience of winners of the first open call of the project. The 'CityScript' use case will provide personalized sensor based services to the citizens. The 'SACCOM' use case will leverage sensor and actuator networks in view of energy savings, by promoting energy saving behaviours.

New requirements are identified from the analysis of the new use cases on one hand, and from a review of the initial requirements on the other hand. Then, an updated architecture is derived based on the identified requirements. It maintains the system interfaces from Cycle 1 which were already well adapted to the new use cases; however new functional entities had to be introduced and specified. The new architecture also introduces the concept of 'Virtual Testbed Device' that will expand experimentation flexibility and capabilities by introducing virtual IoT nodes which will be managed similarly to the real ones.

Beyond the generic architecture specification, architecture realization guidelines are provided as an input towards the work package in charge of specification and design (WP2). A strategy for horizontality support in the second cycle architecture instantiation is presented. New use cases such as 'mobility sensing', 'participatory sensing', and middleware experimentation are then addressed in terms of architecture realization.

2. INTRODUCTION

This deliverable captures use cases and requirements for the second cycle architecture specification of the SmartSantander experimentation facility. It then derives an updated architecture that takes the new requirements into account.

Internal Report [IR1.3], which had a focus on identifying new use cases and inferring new requirements, served as a starting point for the second cycle architecture definition.

Compared to the previous use cases, which primarily focused on the experimental researcher as an end user, the second cycle specification targets use cases that have a greater and direct impact on citizens. Such use cases involve citizens or visitors of the cities in which smart Internet of Things (IoT) facilities are being deployed.

For this purpose, six complementary use cases have been added to the use case portfolio and are described in more details in section 3. Four of them are motivated from the WP4 work on services to be developed in the context of the city of Santander, namely Augmented Reality, Participatory Sensing, Precision Irrigation, and Mobility Sensing. The other two use cases, SACCOM and CityScripts, represent the experiments successfully selected from the First Smart Santander Open Call. Together these use cases provide new challenges and requirements, which will drive the evolution of the experimentation facility in terms of the capabilities and features. Finally, requirements are inferred from the new use cases in section 3.3.

An updated architecture is then proposed in section 4. The initial architecture built during the first cycle ([D1.1]) served as a basis. New functional entities have been introduced to support the new use cases. However, the main architecture interfaces have been kept unchanged. This section also introduces the concept of a Virtual Testbed Device (VTD) which is brought into the new architecture for the purpose of improving experiment flexibility. An overview of the second SmartSantander architecture realization is also proposed in section 5, in order to provide guidelines to be used in WP2 . However detailed specification is out of the scope of the present document and will be produced by WP2 in the deliverable D2.2. Horizontality, which is expected to be introduced in cycle 2 as a way to provide concurrent use of testbed resources, is studied in this section. Guidelines are also proposed for the implementation of new use cases, namely participatory sensing and mobility sensing. Middleware based use case are detailed in this section as well.

Section 6 concludes the document.

3. SECOND CYCLE USE CASES AND REQUIREMENTS

3.1. Introduction

This section presents the six new use cases introduced for the second cycle of the project. Then detailed requirements for phase 2 are derived.

3.2. Service Experiment Use Cases

During the second year several use cases have been identified. Some of them were proposed by the expert reviewers during the first General Review [Technical Review Report]. This is the case of mobility sensing and participatory sensing. Some others come from the work carried out in WP4 as it is the case of augmented reality and irrigation. Last but not least, the experiments proposed by the winners of the first open call are also contributing to the use case assets.

3.2.1. Participatory Sensing

In this scenario users make use of their mobile phones to capture information about the physical environment, e.g. GPS coordinates, compass, environmental information and feed this information to the SmartSantander platform. This scenario aims at exploiting sensors embedded in the smart phones and related information sources to enrich the information made available to the municipality and other entities with city management responsabilities.

The users may also report various events that are made available to other users subscribed to the types of events the newly reported event belongs to. The notification of the occurred event is received via a Smart phone application, phone call, SMS or e-mail in the preferred language.

All users interested in receiving the notifications have to subscribe to the service, define their personal profile (including e.g., the preferred language) and select the information they are interested in.

For example, Maria has finished work for the day and is returning home by bus. The trip usually takes around 20 minutes. Unfortunately, for unknown reasons, the traffic is busier today. After 10 minutes, Maria realizes that the traffic jam is due to an accident. Immediately, Maria uses her smart phone to report the accident. Her colleague Carla, who uses this application, receives an alert on her smart phone about the accident. At the same time, Carmen (who does not have a smart phone) receives an SMS as she is registered to the corresponding notification service. Carla decides to take an alternative route home in her car. After receiving an SMS, Carmen decides to reschedule her planned trip to the gym.

Due to a financial disagreement, George needs to travel to the bank's headquarters in the town centre. Before leaving home, he logs into a route planning application on his mobile phone, defining his destination and providing anonymous information regarding his car's velocity to be used to judge traffic conditions. The application finds out that he is able to join a participatory network of citizens with mobile devices. This application informs him that a part of the previous route is heavily polluted and blocked, therefore an alternate route is proposed.

> *Upon reaching his destination, the sensors deployed around the bank office detect his presence and identify him as a premium bank customer, thus routing him towards a free parking space.*

3.2.2. Augmented Reality

Augmented Reality (AR) technology may support the tourists and enhance the experience of visiting a new city. By means of the AR application running on the tourist smart phone, (s)he will gain direct access to information about a specific monument in his preferred language. The position of the device is used to create an augmented view of the POI and overlay the video produced by the smart phone camera with virtual objects (mainly digital content, video, texts, photos, etc.) about the points of interest (POI).

For each POI, a description of the place or reference image will be provided in the AR view and the type of content (3D model, image, videos, audio, etc.) will be superimposed. The content itself is not part of the POI, but there will be a link to the content instead. Users may also define their own preferences (language, touristic places to visit, monuments, etc) to have a better experience visiting the city.

The use of tags (either RFID tags or QR codes) will provide information that is not included in the POI itself. Furthermore, a trigger can be defined for each POI to execute specific functions (play an audio, advertise events, etc.), when a visitor is in the vicinity. On a small scale, the use of tags provides an opportunity to distribute location-based information in the urban environment. On a larger scale, the tags may be coupled with more advanced services such as “feedback” from the citizens to the city council.

The users’ behaviour may be recorded in an anonymous way. It will provide third party developers the possibility to enrich their application and services, based on the user experience within the city. For the municipality, information sent by AR applications will provide data, for example: the number of people visiting the different POIs, or at least in the vicinity, the visitors’ nationalities etc. Based on this information, the city may improve its touristic offer and adapt its cultural events and activities. Furthermore, users will have the possibility of giving feedback about the POI, either by writing a small text or simply by clicking a “like button” as is done on Facebook.

Juan is visiting Santander for the first time. He decides to visit the city where he wants to see some museums. However he is not aware where the museums are located neither what kind of museums the city has. When going in the city Juan sees a building that looks like a cathedral. He takes his phone and uses an augmented reality application, in the image displayed by his camera he can see that he is in front of the “Catedral de Nuestra Señora de la Asunción de Santander”, some historical information about the catedral is also displayed. He can also see some pictures from the inside of the cathedral taken by other users of the application. Juan decides to take also a picture to the cathedral and upload it to the application, he puts a label to the picture “Cathedral by September 2012”.

3.2.3. Mobility sensing

Current solutions for environment monitoring in urban settings are based on a handful of measurement stations at fixed locations with the support of a large mobile measurement unit (mainly used in case of certain emergency events). The accuracy of the measurement equipment in these units is high as is the cost of each unit, which makes it difficult to scale the solution and make measurements with finer spatial granularity feasible.

With the introduction of the IoT technology, it is now possible to deploy a large number of low cost sensors for a fraction of cost of the current technology. The accuracy of these sensors is not as high as those used in the modern environment measurement stations. However, using a large number of measurement points and intelligent processing of the measurements, it is possible to obtain sufficiently accurate measurements that can be used as initial indicators of the status of environment pollution.

Deployment of the low cost sensors can be done by attaching them to lamp posts or buildings around the city. However, to achieve maximum efficiency in terms of the coverage and of the number of sensors, it is proposed to utilize public transport vehicles, taxis and police cars as the mobile measurement units. By deploying a set of relevant sensors on these vehicles and then taking measurements and tagging them with location as the vehicle is passing through a city, it is possible to cover large areas and obtain environmental pollution maps with high granularity.

As it has been said, the system relies on public transport vehicles (buses, trams etc.) and other public service vehicles (police cars, taxis etc.) to carry various IoT devices across the city. Hence, the system allows:

- Continuous monitoring of several environmental parameters across the area covered by the transport and public service vehicles;
- To exploit the previous data for inferring new types of information such as traffic intensity, and noise prediction.

The public transport vehicles will provide coverage of a large, but fixed area (usually the larger streets). Measurement equipment on the taxis and other public service vehicles will extend coverage to additional areas, i.e. smaller streets not served by public transport. Consequently, the combination of these approaches will enable generation of environment pollution maps with much finer spatial granularity than the current solutions. The accuracy of the individual sensors can be improved by appropriate algorithms for data reasoning and correlation.

An important concern is, of course, to avoid any disturbance to the vehicles and drivers performance, as well as to the citizens as final users of the services these vehicles provide. This criteria will be the basis for the physical and functional architecture definition.

3.2.4. Irrigation

The Precision Irrigation scenario is aimed at complementing automated irrigation systems currently deployed in parks and gardens. It will offer a wide data set acquired in a distributed way, gathering the information of interest from multiple locations within each area.

Current systems manage irrigation through a group of preconfigured programs based on timetables, without considering real-time parameters from each of the areas where usually different species of plants and trees with specific requirements exist.

The WSN precision irrigation provides relevant real-time information to the gardening authorities and park technicians, allowing them to assess the real situation before and after a program has been executed. Therefore, decisions aimed at improving performance and reducing exploitation costs may be taken easily.

Some of the main objectives of precision irrigation service are the following:

- Perform a distributed measurement of several environmental/agricultural parameters in two of the most representative parks in Santander;
- Create reporting tools in order to make all the information available to gardening authorities;
- Develop an alert module capable of informing users when irregular conditions are detected.

Remotely monitor water consumption

- Measure/estimate important KPI's like irrigation applied, water absorbed, water waste, and general savings in water, energy and labour costs.

For example, Pedro is the technician responsible for parks and gardens in Santander. He has updated the irrigation programs for Las Llamas Atlantic Park for the summer season. He modifies these programs according to the period of the year but 2012 is quite a rainy year and the plants and trees in the park need less water than usual. Therefore, after a week executing those new programs, Pedro analyses the soil moisture information given by the USN and the SmartSantander platform using his Android mobile phone and decides to change the current configuration.

It is Friday and Manuel checks the irrigation programs to be executed during the weekend. No rain is expected, but the situation suddenly changes and rain begins to fall on Sunday morning when no one from the gardening department is on duty. As he travels to his parents' village, Manuel receives a SMS informing him that the rain gauges have recorded an important amount of water. As a consequence he connects to the system and stops all the irrigation programs.

Luis, who works for the gardening team, is looking at the information available from the Pereda Gardens using his computer because he has just received an alarm from the system indicating that the soil moisture sensor located in a concrete area close to the south corner of the park has registered some values below the minimum limit. Luis gives a call to Antonio, responsible *for maintaining the irrigation network (pipes, sprinklers…) and asks him to check that area. Some minutes later Antonio finds an obstructed sprinkler and easily solves the problem.*

3.2.5. CityScripts

The main purpose or objective of CityScripts is to integrate in SmartSantander a Web of Things (WoT) scenario in which every object is connected to a pervasive wireless/wired network and can answer directly or by indirection to a HTTP query and return structured data.

Objects like phones, domestic appliances, urban sensor networks, advertisement billboards can become the nodes of the Web of Things.

An important part of the proposal is to enable a personal workspace in which every user can compose public data from city sensors with personal data from their devices/services.

Proposed Scenarios:

Two main scenarios are envisioned in the proposal:

- Consumer scenario: the end-user manages a personal workspace to trigger actions on social networks when some conditions on SmartSantander sensor data occur.
- Business Scenario: Describes how a small business can collect data, make inferences and take decisions according to traffic data.

The aim is to enable businesses (and consumers) to become active actors in the smart city infrastructure to provide better services.

CityScripts will be based upon an existing online service called Paraimpu developed by the partner CRS4 as result of a recent research activity (and available online at **http://paraimpu.crs4.it**) which will be improved in line with the results of the SmartSantander experiments.

CityScripts will use the results of the FP6 IP Hydra, the LinkSmart middleware to contextually integrate different sensors available in SmartSantander. The LinkSmart middleware includes the Context Awareness Framework (CAF) which provides the mechanisms for enabling context-aware applications and services, while being domain-agnostic and easily adaptable. The framework supports the modelling of contexts, using a rule engine in an object-oriented architecture. The rule engine supports Complex Event Processing (CEP). It employs techniques such as detection of complex patterns of many events, event correlation and abstraction, event hierarchies, and relationships between events such as causality, membership, timing, and event-driven processes. Through the CEP, the capability for reasoning over data and events together, as context, is greatly enhanced.

Paraimpu is integrated with existing social networks and enables the import of contacts and friends from them. Moreover, social network profiles can be used as virtual Sensors or as virtual Actuators. For example, Twitter and Facebook are currently seen in Paraimpu as Actuators to post text messages.

The user workspace is currently composed of four areas:

- a list of connected friends. Selecting a friend, it is possible to discover his/her profile and his/her shared things and to bookmark them.
- Sensors/Actuators Palette: it shows the available Sensors and Actuators and allows users to add virtual and real things to the personal workspace. Some sensor and actuator classes are already provided in Paraimpu such as Arduino, Foursquare, GoogleCal. The creation of custom Sensors and Actuators is possible by using the generic Sensor/Actuator palette item.
- Added/Bookmarked Sensors and Actuators: a list of added (using the palette) or bookmarked (shared things) Sensors and Actuators.
- Connections: the connections list with the controls to activate and deactivate each of them.

The following scenarios are envisaged:

Scenario 1:

The end user will be able to select subsets of data according to various criteria (location, type, time, etc.) and these subsets will become first class “objects” in Paraimpu, stored in the user workspace as data sources with a uniform interface enabling the composition with other data sources and the external services. The following scenario will summarise the background of this objective.

> *A restaurant manager wants to keep his regular customers who follow his Facebook page well informed about nearby parking capacity in real time. In this way the restaurant manager hopes to provide an added service to potential customers who may decide to come from their office to his restaurant during the rush hour. The manager connects to the CityScripts Web site and selects from the sensor palette a geographical selection of parking slot sensors in the neighbourhood saving this selection as a data source in his work space called “parkingSlots”. Then, he can create a connection between the selected data source with his Facebook profile using the tool “Connect” in his workspace. Now his contacts can receive the number of slots directly on their mobile or PC and decide to go there by car or by other means.*

Scenario 2:

> *A delivery agency wants a tool to collect data and gather statistics and inferences from traffic flow in the city and daily fuel expenditure. The agency asks a software house to build a software platform to do that. The software house decides to base the system on CityScripts. They start coding plugins to track couriers and to collect data from sensors in SmartSantander. This work would be easy and inexpensive thanks to the pluggable architecture of CityScripts/Paraimpu. The plugins are then deployed in the CityScripts cloud in a virtual server. At the end of the development the agency has a Web user interface where courier’s tracks and traffic data source can be put in context with each other and some information can be mined and new strategies could be devised.*

3.2.6. Soft Actuation over Cooperating Objects Middleware (SACCOM)

SACCOM is a wining proposal from the first SmartSantander Open call focusing on the evaluation of a developed middleware platform (POBICOS) for pervasive environments and the evaluation of a proposed soft-actuation concept on top of this platform through experimental trials with several example services in an indoor environment. SACCOM addresses two key objectives namely experimentation with IoT enabled smart services and application and IoT middleware.

For its experiments, SACCOM will utilise the Smart Building deployment of the SmartSantander facility located in Guildford. The service experiments will use the underlying HW of the testbed for service provisioning and involve a local living lab, recruited from the University of Surrey to study the effectiveness of the proposed concepts. The middleware related experiments will try to utilise a large number of available IoT nodes, which will help to stress-test and mature the SmartSantander testbed service implementations.

In the following, details for each of these experimentations are provided.

Service experimentation:

In terms of service related experiments, the proposal aims to validate 3 soft actuation scenarios, namely a heat savings application, an ergonomic lighting application and an energy savings application, all realised on top of the Guildford site of the SmartSAntander testbed facility. The experiment aims to involve 80 users for the study at the University of Surrey.

The applications aim to provide users with feedback on their behaviour through soft actuations using desk lamps deployed in their environment. Through these soft actuations (e.g. realised by lamps blinking in a certain pattern) users are encouraged to take actions towards a desired behaviour. Wherever possible, results of actions are observed by the IoT nodes. In addition, user feedback from half of the participants is captured through game controllers after soft actuation interventions.

The service requires two extensions to the existing platform (and possibly architecture). The first is the interfacing of actuators to the system. The second is the integration of game controllers to provide end user feedback during experimentation.

Middleware experimentation:

The applications and services are realised on top of a distributed middleware for cooperating objects called POBICOS. The main objectives of these experiments is the scalability testing of this middleware based on an involvement of 200 IoT nodes, with particular emphasis on the responsiveness and scalability of the proxy environment. The experiments will require deployment of the middleware platform on the application servers of the testbed which will host a network of proxies and the deployment of a light weight interaction protocol between the proxies and the IoT nodes. The deployment of code on the GW may be necessary to integrate the lightweight POBICOS interaction protocol with legacy sensor/actuator devices that are non-programmable.

Tests will be performed by dedicated stress test applications and at later stage implicitly through the service experiments above.

3.3. Detailed Requirements for SmartSantander Phase 2

This section lists functional and non-functional requirements for the SmartSantander test facility based on the use cases defined in WP1-T1.1 and WP4-T4.1 which have been prioritized for the current phase of the project. These requirements complete the requirements already addressed in phase 0 and 1, and enhance the architecture specification and the detailed specification of components carried out in WP2. For a complete list of all requirements and their associated status derived so far, please see [IR1.3].

Functional Module	ID	Title	Text
AAA	FR001	Authentication	When the researcher provides his personal credentials to the SmartSantander experimental facility, it must authenticate the researcher.
	FR022	User account management.	At any time the SmartSantander experimental facility shall enable the administrator to grant and revoke user access privileges.
	FR097	Authorization	When a user wants to execute any action, the system has to verify that he is authorized to perform this action.
Application Support Interface	FR176	Retrieval of Sensor Parameters	For CityScripts the following retrieval methods are needed: - for a sensorID get type (temperature, humidity, ...) and measure unit - for a sensorID get its value (current or last with timestamp) - for a sensorID get its position (lat-lon in GPS coordinates) - for a sensorID apply for publish/subscribe (push notification) - push notification for new/delete sensor
Configuration Management	FR061	Node Characteristic Specification	As part of an experiment specification, the system shall allow users to specify the attributes of the sensor nodes (e.g. hardware characteristics, failure patterns and other metrics) they wish to use for the experiments. If specified by the user, these attributes may be used by the system to locate matching nodes. When node reservation clashes occur, these attributes may also be used by the system to locate alternative nodes.

Functional Module	ID	Title	Text
	FR091	Semantic Experiment Specification	There needs to be a way for a testbed user to provide a high level declarative specification of the experiment and its requirements (such as specifications and constraints on the type and number of resources, required topology and corresponding link behavior, target environment, required SW components etc).
	NFR014	Code Quarantine System	When the researcher wants to run an experiment on the SmartSantander testbed, the code to be loaded in the nodes has to be tested before the upload in a controlled and non-critical part of the testbed.
	NFR016	Code Quarantine System	The part of the SmartSantander testbed dedicated to the code quarantine must be representative of the overall SmartSantander testbed in terms of heterogeneity, capacity and manageability.
Experiment Support System	FR150	Broadcast / Multicast OTAP in TR	Testbed Runtime should be capable of supporting a broadcast or multicast OTAP.
IoT Node Requirement	FR169	Virtual Testbed Device	The testbed should allow the creation of Virtual Testbed Devices (VTD) on the GW. VTD provide the same interface towards the testbed as physical device drivers and in addition allow the instantiation of (arbitrary) experimentation code that can link between a physical testbed device and the testbed.
	FR174	Noise Level Sensors	Both CityScripts scenarios require noise level sensors. This requirement should be met.
	FR175	Rain Sensors	Both CityScripts scenarios require rain sensors. This requirement should be met.

Functional Module	ID	Title	Text
IoT Node Requirement	FR177	Calibrated Sensors	The platform has to include a number of well calibrated sensors (of each relevant type) deployed across several areas of interest. These sensors will be used for calibration purposes of other sensors and have to be properly designated so that other system components can discover them.
	NFR007	Mobility of nodes	The system shall integrate and support wireless nodes that can be moved over distance.
Resource Configuration	FR171	Traffic isolation	The testbed system must provide a way of isolating the traffic of concurrent experiments, in order to avoid interference that negatively impacts ongoing experiments
	FR172	Experimental Conflict Detection	The testbed system must be able to detect wrongly specified experiment configuration of the user, which may cause conflict. This should be ideally done prior to allocation and/or execution of an experiment.
	FR083	Uniform Resource Description	A testbed resource requires a standardised representation within the information model of the SmartSantander facility.
	FR129	HW / SW Inventory Database	The system shall support an inventory database where it shall keep track of at least the following: all devices in the network and their position, device vendor, services they are being used for and functionalities/information they can provide, their hardware configuration, release of the software installed on each of them.
Resource Directory	FR092	Semantic discovery of valid testbed resource configurations	When the experiment requirements are provided, the system processes the declarative query and performs semantic matching in order to provide a corresponding mapping with the resources satisfying the requirements.

Functional Module	ID	Title	Text
Resource Discovery	NFR031	Use of devices in different testbeds	The users of one test-bed shall have information about the available resources in all federated test-beds so that they can be used for experiments.
Service Platform	FR157	Storage of Digital Content	Augmented Reality/Cultural scenario: Services need to know historical information about a tag, e.g., the number of times this tag was read and a statistic of the types of devices that were used to read this tag (OS, device id,…)
	FR158	Registration of devices dynamically / runtime	Participatory Sensing and Augmented Reality/Cultural Scenarios: The registration of devices to the Resource Directory should be done dynamically. In this case when the application starts for the first time a registration should be done in the Resource Directory and further to the USN.
	FR159	Deregistration of devices dynamically /runtime	For Participatory sensing and augmented reality applications, when the users uninstall the application or delete their service accounts, the device status must be updated (deactivated or deleted) in both the Resource Directory and USN.
	FR160	Access to historical data	Participatory Sensing and Augmented Reality/Cultural Scenarios: The users can access historical data for instance regarding physical sensing.
	FR161	Store physical sensing messages	Participatory Sensing and Augmented Reality/Cultural Scenarios: Periodically, mobile devices send physical sensing messages that contain sensing data, being these messages sent from the PSens Server to the USN, and typically have the structure of a standard observation message plus the GPS position of the mobile node when the measurement was taken.
	FR162	REST Interface on UAS	All Service Scenarios: It is necessary to define a specific REST interface to allow the communication between UAS and PSens Server

Functional Module	ID	Title	Text
	FR163	UAS Web interface for PC users	It is necessary to define a specific web interface to allow the users their provisioning on UAS
	FR164	API for raw data to SensorML representation	All Scenarios: The USN should provide API in order to create a SensorML representation starting from the raw sensor data.
	FR165	REST Interface on USN	All scenarios: It is necessary to define a specific REST interface to allow the communication between USN and data control centre Server
	FR166	Location with measurements	It is necessary to store location with each measurement.
	FR167	Device management	Eco Bus: It is necessary to provide mechanisms for monitoring of traffic sent over the mobile network to ensure compliance with agreed data transfer policy with the mobile operators.
	FR168	Proxy for mobile devices	Eco Bus: Due to the restrictions imposed by mobile networks (no public, routable IP addresses allocated to mobile devices), it is not possible to initiate an IP connection with a mobile device from a remote location. Therefore, it is necessary to have proxies (implemented on a server, with public URL) for each mobile device that will receive measurements continuously/periodically from the devices, store them in a database and represent the devices on Internet.
	FR168	Ability to send SMS to mobile devices	Eco Bus: It is necessary for the platform to have the ability to send SMS messages to mobile devices in order to initiate some actions on the devices or update certain configuration parameters.

Functional Module	ID	Title	Text
	FR135	Tracking of mobile nodes	The system must provide the tracking of mobile nodes, offering a dynamic map with the position of these nodes and also allowing the estimation of future positions through geographic positioning tools. The system must be able to provide the position of buses, taxis, and eventually bicycles, in a real time fashion.

Table 1. List of functional and non-functional requirements

4. SECOND CYCLE SMARTSANTANDER ARCHITECTURE SPECIFICATION

This section provides an overview of the second cycle architecture specification. It represents a revision of the initial architecture for the SmartSantander experimental facility that has been introduced in deliverable [D1.1]. The second cycle architecture does not propose to change any functionality or interfaces that have been identified in the initial version. It proposes extensions to the architecture, in order to accommodate newly identified requirements.

The previous section captures all remaining or new requirements that have emerged during the second phase of the project. Some of these requirements have direct impact on the high level architecture and result in the aforementioned extension. The extensions are introduced in the context of the overall architecture in section 4.1 and further details are given in section 4.2.

Many of the requirements do not have direct high level architectural implications but have an impact on how functions of the architecture are realized or evolve to accommodate new features. Section 5 provides a more detailed account of these features to serve as the starting points for WP2/WP3 in developing and updating the details of the underlying specifications and corresponding implementations.

4.1. High level overview of the revised SmartSantander architecture

The second cycle architecture specification proposes to keep all existing interfaces and system functions and to introduce two additional system functions and the corresponding functional entities necessary for their realisation:

- Participatory sensing support
- Vitrual testbed device (VTD) support.

Figure 1 shows an updated architectural diagram, highlighting in red colour the proposed architectural modifications.

Motivated by the requirements FR158-162, the participatory sensing support is realised by two new functional entities which are part of the application subsystem: the *PSens server* and the *PSens client,* as well as a *PSens data base (DB)* that holds required state information. The *PSens server* is located on the server tier of the testbed architecture and exposes its functions towards application and services via the application support interface (ASI). The *PSens client* is located on the nodes of the GW and IoT node tier and interacts with the *PSens server* entity to support exchange of participatory sensing information.

The virtual testbed support is motivated by requirement FR169. It is realised by new functional entities in the experimental support system. It provides a configuration function towards an experimental user via the experimentation support interface (ESI) in order to allow full life-cycle management of VTDs. Functional entities called *VTD config* in the diagram below are located at both server and GW tiers.

All other functional entities that have been identified as part of the first cycle architecture remain intact and are not impacted by the two proposed extensions. Section 4.2 provides an initial description of the newly introduced system function. This overview is complemented by a more detailed discussion in section 5.

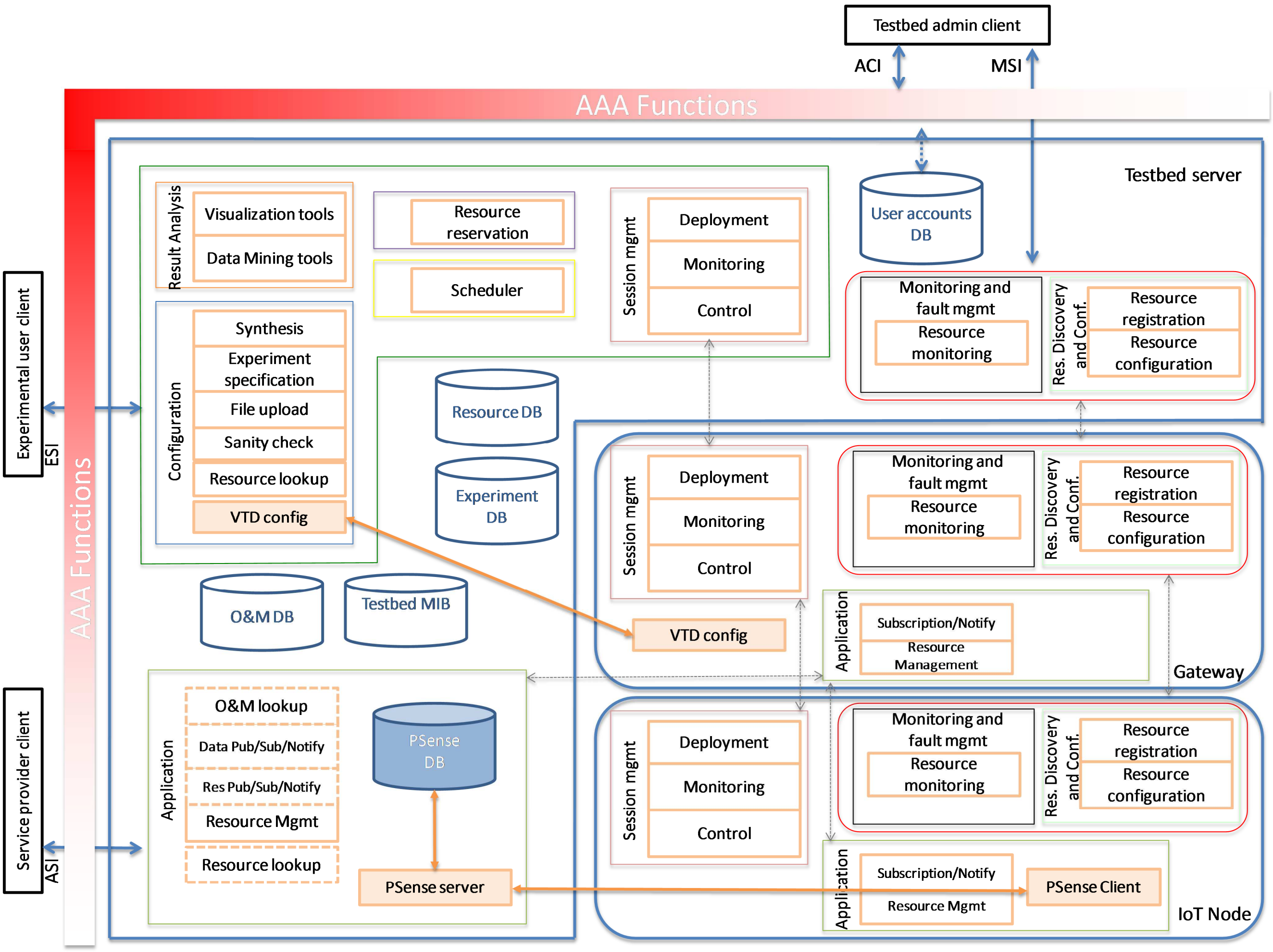

Figure 1: Overview of updated SmartSantander architecture.

4.2. Specification of new system functions

4.2.1. Participatory sensing support

Conceptual overview

For supporting a participatory sensing approach we consider development of a new component called the PSens Server. This component interfaces with the applications allowing the management of devices in the SmartSantander platform, as well as, supporting submission of observation messages. The added functionalities are the following:

- Support for dynamic device management (device registration and deregistration).
- Support for submission of observation messages (both related to physical sensing and application related observations)
- Support for topic-based publish/subscribe functionality, allowing the users to receive events related to specific topics.
- Support for retrieving historical device data (observations historical).

The following diagram shows the interaction between the components:

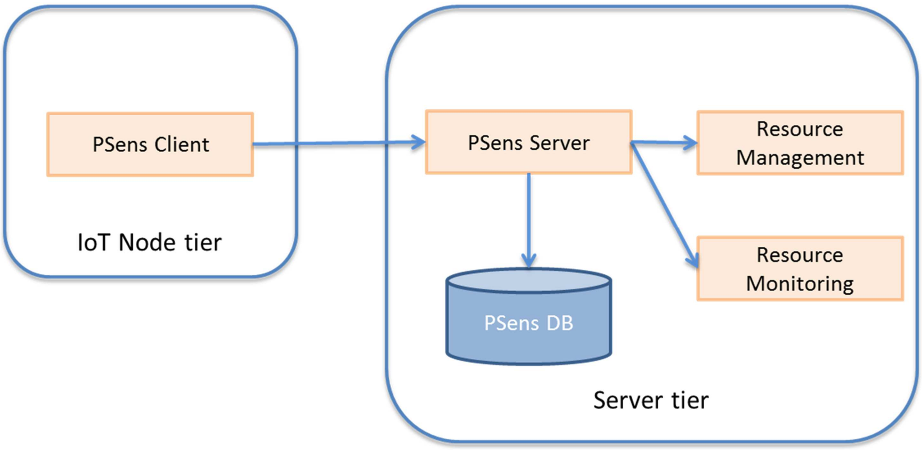

Figure 2: Participatory Sensing Management.

For device management the PSens Server receives from the application the device information to be registered, stores/deletes/updates the device information into its local database and sends a node registration/deregistration message to the EventBroker component of the SmartSantander platform.

For submission of observation messages, the PSens Server receives the observation data from the applications and sends this information to the USN component, which stores this information.

Applications can subscribe/unsubscribe to specific application related events. Subscriptions information is stored in the PSens Server local database.

For retrieving historical device information the PSens Server communicates with the USN component.

Further design considerations

Realisation of the new component and its functions is addressed in the ongoing work in WP2, WP3 and WP4. For example, device status updates sent from the PSens Server, as well as, the definition of a new node type (PSens Node) and its characteristics are being considered.

In WP4, the possibilities of adding new application features that are useful to the users are being considered always taking into account and considering the possible architecture changes or adaptations.

One of the open issues is storage of data related to the application observation (e.g. pictures), which is currently not possible to store in the USN component.

4.2.2. Virtual testbed device support

Conceptual overview

The virtual testbed device (VTD) support represents a new system function of the experimentation support subsystem. It allows the testbed users to create a set VTDs as part of an experiment configuration. These VTDs will be available at specific GW nodes for the experiment. Analogous to physical testbed devices, a VTD that can be managed in the same way as the physical testbed devices from the testbed management framework. VTDs are instantiated on the GW tier of the testbed and will be treated as devices belonging to the IoT node tier. VTDs can extend the capabilities of the testbed by the following features:

- Allow instantiation of emulated sensors or actuators without supporting HW on the GW (e.g. mock devices for debugging purpose or scaling up of experiments)
- Extend capabilities of the existing IoT nodes with GW resources (processing or storage) to emulate more powerful sensor nodes (e.g. parts of more complex machine learning algorithms on resource constrained sensor nodes can be offloaded to the GW device)
- Make it easier to realise composite sensor and actuator control loops in GW devices (Virtual Sensors) and expose those to the testbed framework
- Integrate into TR on the fly new HW not currently supported by the WSN device drivers (legacy hardware such as weather station and game-pad controller).

Figure 3 shows how the newly proposed *VTD config* function handles the dynamic deployment management of VTD, which has to be more flexible than the one for the more manual deployment of hardware IoT nodes.

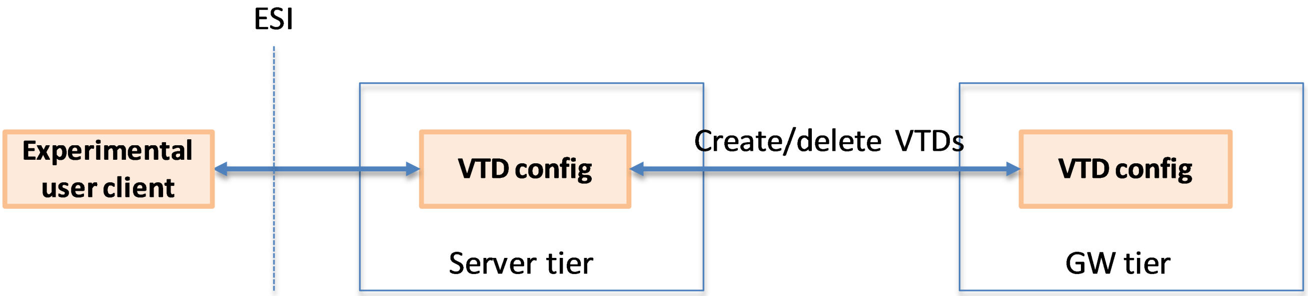

Figure 3: Support for the life cycle management of Virtual Testbed Devices.

Through the ESI, an experimenter/testbed-user specifies how many VTDs are required for an experiment and at which GW nodes the VTDs should be instantiated. The server side *VTD config* function validates the feasibility of the user configuration requests and triggers instantiation of the

VTDs at the specified GW nodes. An user can also trigger deletion of the VTDs. After an experiment, the system will automatically remove all VTDs associated with the experiment.

Please note that the newly proposed functions only manage the lifecycle of VTDs (creation/ deletion) inside the system. The registration of VTDs with the platform or the interaction with VTDs during experiments, takes place via the same existing framework functions as for physical testbed devices.

Further design considerations

In order to realise the proposed system functions, further specification and implementation work is necessary in WP2 and WP3 in order to adapt the existing WSN device driver architecture to support such VTDs. Some initial considerations for further studies are provided as follows.

The Virtual Testbed Device (VTD) is a generic device that can be instantiated on a GW as any other testbed resource. The VTD can be then considered as a regular testbed resource - it can be selected for an experiment, reserved and used during the experiment itself. For this reason a VTD instance should be reachable through the iWSN API (such as flash, reset, send and receive message using the serial connection).

In order to achieve the functions above, the VTD needs to implement the WSN device driver interface in the same way other physical devices such as iSENSE, WaspMOTE or Telosb already do.

Like IoT nodes, the VTD internal behaviour needs to be specified by provisioning of internal application logic during an experiment. For this reason the VTD should provide an internal container that can be customized at run time with experimentation code.

5. SECOND CYCLE SMARTSANTANDER ARCHITECTURE REALIZATION

5.1. Horizontality

5.1.1. Reservation of nodes that provide services

The testbed management software used in SmartSantander is called *Testbed Runtime,* inherited from the WISEBED project.

Authorization was a minor issue in WISEBED as all testbeds were used for research and experimentation only and all participating parties allowed full access on their test beds to each other as well as external users. If a user had reserved a set of nodes and is authenticated to a WISEBED testbed via the Testbed Runtime he/she immediately had the rights to flash and reset any node contained in the reservation. This is not suitable for SmartSantander where public services, which may not be interrupted by experiments, are provided by parts of the testbed in addition to the research and experimentation.

In such environment, a fine grained authorization scheme is needed to allow or deny access to and reservation of specific sensor nodes (FR001, FR022, FR097). This fine grained authorization scheme for the testbed usage is realized in the 2nd cycle architecture of the SmartSantander project.

In the SmartSantander project sensor nodes providing services to the public (environmental data, parking space occupation) are called *service nodes*. Users of the SmartSantander test bed are separated into *experimenters* and *administrators*. Experimenters are granted read-only access rights to the service nodes while administrators may reserve, flash and reset all nodes at any time. Thus experimenters may not interrupt the service provision.

Nodes with mixed usage (services and experimentation) may only be flashed with certain images which do not disrupt the service provision.

To realize the fine grained authorization, the existing SNAA component of the Testbed Runtime is extended by a database containing the users, roles and access rights. Authorization functionality ist then added to the depending components like the Session Management.

Furthermore the Session Management component (SM) is extended so that it supports multiple read-only sessions which allow concurrent access to the same set of nodes. This will give the experimenters read-only access to service nodes. They can use data from the service nodes without having to ‘reserve’ them.

With the inherited WISEBED software this can only be achieved via a workaround: To share outputs of an experiment, the experimenter who reserved a set of nodes might share the secret reservation key with other users, which allows the other users to access the same nodes. However, this other users may interfere with the actual experiment since the right to flash or reset these nodes is granted to whoever possessed the appropriate secret reservation key.

Allowing for fine grained authorization and read-only sessions will solve this. The output of an experiment, e.g., the current temperature or the occupancy of a parking area, will be available to experimenters who did not reserve the corresponding nodes. But these experimenters will not have the possibility to interfere with the original experiment or service.

5.1.2. Service/Experimentation concurrent operation

The capacity to support experimentation at the same time as service provision, implies that the modification and improvement of the Application module of the IoT Node is needed. In particular, it has to support functionality of the two radio interfaces, Digimesh and native 802.15.4, simultaneously, as well as processing data associated with the experimentation, without disrupting the service provision.

At the same time as handling service and experimentation traffic, the Session Management module is modified to be able to process commands associated with network management, thus assuring that the node is in a manageable and reliable mode at all times. Among these network management commands, those as OTAP procedure or reset of a node, can be indicated.

It must be taken into account that some nodes will only be providing service and no experimentation can be carried out over them.

5.1.3. Node availability

As previously commented, all deployed nodes are providing a service. This service provision is associated with transmission of information in a periodic way from the deployed nodes to their corresponding gateways. Taking into consideration this periodical transmission, some additional mechanisms have been implemented at Session Management and Monitoring & Fault Management modules at gateway level, in order to register, maintain and remove nodes in a dynamic way from the Resource Database. For this purpose, Node Manager, IoT Manager and Event Broker (explained in detailed in [D2.2]), have been implemented in order to overhear the service messages sent by deployed nodes, thus registering them at the Resource Database when a new packet is received, maintaining node at the database if this is periodically sending service packets, and removing it from database when no packet is received from it after a determined timeout.

5.1.4. OTAP/MOTAP Management

Over The Air Programming (OTAP) and Multihop OTAP functionalities (FR150) are implemented and integrated within the TR in order to remotely flash nodes as many times and with as many codes as needed. This flashing process can be only carried out by users with appropriate rights (i.e. administration).

To perform the MOTAP process, functionality to flash the reserved nodes belonging to the corresponding gateway is implemented within the Session Management module at the gateway level.

From the IoT Node point of view, the Session Management module must be also modified for receiving and processing MOTAP packets in the correct way, thus assuring the correct operation of the node when the flashing process has finished. Within the fault management module, the corresponding mechanisms to report failures during the whole process are enabled.

In order to run a new experiment, the researcher can reserve nodes through the Resource Reservation module at Portal Server level, thus flashing them with the corresponding code through the corresponding gateways, such that a set of nodes hanging from different gateways will be flashed from the corresponding gateway on which they depend. Once the node has been correctly flashed, new code will be launched and the experiment will take place during the reservation time.

5.2. Mobility

The first phase of the project has been focused on the deployment of fixed environmental monitoring nodes where thousands of sensors have been already installed in the Santander city centre. For the second phase, some of the envisioned scenarios, which are covered in this document, are intended to enlarge and improve the possibilities offered by the current fixed infrastructure, extending its coverage area and enriching the experimentation capabilities through the installation of mobile devices on buses, taxis, police cars and other municipal vehicles (NFR007).

The architectural integration of these new mobile devices within the SmartSantander architecture is shown in the figure bellow:

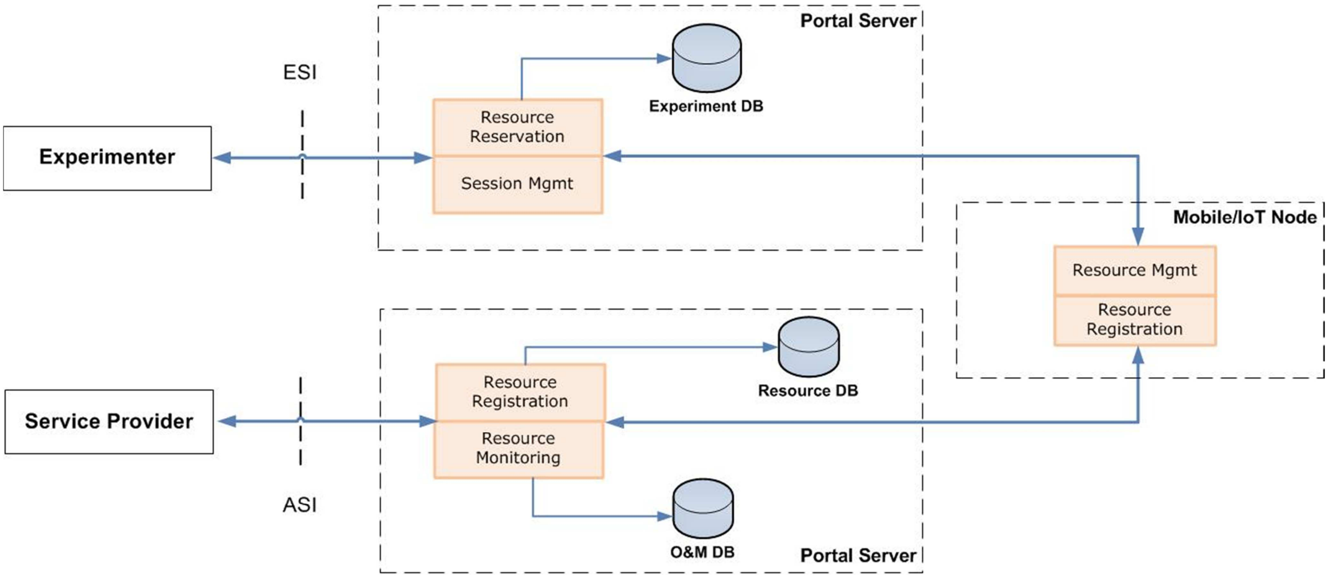

Figure 4: Mobile node integration

As can be derived from Figure 4, the mobility concept does not add special complexity to the system architecture. The most important functional change can lie in the necessity for all the observations and measurements to be tagged with the GPS information associated to the location at which they were gathered. On the other hand, nodes need to incorporate a WAN (GPRS/UMTS) communication module aimed at providing the necessary service connectivity. These new characteristics will lead to additional implementation issues that will be addressed and further explained in [D2.2] but, from the architectural point of view, mobility management does not imply the addition of a new functional block to the architecture. The integration of mobile nodes in the SmartSantander architecture will be performed as follows:

- The deployed nodes will be registered in the Resource Database as mobile nodes. In this sense, these nodes will send periodic measurements on public transportation management information (e.g. vehicle speed, fuel consumption, etc.), as well as information associated with mobile environmental monitoring, including GPS position of the nodes(major information due to the mobile characteristic associated to this type of nodes). The measurements sent will be stored in the O&M Database, while the information on GPS position will enable the update of nodes information in the Resource Database.
- Regarding the experimentation issues, mobile nodes will be able to interact, through an additional IEEE 802.15.4 transceiver, with the already installed fixed infrastructure. This will allow experimentation for a mobile scenario. Apart from enabling the integration of mobile nodes with the existing platform, this radio module will also allow communication with future platforms relying on IEEE 802.15.4 protocol. From the architectural point of view, this interface will be treated in an identical way to the interface already used within the currently deployed fixed network. All the information regarding experimentation is then gathered in the Experiment Database.
- Resource monitoring will be handled by the IoT Manager and the Event Broker (for more details see [D2.2]), and therefore defining the policies for determining a node failure, update or removal from their corresponding databases (Resource database, O&M database and Experiment database). This feature enables taking into account the mobile nature associated with these devices. Issues associated with fleet management must also be explored, in order to handle mobility issues in a more straightforward way, as node failures can be associated not only to a communication loss or a node operation failure, but also to the movement of the nodes beyond the coverage area of the experimentation fixed network.
- Experimenter/Testbed user will access the Testbed server through the ESI in order to reserve the corresponding fixed and mobile nodes. Some of them can only be accessed through GPRS interface. Once the experiment is reserved, all the information regarding these nodes is stored in the Experiment database. In this sense, mobile nodes behave in the same way as fixed nodes but the system considers that the nodes will not always be available to interact with the fixed architecture due to mobility (out of coverage area of the static experimentation network deployed in Phase 1).
- Service Provider user will access the platform through the ASI interface in order to get information (stored in the O&M database) associated with a given service running over a set of nodes.
- Node Management Mobility offers the possibility to achieve maximum efficiency in terms of the coverage and the number of sensors without the need of gateways in the system. For this purpose, nodes are equipped with their own WAN physical interface by means of a GPRS/UMTS module. Considering that the service communication interface is different from the one used in the fixed network, a module for interaction with the mobile management

system (fleet management system) able to send and receive information and manage mobility issues, is needed to retrieve and store the corresponding information within the O&M database. This interface will fulfil two different requirements:

- Observations will be directly transmitted to the system backend, through the corresponding communication management module.
- Wireless sensor nodes can be accessed and managed through this WAN interface, thus easing configuration changes, as well as allowing remote monitoring (commands transmission/reception) and OTAP (flashing of the node), using a defined communication protocol.

Depending on the system configuration and necessities, each node can have a persistent connection established with the server (GPRS data are not consumed while nodes stay idle) or just initiate a new one each time updated values are available for transmission. This operation mode will depend on the way the nodes get registered on the Resource Database by the IoT Manager. A connection interruption can be understood as a communication loss with the node, implying the corresponding removal from the Resource Database and its corresponding registration when communication is established again.

5.3. Participatory Sensing

In this second year of the project we will have a look at a Participatory Sensing approach where the end users (citizens) feed the SmartSantander platform with data. The definition of this scenario in WP4 introduces new functional and non-functional requirements to the platform, and thus, the development of new software components that fulfil those requirements. More specifically, we have considered the use of mobile phones as sources of both physical sensing and “social” data through the use of an Android application.

In order to support this scenario, the SmartSantander platform must be able to dynamically register/deregister the devices and its sensors; support the submission of observation messages with the physical sensing data and support the submission of application related observations (“social” data).

In order to support the functionalities mentioned above, we will implement a software component that serves as an interface between the application and the SmartSantander platform. This component is called the PSens Server and communicates with the EventBroker component of the SmartSantander platform in order to register/deregister the mobile devices in the platform and communicates with the USN component in order to submit observations related both to physical sensing and “social” events.

Physical sensing includes acceleration, temperature, luminosity, noise, battery charge, gravity, pressure, linear acceleration, magnetic field, orientation, proximity and angular velocity. Some of these parameters are new and therefore we will need to update the USN component in order to support registration of these sensors, as well as, observation messages related to these parameters.

Another feature in the application allows users to share or report events occurring in the city (Pace of the City events); these events have a specific type, i.e., traffic, accident, sports, problem, etc. Users can also submit a description of the event, event location and a picture. These events are application specific, and also need to be registered to the SmartSantander platform. While the different types of sensor information are considered as device capabilities, these are seen as application capabilities. The Pace of the City Events are also sent to the USN in the form of observation messages by the PSens Server. After receiving the event data, the USN will then notify the Pace of the City Server, which will trigger the Universal Alert System (UAS) that is in charge of notifying the several users that are subscribed to that specific type of event.

The following diagram shows the interaction between the involved components:

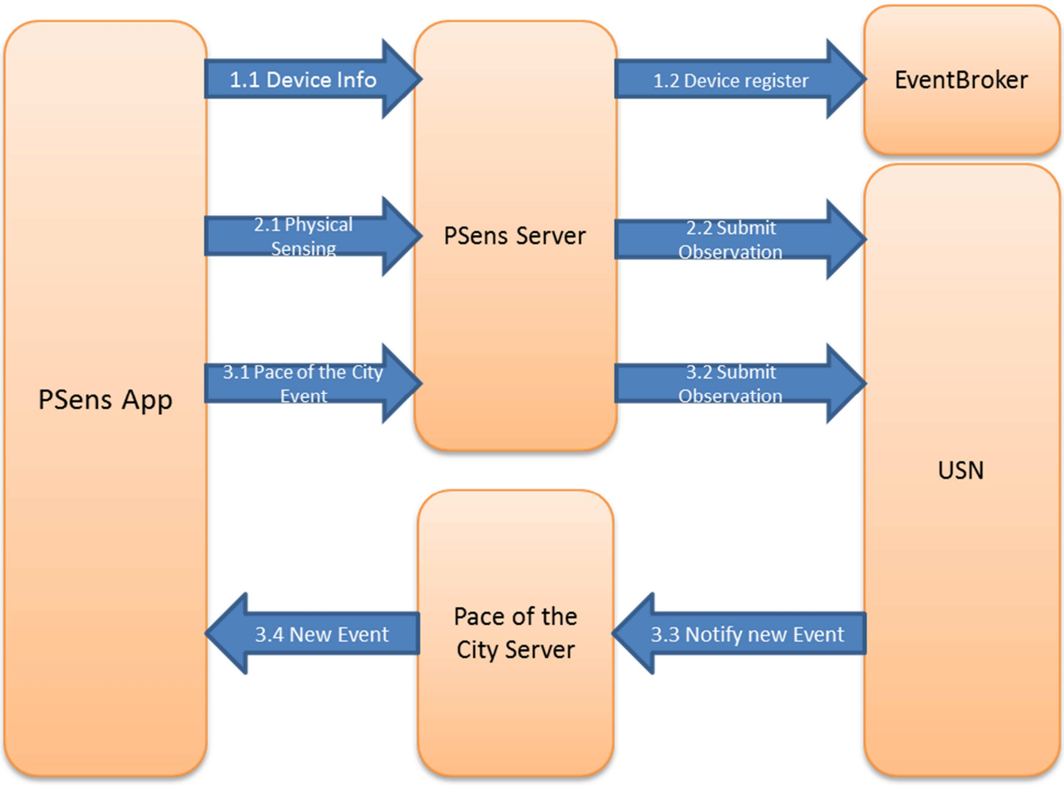

Figure 5: Participatory Sensing Overall Architecture.

5.4. Middleware Testing Support

The current SmartSantander architecture allows testing of different IoT middleware solutions without major modifications. In order to support this claim, we will describe several examples of how experiments with different middleware solutions can be realised.

CITYSCRIPTS Middleware case

GENERAL: CityScripts needs an API to receive data produced by an arbitrary sensor. SmartSantander disposes of a publish/subscribe API. By using SOAP messages, it is possible for the CITYSCRIPTS to register its middleware as a listener of sensor events.

ABSTRACT REGION: CityScripts requires that all the sensors in a given region act as a single Virtual Sensor where multiple data are aggregated with a mathematical formula. In this respect SmartSantander testBed should implement a Virtual Sensor Infrastructure and should provide the GPS position of installed sensors. As it is not possible for SmartSantader to implement geographical queries, CityScripts will implement a local index with (latitude,longitude,sensorID) tuples.

CONTEXT-AWARE: CityScripts requires that different sensors (measuring different physical quantities) are grouped into a single virtual sensor entity and all sensor values are packed in a tuple instance to be processed by the internals of Hydra and Paraimpu. This should be attained with the Virtual Sensor Infrastructure.

POBICOS Middleware case

The following section briefly describes how the POBICOS middleware can be testbed on top of the SmartSantander testbed site in Guildford. Figure 6 provides a highlevel view, showing where POBICOS software components are to be deployed and how they interface or interact via the testbed runtime (TBR)..

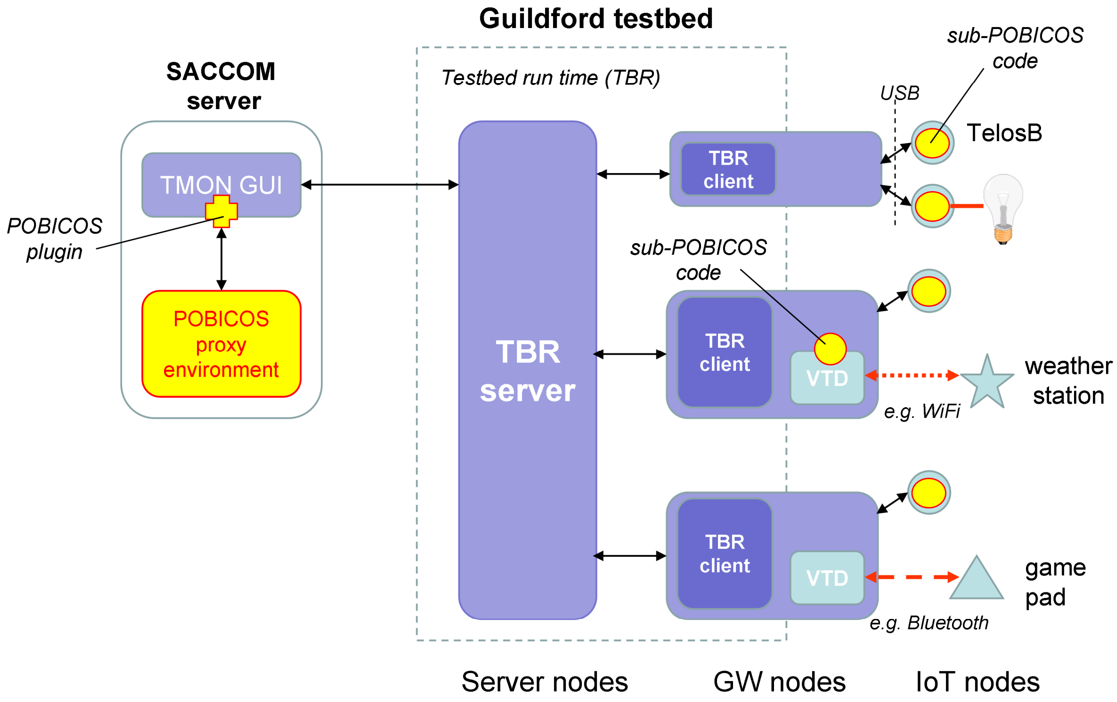

Figure 6: Integration of the POBICOS middleware into the existing testbed architecture.

The POBICOS assumes that each node features a middleware layer that supports POBICOS protocols and POBICOS API. However, this middleware layer cannot be placed on nodes with limited computing capabilities such as TelosB nodes. To address this problem the POBICOS platform introduces a proxy node environment which is installed on a server outside the TestBed infrastructure and acts as the execution host for POBICOS proxy nodes. A proxy carefully supports the POBICOS API and therefore enables the transparent execution of the application code, as if it were running on a physical POBICOS node. The communication between a proxy node and a real node is achieved via a lightweight protocol which is called sub-POBICOS protocol. The code which implements the sub-POBICOS protocol is placed on top of real nodes to enable them to act as sub-POBICOS nodes. Using the sub-POBICOS protocol, a sub-POBICOS node introduces its resources (sensors and actuators) to the POBICOS proxy environment and allows the corresponding proxy node to access them in a flexible way.

In principle, a functional and operational deployment of the POBICOS platform on top of a testbed infrastructure requires node reprogramming capability and bi-directional communication between the proxy environment and the real nodes. Apart from the glue code which is necessary for the smooth integration of the POBICOS platform and the integration of additional resources which are already described in previous sections, the current testbed architecture fully supports the POBICOS deployment's essential requirements, thus there is no need for additional architectural extensions.

As can be seen in Figure 3 the subPOBICOS code can be configured to directly run on top of the TelosB based IoT nodes at the Guildford testbed site. In addition, the newly proposed VTD functionality allows the integration of legacy sensing systems such as the envisioned weather station into the testbed. VTDs are also used to support the integration of game controller pads as IoT nodes to provide end user feedback.

6. CONCLUSIONS

The report has shown that the SmartSantander experimental facility will have to support a diverse set of new use cases for service provisioning and experimentation. They result in a variety of requirements that have not been captured in the initial architecture specification work. Consequently the initial architecture specification has to evolve during the second development cycle of the project, in order to take into account the emerging needs. Based on detailed analysis of these use cases and requirements the document proposed further extensions to the architecture, and detailed design considerations in the form of high level specifications for the developments of underlying software components and the sourcing of the necessary underlying hardware infrastructure.

One promising finding is that the initially specificied architecture, although developed with a different set of use cases in mind, can cope surprisingly well with the new set of use cases, despite their differences in scope. In fact none of the proposed architectural functions had to be changed or removed. Only a few extensions had to be made.

One of these extension introduces a new application support subsystem for participatory sensing applications and requires additional functional components to be added to the server tier and the IoT node tier. This new subsystem will make it easier for developers to tap into citizens provided information in order to realise more powerful application and service for the city and user community in Santander and the other testbed sites.

The second extension introduces additional management functions on the server and gateway tier that are necessary for the creation and removal of virtual IoT testbed devices on gateways. This will provide experimenters and service developers with an increased level of flexibility by allowing them to use gateway resources for resource constrained IoT device or IoT device emulation with the same interfaces and mechanisms offered by the testbed for current IoT tier resources.

While at architectural level most things are abstract and look less complex to handle, the difficulty resides obviously in the detail of the implementation of the corresponding software and hardware components of the testbed. Many of the requirements derived in this document provide the necessary starting points for detailed specifications of further software (SW) and hardware (HW) components.

In terms of HW the requirements have been effectively translated by WP3 into a set of call for tender documents wich are currently being distributed for public competition. Two of them have been recently released [Calls tenders] while the remainder will follow shortly.

For the SW components, intial high level specification and realisation guidelines have been created which are now further detailed by WP2 and will be implemented in WP3. Various of these detailed specification are already emerging and will be shortly released to the public in [D2.2].

The intial feedback received from both WP2 and WP3 on the proposed architecture evolution and specified guidelines has been so far very promising. It will be exciting to see the architecture and the underlying realisation to be validated during the second cycle of the project through the actual facility use thorugh the service and experimental use cases and to share the lessons learned from those experiences.

7. REFERENCES

[D1.1] ALU-SP (Editor), “D1.2: First Cycle Architecture Specification”, INFSO-ICT-257992 SmartSantander, April 2011

[D2.2] CTI (Editor), “D2.2: Second Cycle Component Design”, INFSO-ICT-257992 SmartSantander, June 2012

[IR1.3] CEA (Editors), “IR1.3: Updated Use Cases and Requirements”, INFSO-ICT-257992 SmartSantander, April 2012

[Technical Review Report] Technical Review Report, INFO-ICT-257992 SmartSantander, period No. 1, from 01/09/2010 to 31/08/2011, 17 October 2011, available at

https://proyectosidi.unican.es/proyectos/tlmat/SmartSantander/Shared%20Documents/Forms/AllItems.aspx?RootFolder=%2fproyectos%2ftlmat%2fSmartSantander%2fShared%20Documents%2fTechnical_Audits%2f17_10_2011%2fReview_Report

[Calls tender] https://proyectosidi.unican.es/proyectos/tlmat/SmartSantander/Shared%20Documents/Forms/AllItems.aspx?RootFolder=%2fproyectos%2ftlmat%2fSmartSantander%2fShared%20Documents%2fWP3%20Building%20an%20Experimental%20Facility%2fTask%203%2e2%2fDocuments%2fPhase%202%20%2d%20Deployment%2fFinal&FolderCTID=&View=%7b55F7C9A9%2d5DD0%2d4ADF%2d860E%2d6FFC6D20AA53%7d