



# iCore

**Internet Connected Objects for Reconfigurable Ecosystem**

Grant Agreement N° 287708

## D6.6 Report on the validation of the final iCore prototypes

<p><b>Version:</b> 1.0</p> <p><b>Due Date:</b> 30/09/2014</p> <p><b>Delivery Date:</b> 10/11/2014</p> <p><b>Nature:</b> Report</p> <p><b>Dissemination Level:</b> Public</p> <p><b>Lead partner:</b> UPRC</p> <p><b>Authors:</b> See contributors table</p> <p><b>Internal reviewers:</b> ALCATEL-LUCENT, CREATE-NET, SIEMENS, THALES, VTT</p>	
<p><a href="http://www.iot-icore.eu">www.iot-icore.eu</a></p>	
	<p>The research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement n° 287708</p>

**Deliverable Title** Report on the validation of the final iCore prototypes

**Deliverable Number** 6.6

**Keywords:** Proof of concept, Prototype , Trial, Technical evaluation, Non-technical evaluation, Smart Home, Smart Office, Smart City, Transportation, Urban Security, Smart Logistics, Smart Hospital, Smart Tour

**Executive Summary:** This deliverable reports on the validation of the final iCore proof of concept prototypes as well as the iCore trials. The document starts from an overview of the different proof of concept prototype and trial environments in terms of functional components and hardware elements. Then results on the technical validation of the various proof of concept prototypes and trials are presented. More specifically, for each proof of concept prototype and trial, a set of test cases is presented. Each test case description includes information on the involved entities of proof of concept prototype and trial environment, the relevant functional requirements, pre-conditions, execution steps, success criteria and finally results of the execution of the test case in the respective proof of concept prototype or trial environment. Finally, results on the non-technical validation of the different proof of concept prototypes and trials are presented.



## Contributors

First Name	Last Name	Affiliation	Email
Marc	Roelands	Alcatel-Lucent	marc.roelands@alcatel-lucent.com
Wolfgang	Van Raemdonck	Alcatel-Lucent	wolfgang.van_raemdonck@alcatel-lucent.com
Lodewijk	Van Hoesel	AMBIENT	lodewijk.vanhoesel@ambient-systems.net
Sebastien	Creiche	ARAGO Systems	Sebastien.creiche@aragosystems.com
Ana	Bildea	ARAGO Systems	Ana.bildea@aragosystems.com
Jorge	Pereira Carlos	ATOS	jorge.pereirac@atos.net
Miguel	Rodríguez Fuentes	ATOS	miguel.rodriguez@atos
Andrey	Somov	CREATE-NET	asomov@create-net.org
Swaytha	Sasidharan	CREATE-NET	ssasidharan@create-net.org
Raffaele	Giaffreda	CREATE-NET	rgiaffreda@create-net.org
Michele	Provera	CRF	michele.provera@crf.it
Sven	Zeisberg	IT21	sven.zeisberg@innotec21.de
Christine	Noack	IT21	christine.noack@innotec21.de
Andrea	Parodi	M3S	a.parodi@m3s.it
Joachim	Teutsch	SAG	Joachim.Teutsch@softwareag.com
Dan	Puiu	SIEMENS	dan.puiu@siemens.com
Lucian	Sasu	SIEMENS	lucian.sasu@siemens.com
Stéphane	Ménoret	Thales CS	Stephane.menoret@thalesgroup.com

Hu	Willy	Thales TS	Willy.hu@thalesgroup.com
Fabien	Flacher	Thales TS	Fabien.flacher@thalesgroup.com
Nikolaos-Stylianios	Loumis	UniS	n.loumis@surrey.ac.uk
Stylianios	Georgoulas	UniS	s.georgoulas@surrey.ac.uk
Stefan	Meissner	UniS	s.meissner@surrey.ac.uk
Vera	Stavroulaki	UPRC	veras@unipi.gr
Panagiotis	Vlacheas	UPRC	panvlah@unipi.gr
Dimitris	Kelaidonis	UPRC	dkelaid@unipi.gr
Panagiotis	Demestichas	UPRC	pdemest@unipi.gr
Evangelia	Tzifa	UPRC	etzifa@unipi.gr
Georgios	Poulios	UPRC	gpoulios@unipi.gr
Vassileios	Foteinos	UPRC	vfotein@unipi.gr
Konstantinos	Tsagkaris	UPRC	ktsagk@unipi.gr
Yioli	Kritikou	UPRC	kritikou@unipi.gr
Marios	Logothetis	UPRC	mlogothe@unipi.gr
Dimitrios	Karvounas	UPRC	dkarvoyn@unipi.gr
Aristi	Galani	UPRC	agalani@unipi.gr
Alexandros	Antzoulatos	UPRC	alexant@unipi.gr
Aristotelis	Margaris	UPRC	amargaris@unipi.gr
Aimilia	Bantouna	UPRC	abantoun@unipi.gr
Andreas	Georgakopoulos	UPRC	andgeorg@unipi.gr
Matti	Eteläperä	VTT	matti.etelapera@vtt.fi
Erik	Mademann	ZIGPOS	mademann@zigpos.com
Danny	Schumann	ZIGPOS	Danny.schumann@zigpos.com



## **Table of Contents**

<b>1</b>	<b>Introduction.....</b>	<b>14</b>
<b>2</b>	<b>Proof-of -concept prototype and trial environment .....</b>	<b>15</b>
2.1	Smart home: Living Assistant .....	15
2.1.1.	Overview .....	15
2.1.2.	Components.....	16
2.1.3.	Hardware .....	23
2.2	Smart Office: Easy meeting .....	24
2.2.1.	Overview .....	24
2.2.2.	Components.....	25
2.2.3.	Hardware .....	31
2.3	Smart City: Transportation .....	33
2.3.1.	Overview .....	33
2.3.2.	Components.....	34
2.3.3.	Hardware .....	35
2.4	Smart City: Urban Security .....	37
2.4.1.	Overview .....	37
2.4.2.	Components.....	40
2.4.3.	Hardware .....	46
2.5	Smart Business: Supply Chain Management and Logistics .....	49
2.5.1.	Overview .....	49
2.5.2.	Components.....	50
2.5.3.	Hardware .....	56
2.6	Smart Tour in the City trial .....	57
2.6.1.	Overview .....	58
2.6.2.	Components.....	60
2.6.3.	Hardware .....	71
2.7	Smart Hospital Trial .....	71
2.7.1.	Overview .....	71
2.7.2.	Components.....	72
2.7.3.	Hardware .....	76
2.7.4.	ZIGPOS Localization Technology .....	77
2.7.5.	Energy Consumption Data .....	78
2.7.6.	OTAU.....	81
2.8	Smart Theme Park Trial .....	81
2.8.1.	Overview .....	81

2.8.2.	Components.....	82
2.8.3.	Hardware .....	83
<b>3</b>	<b>Technical validation.....</b>	<b>86</b>
3.1	Smart Home: Living assistant .....	86
3.1.1.	Test case #1 – Dynamic CVO creation.....	86
3.1.2.	Test case #2 – CVO Self-healing .....	99
3.1.3.	Test case #3 – Knowledge-based CVO creation .....	103
3.1.4.	Test case #4 – CVO Coordination .....	110
3.2	Smart Office: Easy meeting .....	113
3.2.1.	Test case #1 – Smart Meeting Organisation .....	114
3.2.2.	Test case #2 – Smart Meeting Guidance.....	116
3.2.3.	Test case #3 – Smart Meeting Break.....	120
3.2.4.	Test case #4 – Smart Meeting Recording.....	123
3.2.5.	Test case #5 – Smart Meeting Wrap-Up .....	127
3.3	Smart City: Transportation .....	129
3.3.1.	Test case #1 – The iCore Services are available to the OnBoard Unit.....	130
3.3.2.	Test case #2 –The Parking availability information is correctly and timely sent to the OnBoard Unit – case “Parking Around Me” .....	132
3.3.3.	Test case #3 – The User Behaviour learning module is able to correctly recognize and store recurrent trips/destinations .....	135
3.3.4.	Test case #4 – The User Behaviour prediction module is able to correctly infer recurrent trips .....	137
3.3.5.	Test case #5 – The CVO are available for execution based on identified recurrent User destination.....	140
3.4	Smart City: Urban security.....	142
3.4.1.	Test case #1 – Areas of interest video overview (video C2 GUI).....	143
3.4.2.	Test case #2 – CBRNE monitoring through detections with deployed chemical sensors (CBRNE C2 GUI) .....	151
3.4.3.	Test case #3 – VIP tracking and control (video C2 GUI) .....	152
3.4.4.	Test case #4 – Crowds monitoring (video C2 GUI).....	156
3.4.5.	Test case #5 – Toxic cloud monitoring by people effect through video cameras (video C2 GUI) .....	159
3.5	Smart Business: Supply Chain Management and Logistics .....	162
3.5.1.	Test case #1 Transportation reports .....	162
3.5.2.	Test case #2: Early warnings .....	165
3.5.3.	Test case #3 AC unit control based on container internal conditions .....	169
3.6	Smart Tour in the city trial.....	171
3.6.1.	Test case #1 – Retrieval of best route according to live traffic in Athens.....	171
3.6.2.	Test case #2 – Derivation of real world information (Hadoop part in iCore platform).....	175
3.6.3.	Test case #3 – Dynamic Composition of Virtual Objects.....	176

3.6.4. Test case #4 – Smart Santander acquisition 3 – Dynamic Composition of data through FI-WARE GEs – (VOs and CVOs) Virtual Objects .....	179
3.7 Smart Hospital trial.....	184
3.7.1. Test case #1 – Localization of assets.....	184
3.8 Smart Theme Park trial .....	193
3.8.1. Test case #1 The highlights data corresponding to a round of ride could be precisely retrieved. 193	
3.8.2. Test case #2 The highlights data of a customer should be correctly retrieved. ....	195
<b>4 Non-technical validation .....</b>	<b>197</b>
4.1 Smart home: Living Assistant .....	197
4.2 Smart Office: Easy meeting .....	198
4.3 Smart City: Transportation .....	199
4.4 Smart City: Urban security.....	200
4.5 Smart Business: Supply Chain Management and Logistics .....	200
4.6 Smart Hospital Trial .....	202
4.6.1. Stakeholder Description .....	202
4.7 Smart tour in the city trial .....	203
4.8 Smart Theme Park trial .....	204
<b>5 Conclusion .....</b>	<b>206</b>
<b>6 References .....</b>	<b>207</b>
<b>7 List of Acronyms and Abbreviations.....</b>	<b>208</b>
<b>8 Appendix: Questionnaire used for non-technical validation .....</b>	<b>210</b>
<b>iCore Stakeholder interview guide .....</b>	<b>210</b>
Part 0: Introduction .....	210
Part A: Stakeholder jobs, needs and wishes.....	211
Exploring use and opportunities of data, information, knowledge .....	211
Part B: iCore value proposition .....	212
Part C: Compliance with stakeholder’s needs and wishes (socio-economic requirements) .....	213
Part D: Competing solutions.....	213
Part E: Acceptance.....	213
Part F: Conclusion.....	213

## List of Figures

Figure 1: High-level view of Smart Home proof of concept prototype and supported operations .....	15
Figure 2: VO Visualizer with Smart Home VOs. ....	23
Figure 3: iCore architecture instantiation for the Smart Office: Easy Meeting use case.....	25
Figure 4: Graphical User Interfaces for Smart Break Recommender Situation Observer .....	29
Figure 5: Android Tablet.....	32
Figure 6 : Sony Xperia S .....	32
Figure 7 : HTC One S .....	32
Figure 8: iCore Smart City Transportation Demonstrator .....	33
Figure 9: End of Year 2 Smart City Transportation Prototype Implementation .....	36
Figure 10: The Uconnect™ Onboard Unit front-end on FIAT 500L.....	36
Figure 11: Overview of Urban Security setup.....	38
Figure 12: Urban security OGC / CoAP based VO level software components .....	40
Figure 13: iCore architecture VO level functions and urban security implementation .....	41
Figure 14: Connected devices compliant with OGC URN .....	42
Figure 15: Network flow priority control leveraging RWK-based situation prediction in Service instances .....	44
Figure 16: Close view of the Parc des Exposition entrance, without any virtual actors.....	45
Figure 17: WiSGate hardware main components .....	46
Figure 18: WiSGate runs the customer applications under Linux OS.....	47
Figure 19: WiSMote RF - hardware architecture.....	47
Figure 20: Urban Security use case – ARAGO wireless platform at functional level .....	48
Figure 21: Urban Security use case – ARAGO WisCore wireless platform picture .....	48
Figure 22: The GUI used by the transport company clients for registering a transportation request. ....	50
Figure 23: The GUI used by the transport company employees for registering the binding of a sensor to a package.....	51
Figure 24: Choosing the base forecaster for the time series forecasting model.....	52
Figure 25: Choosing models' hyperparameters.....	52
Figure 26: Set of CVO templates for predicting temperature .....	54
Figure 27: Dispatcher Dashboard showing truck position with temperature values .....	55
Figure 28: Wireless temperature sensor deployed close to temperature sensitive product.....	57
Figure 29: A wireless sensor network gateway device .....	57
Figure 30: View of additional/enhanced components for Smart Tour in the City trial .....	60
Figure 31: Interactions between additional components in Smart Tour in the City trial .....	61
Figure 32: Smart Tour in the City Android application - Navigation Section .....	62
Figure 33: Smart Tour in the City Android application - Points of Interest Settings.....	63
Figure 34: Smart Tour in the City Android application - Map Section .....	64

Figure 35: Smart Tour in the City Android application - Point of Interest information window .....	64
Figure 36: Smart Tour in the City Android application - Route calculation after clicking on the information window of a point of interest .....	64
Figure 37: Smart Tour in the City Android application – (a) Instructions Section (Santander); (b) Instructions Section (Athens) .....	65
Figure 38: Smart Tour in the City Android application -Weather Section .....	65
Figure 39: Smart Tour in the City Android application -Feedback Dialog.....	66
Figure 40: Smart Tour in the City Android application – Weather information display .....	66
Figure 41: Bridging iCore with Smart Santander Architecture Overview - Smart Santander CVO .....	69
Figure 42: Formatting HDFS and starting Hadoop daemons .....	70
Figure 43: Browsing HDFS location (through Eclipse) .....	70
Figure 44: Architectural Overview of the iCore components used in the Smart Hospital Trial.....	72
Figure 45: AR Model .....	75
Figure 46: Interactions between the Android application and other software components.....	76
Figure 47: Wireless sensor node used in the Smart Hospital trial: (a) hardware architecture and (b) photo of the node.....	77
Figure 48: Architecture of Smart Theme Park trial based on iCore.....	82
Figure 49: RFID scanning terminal.....	84
Figure 50: Data collection terminal .....	84
Figure 51: Digital camera.....	84
Figure 52: Infrared sensor .....	85
Figure 53: Service request triggers dynamic CVO creation process .....	93
Figure 54: Search for an existing CVO that can fulfil the request.....	93
Figure 55: CVO optimal composition of VOs, according to requested functions and policies .....	94
Figure 56: CVO registration and instantiation .....	95
Figure 57: Service Level execution time vs. Number of Available Service Templates.....	96
Figure 58: Approximation & Reuse Opportunity Detection execution time vs. Number of Available VOs.....	97
Figure 59: CVO Composition Engine execution time vs. Number of Available VOs .....	97
Figure 60: Planner execution time vs. Number of goals.....	98
Figure 61: Planner execution time vs. Levels of plan .....	98
Figure 62: VO Discovery Process execution time vs. Number of Available VOs.....	99
Figure 63: Received Bytes vs. Number of Available VOs .....	99
Figure 64: VO Failure identification.....	102
Figure 65: Triggering of reconfiguration process .....	102
Figure 66: Replacement of problematic VO .....	103
Figure 67: Time required for the CVO Self-healing process .....	103
Figure 68: Service request triggers dynamic CVO creation process .....	108
Figure 69: Search for an existing CVO that can fulfil the request.....	108



Figure 70: CVO instantiation and execution.....	109
Figure 71: Overall CVO Creation/Instantiation execution time vs. Number of Available VOs .....	109
Figure 72: CVO Coordination overview .....	113
Figure 73: Create meeting UI .....	115
Figure 74: New meeting CVO notification.....	116
Figure 75: New meeting generated email.....	116
Figure 76: QR-Code scanning.....	119
Figure 77: System Logging.....	119
Figure 78: Indoor Navigation.....	120
Figure 79: Meeting Organiser Feedback Provider .....	123
Figure 80: Smart Break Outcome .....	123
Figure 81: Meeting recording notification .....	126
Figure 82: Smart Meeting Recording Service Request at Smart Office CVO .....	127
Figure 83: Smart Meeting Wrap-up CVO Handling Meeting Recordings.....	129
Figure 84: Meeting wrap-up notification.....	129
Figure 85: iCore Services available on the Onboard Unit.....	132
Figure 86: Parking Availability Information shown on the Onboard Unit (Parking around Me) .....	135
Figure 87: Comparison between actual and detected trips .....	137
Figure 88: The inferred destination and route are shown to the Driver.....	139
Figure 89: A new Service is available for the Driver .....	142
Figure 90: Smart city urban security use case - video based Control and Command and areas overview .....	150
Figure 91: Smart city urban security use case - CBRNE Control and Command and detected toxic cloud .....	152
Figure 92: Smart city urban security use case - video based Control and Command and tracked VIP .....	155
Figure 93: Smart city urban security use case - video based Control and Command and crowds tracking .....	158
Figure 94: Smart city urban security use case - video based Control and Command and toxic cloud tracking ..	161
Figure 95: The blue line shows the predicted temperature value of the parcel, based on the current input provided by VOs. The second dropdown contains the ID of the temperature sensor attached to the parcel. ....	168
Figure 96: Predicted vs. actual values for the temperature sensor attached to a parcel, for a 10 minute forecasting horizon.....	168
Figure 97: The transportation emulator interface.....	171
Figure 98: Output of Traffic Analyzer CVO – first iteration .....	173
Figure 99: Output of Traffic Analyzer CVO – second iteration .....	174
Figure 100: Traffic Analyzer CVO performance chart.....	174
Figure 101: Hadoop processing time for different volume of data measurements.....	176
Figure 102: CCE processing time for different numbers of available VOs in the case of 10 requested functions .....	178

Figure 103: CCE processing time for different numbers of requested functions in the case of 100 available VOs .....	179
Figure 104: iCore VO Registry public instance.....	180
Figure 105: Pubic instance of iCore VO Container for Smart Tour in the city Trial deployment .....	180
Figure 106: Smart Santander CVO UI Instance .....	181
Figure 107: Pubic instance of iCore CVO Container for Smart Tour in the city Trial deployment.....	181
Figure 108: Average retrieval time for real-time and historical data .....	184
Figure 109: An example of Android mobile application (screenshots). ....	189

## List of Tables

Table 1: Advantages of a Smart tour in the city application developed by exploiting the iCore framework as opposed to other similar solutions.....	59
Table 2: Requirements related to Smart Home test case #1.....	86
Table 3: Requirements related to Smart Home test case #2.....	100
Table 4: Requirements related to Smart Home test case #3.....	104
Table 5: Requirements related to Smart Home test case #4.....	110
Table 6: Requirements related to Smart Office test case #1.....	114
Table 7: Requirements related to Smart Office test case #2.....	117
Table 8: Requirements related to Smart Office test case #3.....	120
Table 9: Requirements related to Smart Office test case #4.....	123
Table 10: Requirements related to Smart Office test case #5.....	127
Table 11: Requirements related to Smart City: transportation test case #1.....	130
Table 12: Requirements related to Smart City: transportation test case #2.....	132
Table 13: Requirements related to Smart City: transportation test case #3.....	135
Table 14: Requirements related to Smart City: transportation test case #4.....	138
Table 15: Requirements related to Smart City: transportation test case #5.....	140
Table 16: Requirements related to Smart City: Urban security test case #1.....	144
Table 17: Requirements related to Smart City: Urban security test case #2.....	151
Table 18: Requirements related to Smart City: Urban security test case #3.....	153
Table 19: Requirements related to Smart City: Urban security test case #4.....	156
Table 20: Requirements related to Smart City: Urban security test case #5.....	159
Table 21: Requirements related to Smart Business test case #1 .....	162
Table 22: The transportation report generated after a product had been transported.....	165
Table 23: Requirements related to Smart Business test case #2 .....	165
Table 24: Requirements related to Smart Business trial test case #1 .....	169
Table 25: Requirements related to Smart Tour in the City trial test case #1 .....	172
Table 26: Requirements related to Smart Tour in the City trial test case #2 .....	175
Table 27: Requirements related to Smart Tour in the City trial test case #3 .....	177
Table 28: Requirements related to Smart Tour in the City trial test case #4 .....	182
Table 29: Relevant requirements for Smart Hospital test case #1 .....	185
Table 30: Accuracy of AR model for different time granularity.....	190
Table 31: Requirements related to Smart Theme Park trial test case #1.....	193
Table 32: Requirements related to Smart Theme Park trial test case #2.....	195

# 1 Introduction

---

The main goals of WP6 were the demonstration of the horizontal properties of the iCore concepts (i.e. showcasing domain independence), the technical validation of the iCore concepts in terms of feasibility and the non-technical validation in terms of usability and business relevance. In order to achieve these goals the iCore concepts were applied in a variety of use cases through the specification and implementation of a set of proof of concept prototypes and trials, instantiating components of the iCore functional architecture and mechanisms (as specified in WPs 2-5).

This document presents the results on the technical and non-technical validation of iCore concepts derived through the implemented iCore proof of concept prototypes and trials. The document starts from an overview of the different proof of concept prototype and trial environments in terms of functional components and hardware elements. Essentially this first part of the document summarizes the final proof of concept prototype implementations, which have been previously described in Deliverables D6.2-D6.5 and in addition presents aspects of the trials implementation derived in WP9 (through M9.2Contribution to the final report D6.6). Furthermore, results on the technical validation of the various proof of concept prototypes and trials are presented. More specifically, for each proof of concept prototype and trial, a set of test cases is presented. Each test case description includes information on the involved entities of proof of concept prototype and trial environment, the relevant functional requirements, pre-conditions, execution steps, success criteria and finally results of the execution of the test case in the respective proof of concept prototype or trial environment.

In addition, results on the non-technical validation of the different proof of concept prototypes and trials are presented. For the non-technical validation interviews with stakeholders were organised with the aim of evaluating aspects such as rapid application development using iCore concepts/architecture/framework, usability of iCore concepts/architecture/framework, technical skills required for developing new applications using the iCore framework, prototype running environment non-functional features (number of users, response time, etc.), open source components (CVOs and SLs), access to data, security and privacy. The Appendix in section 8 comprises the questionnaire used for the non-technical validation.

## 2 Proof-of-concept prototype and trial environment

### 2.1 Smart home: Living Assistant

#### 2.1.1. Overview

This proof of concept prototype showcases iCore self-management aspects through cognitive functionalities in the scope of a Smart Home.

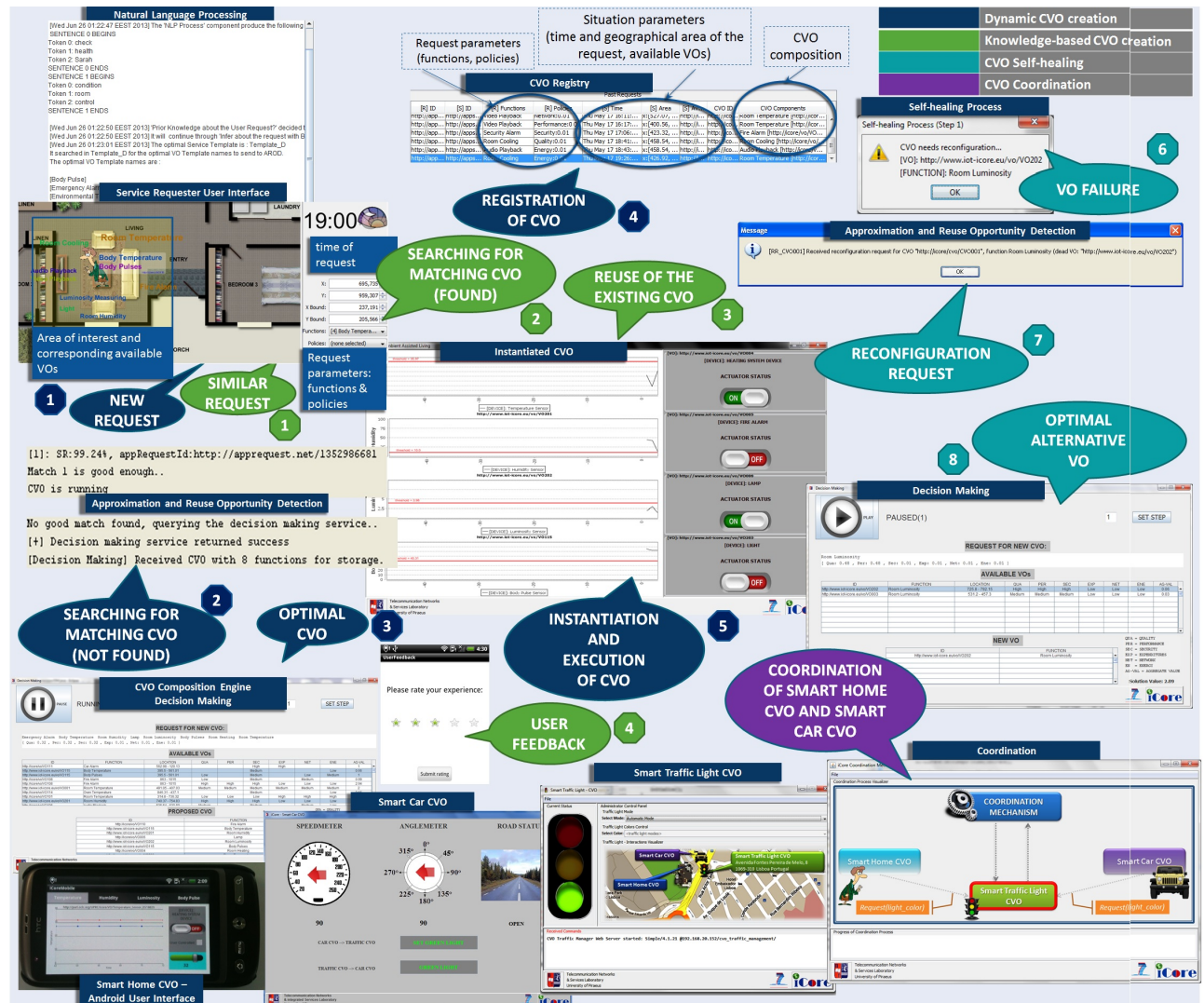


Figure 1: High-level view of Smart Home proof of concept prototype and supported operations

The prototype, apart from software components for the various functional components of the iCore architecture, also comprises Arduino and other hardware platforms combined with various sensors and actuators such as temperature, humidity, luminosity and motion detection sensors, accelerometer, lamp, fan/heating and buzzer. Software technologies used for the implementation include RESTful Web Services, JSON, RDF, SPARQL, Sesame API, etc. The functional components of the iCore architecture implemented in the Smart Home proof of concept prototype enable four main processes, namely; (i) dynamic CVO creation, (ii) knowledge-based CVO instantiation, (iii) self-healing of a CVO and (iv) CVO Coordination.

In the case of dynamic CVO creation a new CVO is created from scratch in case there is no prior knowledge or no suitable CVO is found that can be re-used. The knowledge -based CVO instantiation takes place when an existing CVO can be directly deployed. The self-healing process enables the detection of a problem in the operation of a used VO and enables the dynamic reconfiguration of the corresponding CVO to overcome the problem. Finally, CVO Coordination enables automated dynamic coordination of the simultaneous use of CVOs. The rest of this section outlines the functional components of the iCore architecture as well as the hardware exploited in this proof of concept prototype.

### **2.1.2. Components**

This sub-section provides an overview of the functional components of the iCore architecture developed and exploited in the Smart Home proof-of-concept prototype. Further details on these can also be found in D6.2 [2].

#### **2.1.2.1. Service Level**

##### **2.1.2.1.1. Service Requestor GUI**

This component is intended for the end-user/stakeholder. It has been created with “ease of use” in mind and its purpose is to receive a user request in natural language and then, through interactions with other iCore components, result in the dynamic creation/instantiation of the appropriate Composite Virtual Object (CVO). It has been implemented using a number of technologies; namely: (a) HTML 5, (b) PHP, (c) AJAX, (d) JavaScript, (e) MYSQL, (f) Servlets (Java Web Services) as well as a (g) RESTful interface to facilitate the connection between the Servlets and the iCore infrastructure.

##### **2.1.2.1.2. Service Template Repository**

This component holds a collection of Service Templates that have been drafted by domain experts. A Service Template defines at a high-level the operations, dependencies, and intelligence needed for the automated deployment of a particular service/application. This includes the (generic) types of CVOs and VOs that will be required for realizing these operations and intelligence.

##### **2.1.2.1.3. User Profiling and Preferences**

This component is responsible for obtaining and managing data related to user preferences. This includes the accurate description and representation of this information as well as updating the respective profile information when required. Indicative information includes: (i) the set of available service templates, (ii) the set of services that have been requested/are being used and (iii) an indication of the user preferences associated with a particular service and service template. The functionality for learning user preferences is realized with the use of Bayesian statistics concepts following a similar approach as in [3]. The aim is to estimate the probability of the level of user satisfaction (of a certain user) for a specific service template, given a particular service request, location and time-zone.

##### **2.1.2.1.4. Natural Language Processing (NLP)**

This component allows users/stakeholders to express high-level requirements for particular services/applications in natural language (e.g. “I would like a smart home application”). The



NLP analyses the user request in order to first extract verbs, nouns etc. This is done in order to recognize the “actions” (i.e. verbs) that the user wants to execute onto “which” (i.e. nouns) objects. For this purpose, the NLP exploits Sentence Detection, Sentence Tokenization, Part-of-speech (POS) tagging techniques. The NLP exploits the results of the first analysis so as to derive at a second stage if the system has received a similar user request in the past. This process is done by exploiting semantic similarity/proximity techniques between the current user request and a collection of previous user requests. More specifically, this process results in a score between the current request and each of the previous (natural language) requests. If a score is greater than a specified threshold, it means that there is a similar previous request. Otherwise, the NLP infers that the current user request is encountered for the first time. The output of this process is then utilized by the Service Request Analysis.

#### **2.1.2.1.5. Service Request Analysis (SRA)**

This component aims to select the most suitable Service Template and retrieve corresponding relevant information on CVO and VO types. The SRA exploits the output of the NLP in order to proceed. In particular, if the NLP concludes that there is a similar previous request, the SRA is guided to exploit Bayesian statistics concepts, by taking into account the user preferences from the User Profiling component, in order to infer the optimal Service Template for the current user request. Otherwise, if the NLP infers that there isn't any similar request in the past, the SRA is guided to retrieve a collection of Service Templates from the Service Template Repository component. Then, the SRA mechanism searches in each Service Template and calculates an appropriateness score for each of them by exploiting semantic similarity/proximity techniques. The Service Template which holds the highest score is selected. The outcome is a so called ‘Service Execution Request’ (Fig. 1), which provides information on the type of service logic, CVOs and VOs that will be required to fulfil the service request. This Service Execution Request is forwarded to the CVO Management Unit in the CVO level, described in more detail in the following.

#### **2.1.2.2. Composite Virtual Object Level**

##### **2.1.2.2.1. CVO Registry**

This component stores and provides semantically enriched data on the existing CVOs. This information can be utilized for the re-usability of CVOs and includes information on the context in which CVOs have been created and/or activated. Such context information may comprise the requirements coming from the service level, the time of day, the creation/activation domain, etc. It should be noted that there may be various, CVO registries which may be distributed across several domains, while the information can be exploited by machine learning and knowledge functionality to link the current requirements for a CVO, with previously encountered requirements and consequentially other, existing, VOs and CVOs.

#### **2.1.2.2.2. CVO Factory**

This component is responsible for the instantiation, deployment and activation of a particular CVO, based on the appropriate CVO Template and/or the outcome of the CVO Management Unit functionality.

#### **2.1.2.2.3. CVO Management Unit: Approximation and Re-use Opportunity Detection (AROD)**

The CVO Management Unit is responsible for identifying whether an existing CVO can be re-used as well as composing a new CVO based on respective requirements, information and knowledge on the involved users and the requested service, as well as information in the CVO Template. Various instances of a CVO Management Unit may exist. Based on the provided data, a search is performed in the CVO Registry to discover potentially available, relevant CVOs that can be exploited. Reasoning functionality enables establishing whether certain existing CVOs can be exploited. This is comprised in the so called Approximation and Re-use Opportunity Detection (AROD). The goal of the AROD functional component is to exploit the information in the CVO registry and identify past service requests that match closely enough to the present incoming ones and the situations under which they were issued, so that the CVOs can be re-used and the task of CVO composition from scratch can be avoided under certain circumstances. A set of parameters for the identification and comparison between past and present situations and requests have been defined. The situation parameters consist of the time of day the request occurred, the area of interest (location), and the available VOs at that time, while the request parameters consist of the set of functions requested and corresponding related features to be maximized or minimized. In addition to these, the AROD retrieves performance information (e.g. the CVO execution time, statistics about how energy efficient one CVO is, etc.) from the SK regarding existing CVOs. Requested functions can be matched with approximate ones, e.g. a video capture function can satisfy a requirement for an image capture function. The result of this component is performance gain, as the process of creating a CVO from scratch is more complex, whilst enables the reusability of available resources such as CVOs, enhancing system's functionality. The AROD comprises cognitive techniques, such pattern recognition, used to identify if an already existing CVO can be (re-)used for the current application even in an approximate way, thus reducing the time required for providing a solution for a particular service request. In case a suitable CVO is not found, the AROD triggers the CVO Composition Engine (CCE).

#### **2.1.2.2.4. CVO Management Unit: CVO Composition Engine (CCE)**

The role of this component is to essentially perform an optimization process, the main objective of which is to find the optimal composition of VOs that fulfils the requested functions and policies. It receives as input a set of available VOs, a set of functions for each VO, a set of utility and cost parameter values for these VOs and finally weights for these utilities and costs. Utility parameters provide an indication of positive features of a particular VO (e.g. performance) while cost parameters provide an indication of negative features (e.g. cost, energy consumption). The weights indicate the importance of the various utility and cost parameters and may be obtained through user defined policies. In order to allow for discovering and using also VOs that are not necessarily a perfect match a



correlation matrix is utilized. This matrix provides the suitability level of an offered function of a VO with the requested ones. The outcome of the CCE is the optimal combination of VOs that will form the new CVO. The CCE collaborates with the *Planner* component (see next sub-section), which takes over the dynamic workflow-based development of the CVO Logic, by taking into account the requested functionalities that are included in the Service Execution Request. The CVO Factory is triggered which instantiates and deploys the CVO in the CVO container. After the deployment of the CVO, the corresponding information on the composition of the CVO, as well as the situation under which it was created is stored in the CVO registry. In addition, during the CVO run-time System Knowledge on its performance is gathered, to be exploited for future requests. The CCE is developed as a RESTful web service. Data is XML-formatted/encoded.

#### **2.1.2.2.5. CVO Management Unit: CVO Dynamic Workflow (Planner)**

This component is responsible for deciding the logical connection of the selected VOs. In particular, it provides a graph that describes how input/outputs of VOs are linked, in order to finally achieve a specific goal. Exploiting a well-known algorithm, namely Graphplan (open-source JAVA implementation), it achieves maximum performance in minimum time, enabling the dynamic composition of multiple, diverse VOs into one CVO. Two main phases can be identified in the Graphplan algorithm. Firstly, the algorithm checks whether all goals exist in the current level of the graph, ensuring that there are no conflicts between them (mutex links). Secondly, it checks for a solution in the current graph. If a solution can be found then it extracts it, otherwise it expands the graph adding one more level and the corresponding actions. These two phases are repeated inside a loop process until the algorithm succeeds in finding a solution or decides that there is no solution. The Graphplan algorithm is sound and complete: any plan the algorithm finds is a legal plan, and if there exists a legal plan then Graphplan will find one. The Planner evaluates the VOs selected by the CCE and finds the logical connection that rules them in order to realize the requested service. For this purpose, each of the functions offered by these VOs is characterized through a set of preconditions and effects. This information is retrieved from the VO registry. Taking into account the initial state, the goal (the set of functions that constitute the requested service) and the available functions of the VOs, the Planner can construct the workflow problem. Actions correspond to the available functions of the selected VOs. After defining the problem, the Planner finds the most suitable workflow. At this point, the existence of control loops for each function of VOs should be highlighted. The workflow specifies the interaction among actions while the control loops decide whether predefined requirements are satisfied or not.

#### **2.1.2.2.6. CVO Management Unit: CVO Coordination**

This component aims to eliminate conflicts arising from the simultaneous access and use of CVOs from other entities (e.g. other CVOs). In other words, this component enables dynamic management of the simultaneous access to and use of available CVOs from other entities, such as other CVOs. Specifically, the CVO Coordination component observes the use of available running CVOs by other entities and based on specific priority indicators combined with the current situation, decides on the sequence for the use of the CVO from each user-entity. Consequently, when there is more than one entity simultaneously using/accessing the same CVO, this is monitored by the CVO Coordination component. The CVO

coordination component collects the available information with respect to the current situation, as well as the priority indicators for each entity, aiming to derive a priority list for the CVO use. In this way the best possible management of the concurrently running, inter-related CVOs is achieved, in terms of the avoidance of potential conflicts arising from their simultaneous access and use by other entities.

#### **2.1.2.2.7. Smart Home CVO**

This CVO implements the Smart Home Functionality. That includes automated temperature, lighting adjustment and automated alerting of medical centre as well as a self-healing feature to replace broken sensors.

#### **2.1.2.2.8. Smart Car CVO**

In this CVO basic automated driving functionality is implemented. This is mainly used in conjunction with the Smart Traffic Light CVO to showcase the coordination mechanism.

This component has been implemented using java. To showcase the automated driving functionality an Arduino UNO has been used. Attached to it is a radio control (RC) transmitter corresponding to the RC car that's being used. The car is then automatically driven according to the Smart Traffic Light CVO by sending the correct signals through the transmitter.

#### **2.1.2.2.9. Smart Traffic Light CVO**

This CVO is designed to be the controller of an automated traffic light in a Smart City. In the Smart Home context this CVO is used long with the Smart Car to showcase the coordination mechanism.

#### **2.1.2.2.10. Smart Lighting CVO – (iCore – BUTLER Integration)**

The Smart Lighting CVO is a web-based application that is responsible for the sensing of ambient luminosity values, by using the corresponding VO, as well as the real time location of a human user who is moving around in a place, such as a room of a smart home. The CVO gets the above three measurement values and by using a specific formula, which constitutes its service logic, force decisions regarding the control of light that is available in the place where the user is moving. In particular, the end-user can provide its preferences regarding the light intensity in the room, as well as regarding the distance in which the light should be turned on, since the user is close to the light. The light is turned on if the user is going closer to it than the preferred distance and if the light intensity is lower than the current one that is measured. The CVO has been developed by using responsive website technology and therefore it is available either by PC or by Smart Mobile devices (Smart Phones Tablets, etc.), just by using a web browser.

### **2.1.2.3. Virtual Object Level**

#### **2.1.2.3.1. VO Registry**

The VO registry includes information about VOs that are available in the system. Various entities that are available in the framework can interact with the VO registry so as to perform semantic-based discovery and selection of VOs, as well as to retrieve and/or modify

the corresponding information. Information in the VO Registry describes the associations between VOs, ICT objects and non-ICT objects. Each VO is identified by a “Uniform Resource Identifier” (URI) and contains information on the association with an ICT object, which are also identified by a specific URI. Furthermore, the ICT Objects are classified into various ICT object types (e.g.: sensor, actuator, RFID tag, etc.), have a specific physical location, while they offer one or more functions and are associated with one or more non-ICT objects. Additionally, information on the offered functions and non-ICT objects is also included.

#### **2.1.2.3.2. VO Template Repository**

This component holds a collection of VO Templates. A VO Template can be used to create a particular VO and provides a generic structure for the implementation and semantic description of various VO types.

#### **2.1.2.3.3. VO Creation Tool (VO Factory)**

This component is responsible for the creation of VOs. For this purpose, it interacts with the VO Template Repository to get the appropriate VO Templates in order to instantiate and install VOs. Once a VO is created, installed and deployed it is registered in the VO registry.

#### **2.1.2.3.4. VO Management Unit**

The VO management Unit is one of the main points of interaction of the VO level with the CVO level. It receives a VO Execution Request from the CVO Level from which it extracts the requested VOs as well as requirements regarding their performance for a particular CVO and service execution. The VO Management Unit can retrieve information on available VOs from the VO Registry and also monitors the performance of running VOs with respect to the requirements of various executing CVOs (fulfilling service requests). In case discrepancies in the operation of VOs are observed, the VO Management Unit notifies the CVO Management Unit so as to trigger the appropriate re-configuration actions.

#### **2.1.2.3.5. VO Container**

This component is the execution environment in which the VOs run. It can allow a CVO to receive data from multiple Arduino platforms using a single IP. The VOs can be deployed in the VO container through a user friendly GUI that allows the user to import and upload the VO in the cloud, so as to be accessible remotely, under a specific URI. The VO Container component supports the dynamic deployment of java-based software directly in the cloud and its wrap-up under a Web Service Endpoint so as to be accessible by external users.

#### **2.1.2.3.6. VO Front-End Software and Back-End REST Modules**

The VO Front-End (FE) is responsible for the communication between VO – CVO and the VO Back-End (BE) is responsible for the realization of the communication between VO – ICT object. The VO FE can be developed as one or a set of RESTful Web Services that implements specific functionalities of the VO, such as to provide the measurement data of a sensing process, to retrieve information about VO features, etc. Using a top-level URI that will constitute both the identifier for the VO, as well as the pointer of the VO as a Web Resource that is available in the Internet, it is possible to provide access to external entities which wish to use the VO and its functionalities. An external entity can communicate with

the VO FE, using the URIs that are associated with the VO and its functions. Under each URI different methods of the VO RESTful WS are implemented and executed in order to provide specific functionality to their consumers, while the instantiation of the VO Information Model, which corresponds to this VO, provides data for the description of RESTful WS. On the other hand the VO BE exists in Arduino gateway that is used in the Smart Home PoC. Its purpose is to provide access to the functionalities provided by the ICT Objects connected on the Arduino platform.

#### **2.1.2.3.7. VO Back-End Software REST Module for wireless sensor networks**

The VO Back-End Software REST Module for wireless sensor networks handles functionalities and access of ICT objects connected through standardized and proprietary wireless sensor actor networks (WSAN) like ZigBee, Lightweight Mesh (LWMesh) and ZIGPOS eeRTLS systems.

#### **2.1.2.3.8. Security Agent**

This component is a RESTful web service and it has been implemented in order to ensure some basic security features in the VO level in the Smart Home Proof-of-Concept prototype. Specifically, when a user is registered in the iCore System he is given certain “roles” (indicatively: VO\_OWNER, SIMPLE\_USER, PREMIUM\_USER) that represent his access rights. The user roles are stored using a MYSQL database. These roles are then passed on to the VO Registry component whenever he issues a request. The VO Registry then checks, by means of the Security Agent, whether or not the user has access to each VO he requests.

#### **2.1.2.3.9. VO Visualization**

This component enables visualization of sensors and actuators (e.g. VO functions) the virtual objects expose. Created in WP3 as a part of the Modular VO Container (MVOC) proof-of-concept, it has also been used in the Smart Home use case to display relevant VO sensor values. VO Visualisation is a generic JavaScript/D3.js driven tool which dynamically discovers resources hosted by VOs and adaptively displays the information they hold. It supports historical data queries from the MVOC so that the user immediately receives a thorough view of sensor and actuator value readings. RESTful polling and MQTT (via WebSockets) based pub-sub communication patterns are available. Being browser based, the tool runs on multiple modern platforms. An example of the VO Visualization output can be seen in Figure 2. The visualization is interactive as it provides the possibility of zooming in and out of the data.

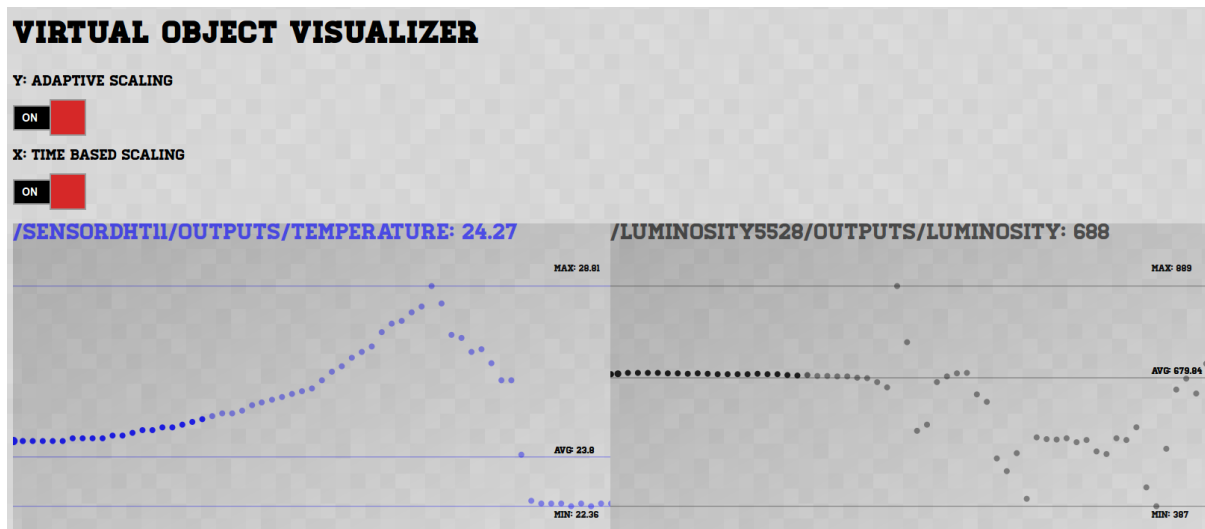


Figure 2: VO Visualizer with Smart Home VOs.

### 2.1.3. Hardware

This sub-section provides a summary of the hardware exploited in the Smart Home proof-of-concept prototype. Further details on these can also be found in D6.2 [2]. Various available sensors/actuators are distributed on several hardware platforms to

For the demonstration of the Smart Home PoC features a number of sensors and actuators are used through multiple available hardware platforms (Arduino, WaspMote, RaspberryPi, Flyport etc...) showcasing hardware independency of the iCore architecture through the use of VOs representing ICT objects. The exploited sensors Motion Sensor, Luminosity Sensors, Temperature and Humidity Sensor, Accelerometer Sensor, LED Lights, Fan, Buzzer, PowerPlug.

The Arduino platform is currently used to acquire the sensor data in the Smart Home PoC. More specifically, 3 Arduino UNOs and 1 Arduino MEGA have been used. On each Arduino an Ethernet shield is attached to enable network on the devices. Combined with the web server that has been developed for the Arduino platform this allows the retrieval of sensor data using a RESTful Interface. An additional hardware platform that has been experimented with in the Smart Home proof of concept prototype is the WaspMote platform which enables sensing of temperature, humidity and light, through wireless connectivity. Moreover, a Flyport platform was exploited. The core element of the platform is a PIC microcontroller (MCU) by Microchip. Communication capability is realized by Wi-Fi module. Furthermore, a Raspberry Pi with integrated IEEE802.15.4 device has been exploited to act as a gateway for wireless sensor networks. ICT object associated in standardized networks (e.g. ZigBee, LWMesh) are automatically associated to the Gateway. Software components implement the VO Back-End Software REST Module for wireless sensor networks connecting automatically to the VO Registry. The RaspberryPi is a credit-card-sized single board computer that that can be used to connect sensors and actuators. It comes with Broadcom system on chip (SoC), which allows the use of various Linux distributions. ZIGPOS developed an IEEE802.15.4 (ZigBee) bridge that can be plugged on the Raspberry Pi.

Additional hardware includes an Android tablet that is used to showcase the mobile app for the Smart Home CVO. The technical specifications of this tablet are: Ram: 1 GB, Processor: 1.6 GHz dual core, Screen: 10.1", 1024 x 600 pixels, Storage: 8 GB, Android version: 4.1 "Jelly Bean". Finally, an Android phone is also used. The technical specifications of this phone are: RAM: 512 MB, Processor: 1 GHz single core, Screen: 3.7", 480 x 800 pixels, Storage: 4 GB, Android version: 4.2.2 "Jelly Bean".

## 2.2 Smart Office: Easy meeting

### 2.2.1. Overview

This proof of concept displays how the iCore functionalities can be used in a real life Smart Office Scenario.

Apart from the software components presented in other sections of this document, the Smart Office prototype combines dynamically adjusted sensing and actuating. Mobile electronic devices such as tablets and smart phones can be exploited offer ubiquitous computing.

The Smart Office environment aims to offer a seamless meeting experience to meeting participants and the meeting organizers with added capabilities of control. The iCore components highlighted in Figure 3 support a set of features of value during the whole meeting lifecycle, from its organization to its execution and aftermath.

These features have been described in detail in the iCore D6.3 deliverable and are, as such, mentioned very briefly here, in order to ease the understanding of the merit, the implemented iCore components brings to the realisation, of these features. These are supported as service requests that can be issued before the meeting (organization and setup of the guidance system) and during the meeting (meeting break, recording and wrap-up)

- Smart Meeting Organization; inviting meeting participants and sending out QR codes to be used for Guidance
- Smart Meeting Guidance; guidance of participants through QR-code scanning at Smart Panels to the meeting venue
- Smart Meeting Break; identifying the need for a break based on monitoring of the attention span or at will as the meeting organizer sees fit
- Smart Meeting Recording; record the meeting using Smart devices the participants bring with them, removing the need of dedicated recording equipment
- Smart Meeting Wrap-up; gathering the recording to a designated web area for further processing and offline availability to participants



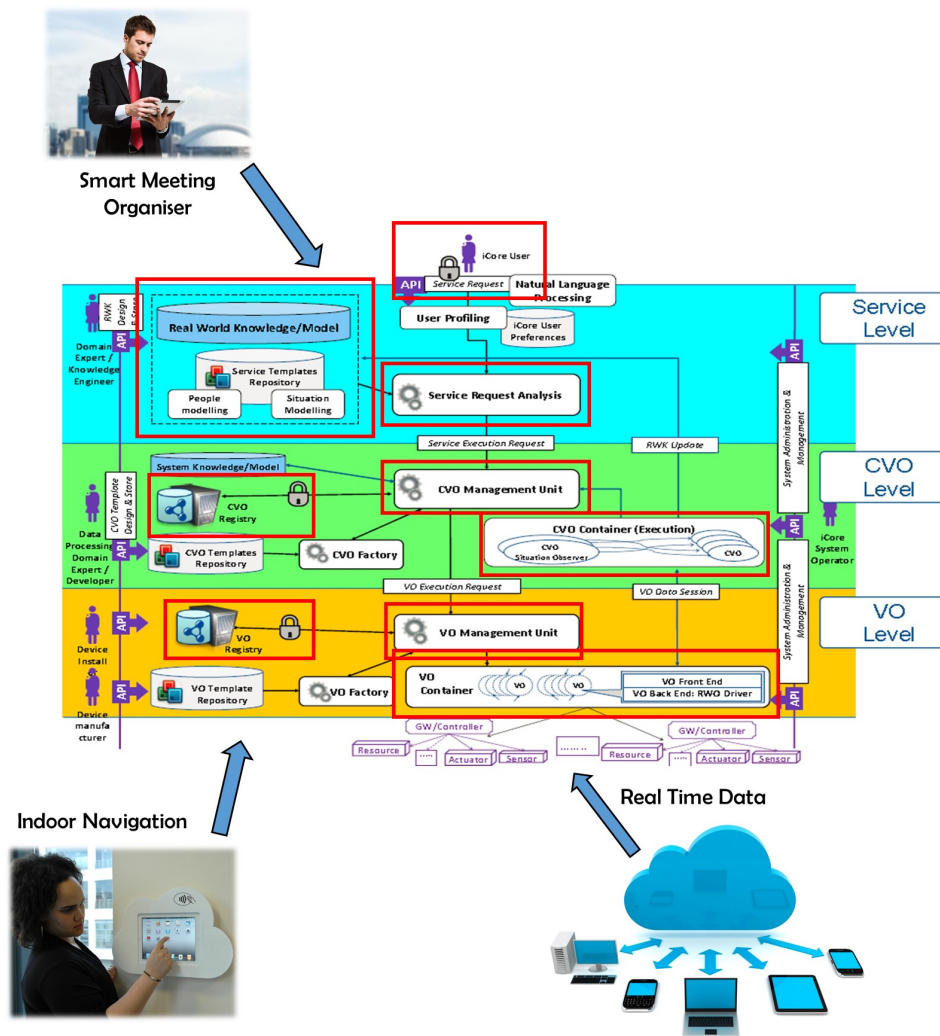


Figure 3: iCore architecture instantiation for the Smart Office: Easy Meeting use case.

In the rest of this section we will describe the involvement of the highlighted components and the value they bring in the Smart Office Realization.

## 2.2.2. Components

### 2.2.2.1. Service Layer

#### 2.2.2.1.1. Service Requestor GUI

This component is intended for the end user. It was designed and implemented keeping in mind all the latest UI/UX guidelines provided from leading software companies, such as Google. Its' friendly user interface is suitable even for people with limited smartphone experience. On-display buttons are responsible for receiving the service request, leading to the instantiation of the appropriate CVO. It has been implemented using technologies such as:

- a) Java
- b) MySQL

- c) Android
- d) XML
- e) MQTT

Summarising, this GUI is used by the meeting organizer to issue the aforementioned service requests. While not offering some special intelligence or functionality it allows the service requests from the meeting organizer to be issued in a very user-friendly way.

#### **2.2.2.1.2. Service Template Repository**

Service Templates Repository (STR) contains the collection of Service Templates. Its target is to provide to the SRA component the definition of the service requests in terms of needed features. This definition is included in the Service Templates.

Service request analyser has been implemented as a Java Standalone program, so it's able to run on standard hardware. The information arrives to the STR via MQTT protocol; in order to understand the MQTT messages, the application uses the Paho Java Client like the SRA does too. The main MQTT operations are subscribing to MQTT-topics and publishing messages events.

The STR receives the service request topic from the SRA, analyses the type of service of the payload of the MQTT message and consults the repository looking for the template that corresponds with the service requested.

The STR sends the XML template to SRA via MQTT protocol.

#### **2.2.2.1.3. Service Request Analysis**

Service Request Analysis (SRA) is the first component that receives the service request. Its target is to identify the type of request of the service and ask for detailed information to the Service Template Repository (STR). STR sends to the SRA the template that defines the service. SRA is also in charge to get the information required by the Real World Knowledge to process a service template.

Service request analyser has been implemented as a Java Standalone program, so it's able to run on standard hardware. The information arrives to the SRA via MQTT protocol; to understand MQTT messages, the application uses the Paho Java Client. The main MQTT operations are subscribing to MQTT-topics and publishing messages events.

The SRA receives XML template from Service template repository and it's parsed for analysing the template and building the XML Template Request. This request is sent to the CVO Management via MQTT protocol.

#### **2.2.2.1.4. Real World Knowledge Model**

This allows, as it will also be shown later, for filling in missing or incomplete information in the service templates before the passing of the Service Execution Request from the SRA component to the CVO Management Unit. As such, service requests that could have not been fulfilled due to lack of information can instead be fulfilled. As an example, for the Smart Meeting Recording service, suitable audio-capturing devices need to be discovered that are currently located in the meeting location. During the Service Request Analysis (SRA) for a Smart Recording service request, the Service Template (ST) for this service demands the meeting organizer to be given as parameter through the CVO level API to trigger the 'START recording'-command. Instead of the meeting organizer the meeting identifier was



given in the service request though. Although the SRA component cannot find the 'meetingID'-parameter in the ST, it is still able to ask for more information about the meeting of the particular identifier in the RWInfo base. The domain-specific knowledge that meetings have an organizer is modelled in the RWK knowledge base accessible by the SRA component.

In a similar way, geo-location is required to be passed through the CVO level API; however during the requests only the meeting location as a room/building number is given. By querying through the RWK base, the specific geo-location of that room can be retrieved and be passed to the CVO level.

In summary it can be said that the RWK model and RWInfo is used to assist the SRA component in filling the parameters for the Service Execution Request through the CVO level API in case the parameters cannot be determined from the service request and the ST alone. This capability eases the interaction with the iCore system since it contributes to fault tolerance by filling incomplete information. The RWInfo knowledge base is not only accessible to iCore developers, also CVOs are able to store and to update information there as their processing result. When executing the Smart Break Recommender service Situation Observers running to observe a particular meeting are able to update the status of that particular meeting in the RWInfo knowledge base. This updated RWInfo can then be used by other services according to their needs.

By providing a central place for RWInfo iCore services are able to share information without direct communication between those services. This system capability avoids dependencies between services that can lead to deadlocks in service execution once another service stops running.

This RWK and RWInfo bases are implemented using the open source tool MySQL. The connection between the iCore components and the bases is performed by exploiting the JDBC connector for java<sup>1</sup>.

#### **2.2.2.2. Composite Virtual Object Level**

##### **2.2.2.2.1. CVO Management Unit**

The CVO Management Unit has the role of determining the CVOs and VOs that will participate in the final execution of a service request.

CVO Management Unit has been implemented as a Java Standalone program, so it's able to run on standard hardware. The SRA information arrives to the CVO Management Unit via MQTT protocol; to understand MQTT messages, the application uses the Paho Java Client. The main MQTT operations are subscribing to MQTT-topics and publishing messages events. The CVO Management Unit requests a CVO available to CVO Registry for executing the specific service request. The request is sent via the Sesame RDF API to the CVO Registry using the HTTP/RESTful user interface supporting SPARQL protocol for RDF. The result of this SPARQL query is the identifier of the CVO available and it has the feature according to the service request.

---

<sup>1</sup><http://dev.mysql.com/downloads/connector/j/>

The CVO Management Unit requests the Virtual Objects involved in the service by consulting the VO Registry using the HTTP proprietary API provided by the developers of Virtual Object Registry.

#### **2.2.2.2.2. CVO Registry**

The CVO Management Unit needs to know the CVOs available for being able to fulfil a service request. In the iCore platform there are a lot of CVOs with different features executing services request. It implies that a status management is needed for finding a specific CVO for executing a service request. The CVO Registry is the component used by the CVO Management for getting the CVO needed for executing the service.

CVO Registry has been implemented as an HTTP server using the HTTP server provided by Sesame. This server implements an RDF repository and is accessed via the HTTP RESTful interface supporting SPARQL protocol for RDF.

The result of this operation is the identifier of the CVO available that has the feature requested by the service.

#### **2.2.2.2.3. CVO container**

All CVOs needed for realising the *Smart Break Recommender Situation Observer* have been implemented in Java Standard Edition to run on standard hardware, like a laptop. The use of Java-based CVOs allows for using a great variety of devices for hosting the Smart Meeting required processing functionalities. The information among the CVOs is exchanged via MQTT protocol. The Paho Java Client<sup>2</sup> library has been used to supply the CVOs with MQTT capabilities, like subscribing to MQTT-topics and publishing events. The Apache Commons Mathematics Library 'Commons Math'<sup>3</sup> has been used for statistical calculations implemented in the *Situation Detection CVO* as well as the *Situation Classification CVO*. Those two CVOs together with the *Situation Projection CVO* are provided with a Graphical User Interface visualising their processing results. The open source library 'JFreeChart'<sup>4</sup> has been used for implementing the graph charts for the GUIs of the CVOs depicted in Figure 4. The *Data Management Agent CVO* aggregates noise level values received as MQTT events from a Smart Phone VO over a time window of 30 seconds. This complex event processing capability is implemented within this single CVO by using the 'Esper for Java'<sup>5</sup>-engine that is integrated with the *Data Management Agent CVO* as Java library.

<sup>2</sup><http://www.eclipse.org/paho/clients/java/>

<sup>3</sup><http://commons.apache.org/proper/commons-math/>

<sup>4</sup><http://www.jfree.org/jfreechart/>

<sup>5</sup><http://esper.codehaus.org/about/esper/esper.html>

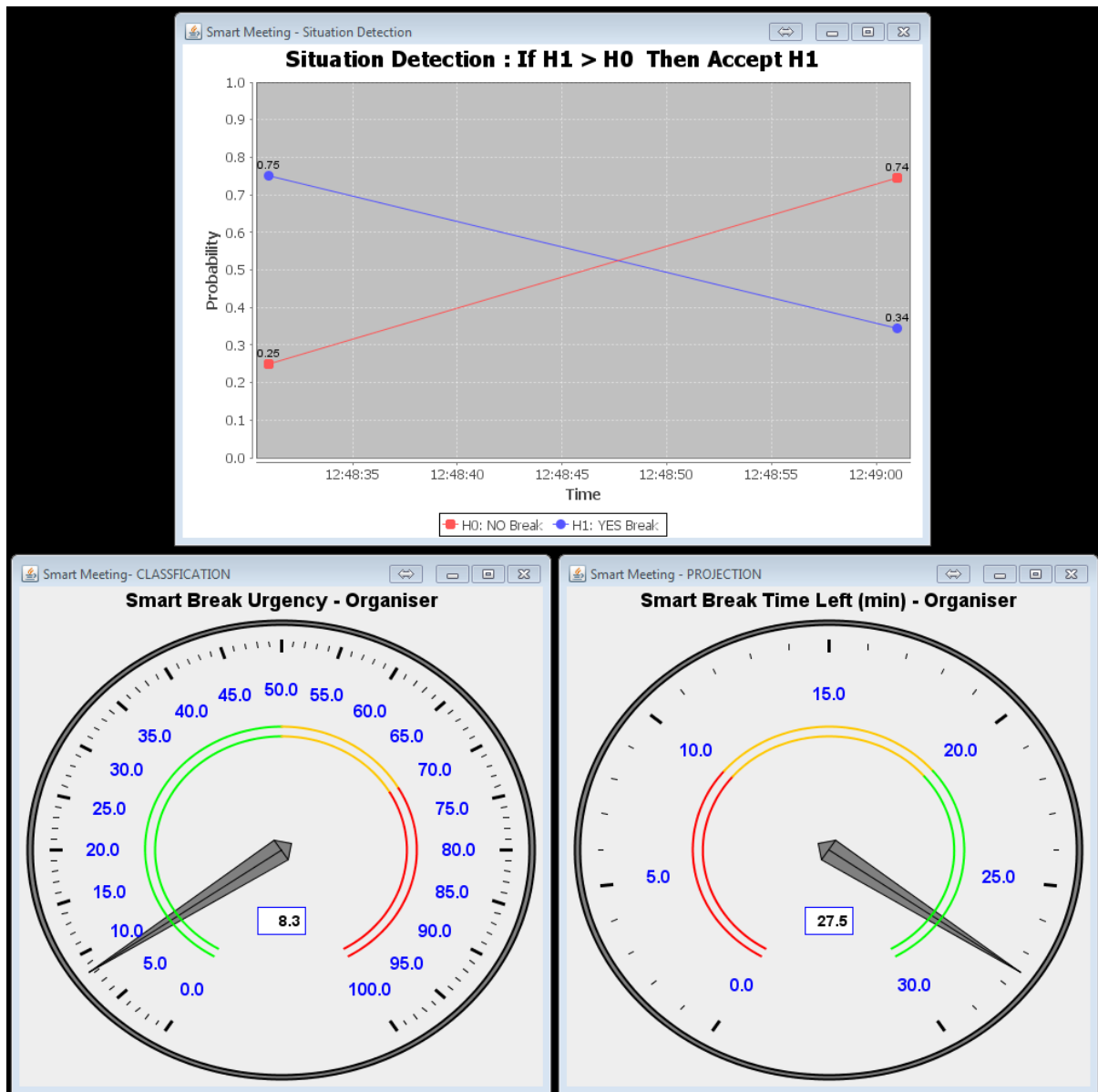


Figure 4: Graphical User Interfaces for Smart Break Recommender Situation Observer

Apart from the *Smart Break Recommender Situation Observer*, additional CVOs have been implemented to achieve our tangible targets.

#### 2.2.2.2.4. Smart Meeting Creation CVO

This CVO implements the smart meeting creating functionality. This includes the creating and storing meeting details, generating QR codes, and sending personalised e-mails to all the participants invited.

#### 2.2.2.2.5. Smart Meeting Guidance CVO

This CVO is responsible for the smart guidance scenario. The system should be able to guide a user to the meeting venue. This is feasible just by scanning the auto-generated QR codes mentioned in the section before.

#### **2.2.2.2.6. Smart Meeting Recording CVO**

Via this CVO the meeting organizer is able to record the meeting. The system can select the most appropriate device to handle the recording, according to the user preferences. The recorded fragment is stored locally to the mobile device for latter usage.

#### **2.2.2.2.7. Smart Meeting Manual Break CVO**

This CVO is the manual version of the Situation Observer presenter in a previous section. Through a user initiated poll, the meeting attendants express their opinion in having a break at any given time.

#### **2.2.2.2.8. Smart Meeting Wrap-Up CVO**

The last CVO presented in this proof of concept is the wrap-up. The main feature that serves is the ability to gather all the meeting-generated data and upload them in a remote location.

### **2.2.2.3. Virtual Object Level**

#### **2.2.2.3.1. VO Management Unit**

This component also plays a key role in the realization of the use case. It allows updating the VO registry so that it depicts at all times the correct properties of the VOs. In the Smart Meeting scenario where the service requests pose location restrictions and the involved VOs are largely mobile (i.e. smart phones that the meeting participants carry with them), maintaining the geolocation property of such VOs up-to-date is paramount for the meaningful and successful fulfilment of a service request. As an example, while many participant devices may have the hardware capabilities of recording the meeting, only the ones actually present in the meeting should be selected for doing so. Therefore, for the proper selection of such VOs to fulfil a smart recording service request, updating and maintaining their current geolocation is essential.

The VO Management Unit was implemented in Java and the connection between the VO management unit and the VO Registry is established using the library built by the UPRC and presented in the Smart Home proof of concept.

#### **2.2.2.3.2. VO registry**

The VO registry, developed for this scenario, captures all the information of an ICT object which is represented as a VO in our system. Details about the geolocation, features, functionalities, and ownership can be stored, updated, and retrieved in a quick and robust way.

Keeping a VO registry updated enables the system to select the most efficient device to handle requests, as well enforces the reusability which is a key factor to the success of iCore. Furthermore, the geolocation information enhances the security and privacy presented by this research study. A VO can be discovered and exploited only when the ICT object it represents is located in a specific place and has granted the proper access rights. The VO registry is updated in a seamless manner following the guidelines set by the project architecture.

More implementation details of the VO registry can be found in the iCore D3.4 deliverable.

### 2.2.2.3.3. VO container

In our case, different VOs were designed and implemented in order to achieve the tangible targets of the Easy Meeting scenario. It is out of the scope of this document to analyse further the implementation techniques and the low-detailed differences among them. However, we can say more about the about their use and main advantages they offer.

The VO container used in the Smart Office scenario offers certain advantages. First of all, being implemented using open-source platforms makes it suitable for a wide range of devices of all budgets. Furthermore, taking under consideration the spectrum of potential host-devices, it offers responsive UI that enables the interaction with the end user. All the available functionalities are presented in a very clear yet discreet way without compromising the user experience. It also:

- ✓ Serves as a platform to inform the VO owner by using the notifications of the ICT device.
- ✓ Allows the system to exploit the hardware resources in a scalable and efficient manner
- ✓ Since the communication relies upon lightweight protocols in terms of energy demands it makes it suitable for the battery constrained nature of the mobile devices.

For all the communications, the open source MQTT library “Paho” is used, which is described before in this document.

### 2.2.3. Hardware

In this section we will present all the hardware used for the Smart Meeting proof of concept. Mainly we are facilitating portable electronic devices that run the open source Android operating system.

Tablets and smartphones of different computational power are exploited in order to achieve the heterogeneity faced in real life scenarios. From stock versions of the OS, to customised ROMS and rooted kernels, the Smart Meeting scenario is possible to be executed in almost all the Android devices.

The android tablets used in this scenario are:

1. Samsung Galaxy Tab 3: Dual-core 1.5 GHz, 1 GB ram, 16 GB flash memory. Running 4.2.2 rooted stock rom.
2. Samsung Galaxy Tab: Dual-core 1 GHz, 1 GB, 32 GB flash memory. Running 4.0.4 stock rom.

Android smartphones exploited are:

1. Google Nexus 5: Quad-Core 2.3 GHz, 2 GB RAM, 32 GB flash memory. Running 4.4.4 rooted stock rom.
2. Sony Xperia S: Dual-Core 1.5 GHz, 1 GB RAM, 32 GB flash memory. Running 4.4.2 rooted CyanogenMod rom.
3. Sony Xperia S: Dual-Core 1.5 GHz, 1 GB RAM, 32 GB flash memory. Running 4.1.2 stock rom.

4. HTC One S: Dual-Core 1.5 GHz, 2 GB RAM, 32 GB flash memory. Running 4.1.1 stock rom.

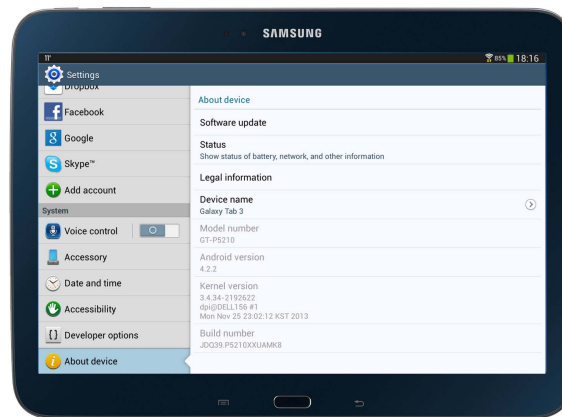


Figure 5: Android Tablet



Figure 6 : Sony Xperia S

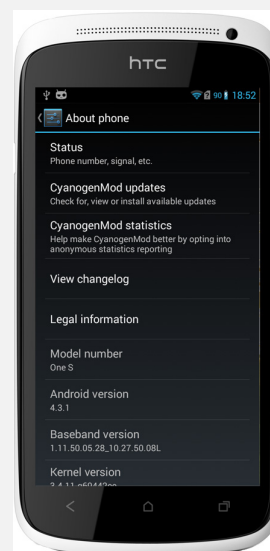


Figure 7 : HTC One S

Our selected devices feature a number of sensors, many of which (in bold below) are used during our use proof of concept scenario; these are used to notify participants of actions they need to perform as part of service requests (e.g. state whether they need a break), to allow participant input whenever needed, to fulfil service requests (e.g. record meeting) while also driving the decision making engines of CVOs (e.g. accelerometer is being used to indicate whether a participant is “stable” enough to carry out a recording) as well as simply notify participants of what tasks their device is currently undertaking (e.g. recording, or uploading recordings during the wrap-up)

- Gyro Sensor
- Proximity Sensor

- **Accelerometer Sensor**
- **Compass**
- **Notification Light**
- **Notification Sound**
- **Vibration**
- **Microphone**
- **Display capabilities**
- **Touch input capabilities**

Furthermore, a virtual machine was deployed in order to keep all the vital components, of the iCore platform, up and running. The details of this VM are presented below:

- Dual Core 2 GHz
- 6 GB ram
- 100 GB hard disk
- 

## 2.3 Smart City: Transportation

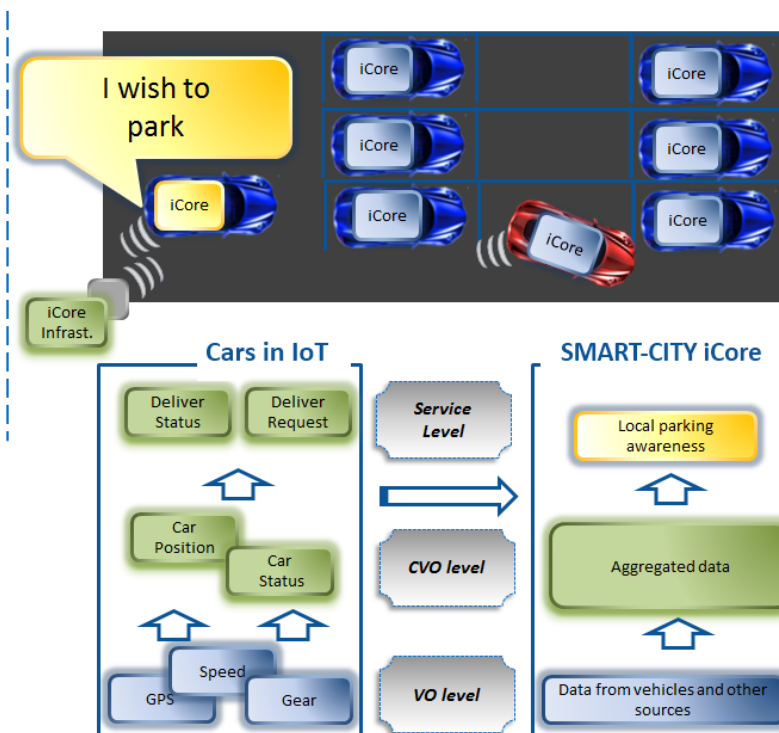
### 2.3.1. Overview

#### Local iCore on Single Vehicles

- VO: Speed, GPS, etc.
- CVO: Car\_Position, Car\_Status, etc.

#### SMART-CITY iCore

- VO: Vehicle data, other data
- CVO: Aggregated data
- Service: Local parking awareness



**Figure 8: iCore Smart City Transportation Demonstrator**

The Smart-City Transportation use case demonstrates the virtualization and use of ICT objects in automotive, to create, configure and use mobility functions and services while driving and, in a seamless way, also in pre-trip and post-trip services, linking to smart home and smart meeting. Although the focus is on a single driver, data provision from several cars can also be addressed, for the mobility management in a smart city. Major aspects and challenges are the availability of objects within the vehicle and from the outside world,



considering the vehicle as a complex and autonomous eco-system and not an always connected environment. Another topic addressed is context awareness using cognitive technologies.

The Smart-City Transportation proof of concept demonstrates the “Smart Parking System” scenario, where two main real-life situations are addressed:

1. The User is driving downtown and needs to park his/her car; thus, he/she requests to the parking city infrastructure the location of the parking area – with parking lots available – which is nearest to the current vehicle position.
2. The User travels along a recurrent route (e.g.: from home to office, every working day early in the morning; from office to home, every working day late in the evening; from office to the swimming pool every Wednesday late in the evening; from home to downtown, every Saturday morning; etc.). A “User Behaviour Adaptation and Prediction” component is able to detect from the geographic position data sent by the car these recurrent behaviours and to infer if the User is currently driving along one of such recurrent routes, so to detect his/her recurrent destination then, it can provide this knowledge to the “Smart Parking Service”, which will exploit it in order to provide parking information based not (only) on the current driver’s position but (also) on the “predicted” destination and “predicted” route. This will have impact on the quality and on the “usefulness” of the information provided to the user.

### 2.3.2. Components

For the Smart City Transportation Demonstrator, the following iCore components have been instantiated

#### VO Container, Parking System VO, Car Display VO, Car GPS VO

The VO Container has been implemented as a Web Service container, providing the VO specific interfaces to be compliant with the M3S CVO Execution Engine. On the VO Container, the following VOs have been deployed:

- Two different Parking System VOs, Turin Parking System VO with real data coming from the City of Turin Open Data Service and the Milan Parking System VO (Simulated)
- The Car OnBoard Unit Display VO, for interfacing the OnBoard Unit Display. The Car GPS VO, for interfacing the Car GPS Location data.

The interface exposed by this VO has been maintained consistent between the two different implementation steps of the Demonstrator, so to show a fundamental concept of the iCore Architecture – the **Object Virtualization**, which allows to define a single interface for different real objects.

#### VO Registry integration

The Smart City Transportation Demonstrator integrates the VO Registry, where the Parking System VOs have been registered, in order to make location-based queries to identify the most correct VO to be bound to the CVO execution, so to show another fundamental concept of the iCore Architecture: the Service Adaptation based on the context (**Context Awareness**).



## CVO Management Unit

In the Smart City Transportation Demonstrator the CVO Management Unit is the endpoint for receiving the Service Execution from the On Board Unit and for providing the User the information on available Services, according to the situation identified by the RWK module. This in turns is translated into a CVO Execution and demonstrates another concept: **CVO dynamic execution** based on User/World situation identification.

## CVO Container

The CVO Container exploited for the Smart City Transportation Demonstrator is the M3S Orchestration Engine, where the CVO is defined as an event-based workflow connecting the needed VOs and providing the behaviour logic. The following CVO have been defined:

- **Parking Around Me CVO:** this CVO will exploit the Car VO and the Parking VO in order to provide the information on available parking places to the Driver, based on the current car position. The CVO will bind the specific Parking VO according to a location-based query to the VO Registry.
- **Parking at Destination CVO:** this CVO will exploit the Car VO and the Parking VO to provide the information on parking based on i) the current location of the car and ii) the identified recurrent destination of the driver.

## Real World information

During the CVO execution, after the information has been sent to the Driver, the car is continuously monitored towards the destination, in order to collect the user behaviour which is useful information for offline data aggregation and traffic modelling. This demonstrates yet another iCore concept: the **Real World Knowledge** growing based on data coming from the Real Objects.

## User Behaviour learning and prediction

This component has been introduced into the Smart City Transportation Demonstrator thanks to the integration of a Recurrent Trip Learning module provided by CRF. The Recurrent trip learning module is able to

- learn the recurrent destinations of a User based on data coming from the GPS VO and
- predict the User destination based on a limited set of car location information during the trip.

Thanks to the exploitation of this component, the Demonstrator shows the possibility of adapting the Service Execution based on the availability of Knowledge (in this case, on User Behaviour).

### 2.3.3. Hardware

In the evolution of the iCore Project, two different Smart City Transportation prototypes have been developed, based on two different hardware OnBoard Units. Figure 9 depicts the prototype which has been demonstrated at the end of Year 2 of the Project, based on the Fiat Blue&Me™ Unit.



**Figure 9: End of Year 2 Smart City Transportation Prototype Implementation**

As shown in the picture, also a prototypal vehicle – a FIAT Grande Punto – equipped with a Blue&Me™ device was prepared to support test activities in real life scenarios (road tests).

Figure 10 shows instead the final Smart City Transportation Proof-of-Concept Prototype environment, which will be based on different On-Board Unit: the Uconnect™ infotainment platform.



**Figure 10: The Uconnect™ Onboard Unit front-end on FIAT 500L**

### ***The Uconnect™ OnBoard Unit***

Uconnect™ Radio system is the new Infotainment platform that FIAT-Chrysler offer on almost all new vehicle models (mainly of FIAT, Alfa Romeo and Chrysler brands) as well as on MYs (Model Year) launched starting from 2013. The first FIAT vehicle equipped with Uconnect™ was the FIAT 500L (launched in 2012); currently, also Alfa Romeo Giulietta MY2014 and MiTo MY2014 offer this device.

The specific version adopted for the EMEA market uses a 5" colour touchscreen, the largest in its class, to interface multimedia devices – telephone, smartphone, radio, media player and mp3 player.

The device implements Bluetooth technology to support hands free and SMS reader functionalities.

For the development of the iCore Proof of Concept prototype, a special version of Uconnect™ is being used: such a “debug” version is derived from the Uconnect™ NAV version, in order to have the availability of the GPS peripheral (normally used to support Compass and Map-based navigations, although these functionalities are not used in the iCore prototype); furthermore, it adds to the other standard features the availability of an Ethernet interface, which is used both for debug purposes and to provide Uconnect™ with Internet access.

Although there could be several means to provide Uconnect™ with 3G connectivity, it was decided to use an external 3G modem having also an integrated Ethernet switch: this allows to share the same 3G access with both Uconnect™ and iCore-enabled laptop PC (implementing CVO and VO container) thus making this prototypal implementation easier and faster.

In order to enable road tests – thus in real driving situations – also for the second implementation, a FIAT 500L was modified by replacing the production Uconnect™ installed there, with a “debug” version and further equipped with the other peripherals needed (mainly the 3G modem / Ethernet hub).

## 2.4 Smart City: Urban Security

### 2.4.1. Overview

The prototype demonstrated is very close to the operational surveillance system that could be used in the near future; indeed, through a realistic VIP evacuation scenario, a representative set of software and hardware components is demonstrated:

- Control and Command (aka C2) end users applications/consoles providing dedicated Common Operational Picture (COP) to the police forces with multiple points of view (video, CBRNE),
- a software framework with built-in cognitive based functions, i.e. an iCore service platform, managing automated operational situation awareness enhancement and automated network management for superior wireless sensors network performance and availability,
- Software providing a simulation of thousands of people in a big area with realistic behaviour (decision and navigation).

To illustrate this iCore platform application for a concrete case, for the experimental setup as shown in Figure 11 we can formulate the **value proposition** to the (platform-owning and operating) police force as:

- 1) Depending on the preferred operational way of working and modus of a police force or a concerned C2 operational team, different types of Service Requests (SRs) can be provided **flexibly** on the C2 consoles.
- 2) Depending on a maximally acceptable cognitive load, C2 operational staff is shown only a limited number of **selected** live sensor observations and video streams that critically matter to the situation-aware decisions they have to take.
- 3) Given the maximal capacity of the WSN, network links and routing paths are reserved for the highest priority streams, **dynamically according to RWK-based**

**relevance predictions**, so that the network provides the streams requested in the C2 in the best way possible.

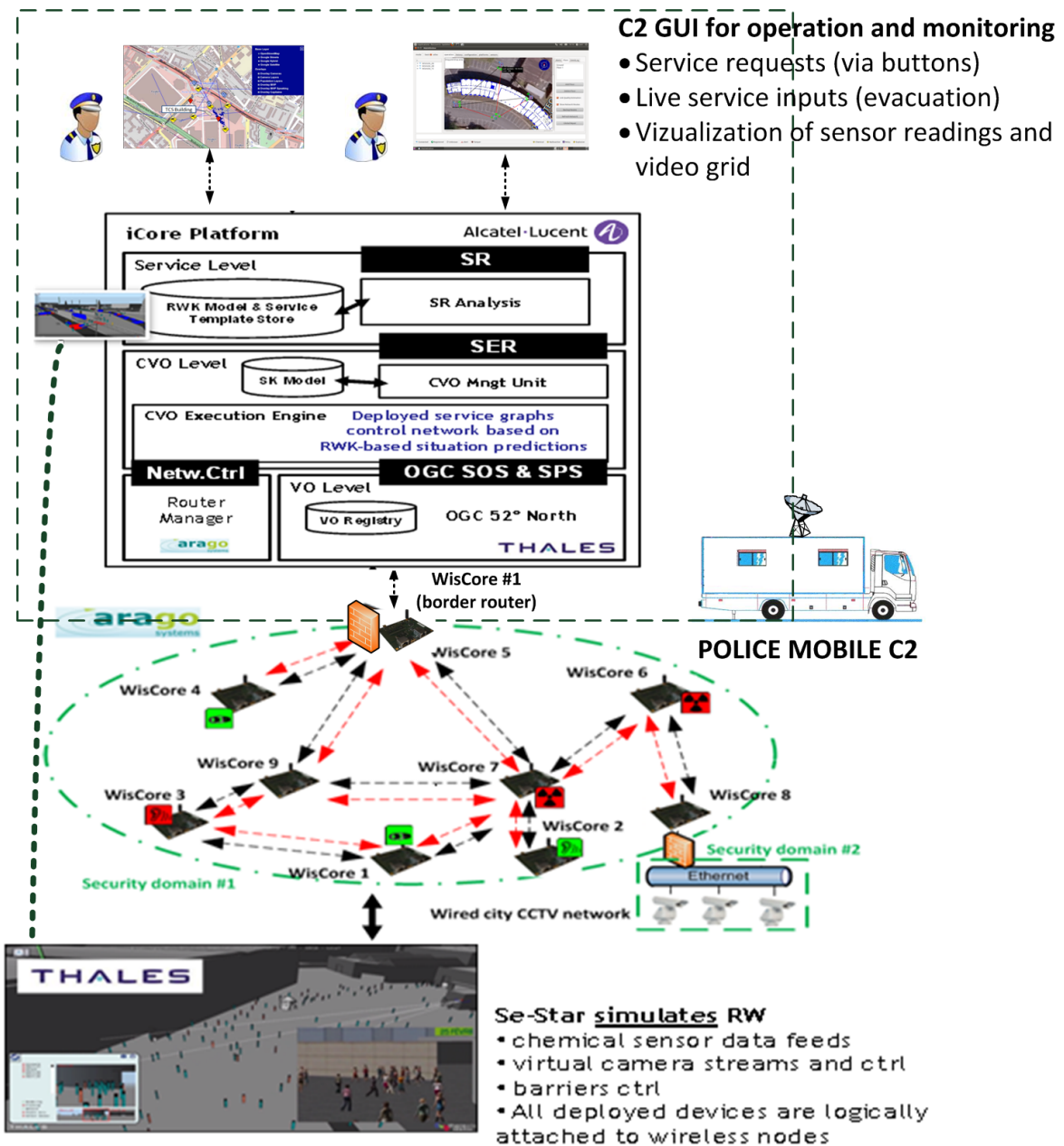


Figure 11: Overview of Urban Security setup

From the design analysis of this case for the iCore platform, we chose 5 representative SRs, with corresponding C2 GUI interaction, and identified the relevant RWK to be foreseen (in brackets we mention the ST name and the ST parameters as to be provided in the SR):

- **SR1("area of interest video overview", geo-coordinates, Prio 7)**
  - **Service purpose:** Shows a 'less redundant' selection of relevant camera views on a given geo-area (issued typically during whole operation)
  - **RWK assumed:** polygons of "areas of interest" for which at least one camera view is needed (indicates potential redundancies in camera views)

- **C2 GUI assumed:** ‘start’ button with priority indication (default = 7), geo-area selection (graphical, rectangle or polygon) and graphical geo-highlight of active cams & view areas, placeholder for video stream matrix (MJPEG or H.264 over HTTP), ‘close’ button
- **SR2(“VIP monitor & control”, VIP ID, initial set of waypoints, Prio 1)**
  - **Service purpose:** Shows camera views (PTZ controlled) where VIP occurs and is expected to occur 10s ahead + dictate live waypoints to VIP (issued typically at the start of the VIP visit, until the end of it)
  - **Additional RWK assumed:** in addition to the RWK modelling needed for SR1: *prediction model for ‘next 10s’ position of person based on current speed vector that can be weighted with next waypoint*
  - **C2 GUI assumed:** ‘start’ button with priority indication (default = 1), (VIP ID is preconfigured in a configuration file, as this is pre-mission data), graphical GPS location VIP, way to indicate course waypoints (“area of interest” granularity), placeholder for video stream matrix (MJPEG or H.264 over HTTP), ‘close’ button
- **SR3(“crowd monitor & control”, Prio 2)**
  - **Service purpose:** Shows camera views where crowds occurs and are expected to occur 10s ahead, and open/close gates by click on map (as a typical additional request; in case there is a potential crowd or crowd mobility)
  - **Additional RWK assumed:** in addition to the RWK modelling needed for SR1: a *crowd detection and prediction algorithm*
  - **C2 GUI assumed:** ‘start’ button with priority indication (default = 2), indicate view areas in focus on map, placeholder for video stream matrix (MJPEG or H.264 over HTTP), show gates status on map, ‘close’ button
- **SR4(“CBRNE cloud monitor, people effect video”, Prio 3)**
  - **Service purpose:** Shows camera views where people are hit by (and fall down due to) CBRNE cloud (issued typically when an explosion incident occurs)
  - **Additional RWK assumed:** *cloud dispersion model* (that can predict cloud dispersion based on live CBRNE sensor data; simple ‘growing circle’ model is assumed)
  - **C2 GUI assumed:** ‘start’ button with priority indication (default = 2), indicate view areas in focus on map, placeholder for video stream matrix (MJPEG or H.264 over HTTP), ‘close’ button
- **SR5(“CBRNE cloud monitor, CBRNE measurements”, Prio 3)**
  - **Service purpose:** Shows sensor data of CBRNE sensors on map (permanently monitored during whole mission, for immediate situation-aware action in case of an incident)
  - **Additional RWK assumed:** same as in SR4
  - **C2 GUI assumed:** ‘start’ button with priority indication (default = 2), indicate sensor readings on map, ‘close’ button

For each of the Service Requests, a **Service Template** is foreseen which captures the required video stream and sensor data processing as needed for visualisation in the C2 user interface. As an important aspect of the service logic, all of the defined services **observe the situation of a particular RW phenomenon**:

- the general (visual) status of the **area** under surveillance (SR1),



- the location/movement (and visual status) of a **person** (the VIP) via direct GPS readings (SR2),
- the existence and location/movement (and visual status) of **crowds** (SR3),
- the existence and location/movement (and visual status) of **chemical cloud** (SR4&5),

## 2.4.2. Components

We provide here the main functional components of the PoC according to the iCore architecture, so the components supporting VO level, CVO level and Services level. We also describe Thales Se-Star software application a core component of the surveillance system PoC that simulates smart city with a high degree of realism and supports here the use case systematic validation process.

### 2.4.2.1. VO level

At the VO level, one of the main objectives is to support relevant IoT standards related to interoperability between services or applications and devices (sensors, actuators). So we choose to use IETF CoAP and HTTP protocols (REST compliant) to transport devices data and commands over networks: these are the protocols at the southbound interface. Because the wireless network we are demonstrated includes low data rate and high data rate interface for flexibility and energy efficiency reasons, CoAP is obviously used over low rate interface and HTTP goes over high rate interface; CoAP can also go over high rate interface when network usage optimization makes it relevant. For the northbound interface, several OGC (Open Geospatial Consortium) services are combined and provide a sophisticated abstraction of any devices with Plug-n-play based on discovery mechanisms, geo-queries and filtering, XML based data models; HTTP is used as the transport protocol.

Figure 12 highlights the overall set of technologies and interfaces implementing this VO level.

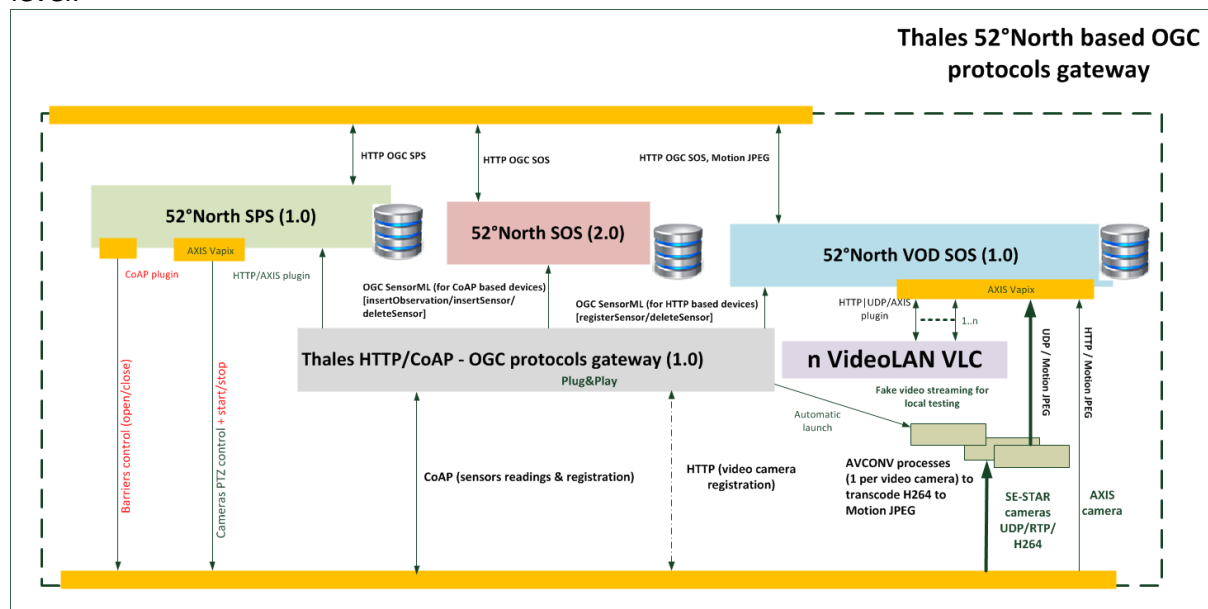
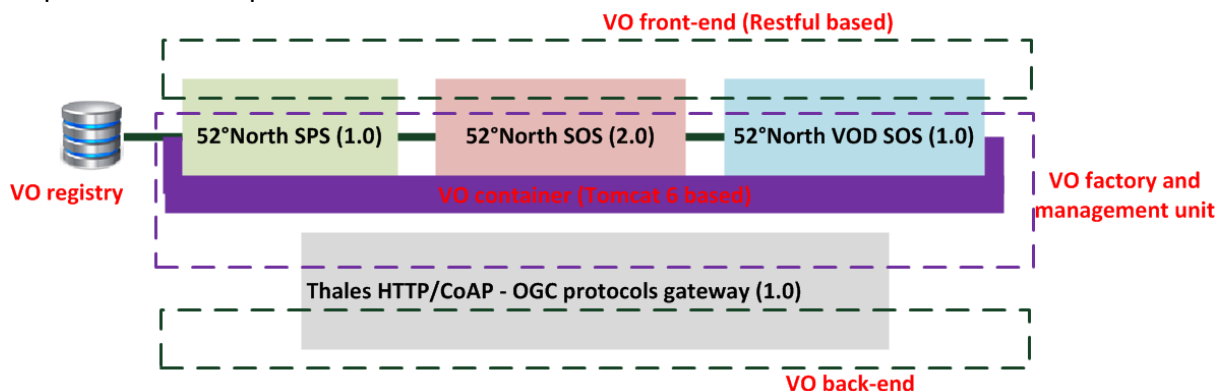


Figure 12: Urban security OGC / CoAP based VO level software components

There are four main components here. Three first are OGC services, namely the Sensor Observation Service (SOS), the Sensor Planning Service (SPS) and the video on demand

Sensor Observation Service based on free software 52North (<http://52north.org/>) implementations following the latest OGC specifications (<http://www.opengeospatial.org/standards/sos>, <http://www.opengeospatial.org/standards/sps>); a fourth component has been developed by Thales to extend OGC services with CoAP / HTTP based sensors Plug-n-play: it means that deployed devices are automatically discovered and registered in OGC services as long as they are connected and running through the network. This component has been developed using Java and CoAP Californium (<https://www.eclipse.org/californium/>). Sensor Observation Services provide access to sensors data (video streaming, chemical detectors, etc.) while Sensor Planning Service provide control over the sensors and actuators. All three OGC services API use HTTP and XML so the interactions with the CVO level are based on this combination. All three implementations by 52North are java based and running over Tomcat 6.

Figure 13 makes the link between the VO level functions of the iCore architecture and the implemented components.



**Figure 13: iCore architecture VO level functions and urban security implementation**

We describe shortly hereafter the VO front-end and back-end.

Regarding naming and addressing technical choices, we have two stacked levels of abstraction because of the high degree of heterogeneity of the sensors network introduced above. Rationales are provided in deliverable D2.5 highlighting iCore architecture.

At network level (VO back-end), “low level” addressing is based on IETF IPv4 and IPv6 standards; “high level” naming for the connected devices is supported by IETF URI general rules specifically refined both by IETF RFC 7252 and VAPIX AXIS interface.

We use for video cameras PTZ (Pan, Tilt and Zoom) control the VAPIX AXIS interface compliant URL scheme (based on HTTP protocol, documentation is provided in [http://www.axis.com/techsup/cam\\_servers/dev/cam\\_http\\_api\\_index.php](http://www.axis.com/techsup/cam_servers/dev/cam_http_api_index.php)); typically “http://IPv4@:port/axis-cgi/com/ptz.cgi?PTZcommand”. For video streaming we use IETF RTSP/RTP/RTCP protocols suite.

For chemical sensors, VIP GPS, barriers, and so on with small piece of data passing over the very constraint IEEE 802.15.4 radio we use CoAP protocol (IETF RFC 7252, <http://tools.ietf.org/html/rfc7252>) then the URI scheme “coap://[IPv6]/path with resource name” where path identifying resource name is kept simple as much as possible, typically “chemical sensor 1”; so a resource here represents an access point to a physical or virtual device (detector or actuator) connected physically to a wired or wireless



communication network node transmitting readings, control, etc. over the network; in a more general way CoAP we have defined a resource as an access point to several (one or more) virtual or physical devices embedded with an integrated platform. Similarly to HTTP, REST type operations (GET, PUT, etc.) can be applied to embedded CoAP server.

Towards CVO level (VO front-end), we use OGC URN (Universal Resource Names) formalized in <http://www.opengeospatial.org/ogcna> with also the document "Definition identifier URNs in OGC namespace" found in <http://www.opengeospatial.org/standards/bp>. Together with SOS interface, connected devices access is manageable through a unique standardized interface to the CVO level. Here is an example for a video camera:

```
<ows:Parameter name="offering">
  <ows:AllowedValues>
    <ows:Value>ifgicam.video</ows:Value>
    <ows:Value>ifgicam.settings</ows:Value>
  </ows:AllowedValues>
</ows:Parameter>
<ows:Parameter name="eventTime">
  <ows:AllowedValues>
    <ows:Range>
      <ows:MinimumValue/>
      <ows:MaximumValue/>
    </ows:Range>
  </ows:AllowedValues>
</ows:Parameter>
<ows:Parameter name="procedure">
  <ows:AllowedValues>
    <ows:Value>urn:ogc:object:sensor:IfGI:ifgicam1</ows:Value>
  </ows:AllowedValues>
</ows:Parameter>
```

**Figure 14: Connected devices compliant with OGC URN**

For example, a request from CVO level to get chemical sensors detections is like:

```
http://localhost:8080/52n-sos-webapp/sos/kvp?service=SOS&version=2.0.0
&request=GetObservation&procedure=urn:ogc:procedure:Thales:Chemical1
```

This request is related to CoAP notification from CoAP resources such as *coap://[aaaa::200:0:0:2]:5683/R2="l=120" h=1406213839* over the sensors network.

A request from CVO level to steer the video camera to the required position will use *http://localhost:8080/SPS/SPS* that will be further translated into *http://172.16.105.11/axis-cgi/com/ptz.cgi?zoom=5400* controlling the given video camera.

#### 2.4.2.2. CVO level

On top of the VO Level, the CVO Level performs the service execution. As indicated before, services consist of monitoring (and control) services for the various RW artefacts important to the C2 surveillance team at a given moment during the operation, and RWK is used for predictions related to the prediction trajectory of the VIP, and crowd and chemical cloud dispersions. From the Service Level, Service Execution Requests are thus issued for each SR, containing each a logical CVO component graph describing the service goal and RWK-based prediction logic.

Based on these SERs, the CVO Management Unit determines the CVO instances that need to be deployed, thereby also adding system-related logic for optimizing execution (in this case: logic for reserving network resources according to live service priorities, ‘deploying’ network flows rather than new CVO instances dynamically).

As a result, based on the services and system considered in the Urban Security case, notably following important system (sub) components and CVO types are considered:

- **Trajectory Forecast CVO** (used for **VIP** track prediction, taking pre-planned course VIP-waypoints and VO-sourced GPS data stream as live input),
- **Circular dispersion prediction CVO** (used for course **chemical cloud** dispersion prediction, taking into account a collection of threshold sensor readings streams),
- **Clustered velocity prediction CVO** (used to predict most probably average movement direction of **crowds**, based on detected person speed vectors in each selected virtual camera video stream),
- **Camera Control CVO** (used to control **PTZ** movement and **video stream quality** of individual Virtual Camera VOs), using also a set of **OGC interfacing CVOs**, and a **Stream Manager** infrastructure function), and
- **Flow (Priority) Controller** (system function part of CVO MU, using a dedicated interface to an OpenFlow-like Router Manager of the WSN to reserve QoS-level assign network Flows ranked in a priority list, taking in priority-labelled flow reservation request by service instances),
- **Stream Priority Prediction CVO** (deriving the likely relevance of virtual cameras, and implied Flow, at an SK-determined future point in time, from RWK-based situation predictions, in the context of the (priority of) each service instance),

An important deployment-time functionality of the CVO Management Unit is the **reuse of CVO (and implied VO) instances** across the service instances, at each moment a new SER is handled:

- The CVO instances capturing RWK-based predictions are **observing and forecasting the same common Real World status / situation** and so are reusable across all service instances.  
(Note that the CVO MU is not aware about this RW aspect, but rather recognizes the reuse opportunity due to the fact that it already deployed identical sub-graphs (i.e. CVO types and parameters connected to the same VO source or sink.)  
(This is the case for e.g. the first 3 CVO Types listed above.)
- The CVO instances (or system functions) relating to the status of the underlying infrastructure (execution environment and in this case network resources), as also this **system infrastructure is shared** (even in an optimized way) among the service instances.  
(This is the case for e.g. the Flow Priority Controller function and the Camera Control CVO.)

Note that for both categories a CVO (or a system function) needs to **aggregate and arbitrate among control inputs** from multiple service instances in case it concerns an **actuation** of the Real World (VO sink) or the System. (This is again the case for the Flow Priority Controller function and the Camera Control CVO.)

As a result of the mix of service instances requested to be executed, corresponding to the SRs by C2 human operators (surveillance police team), the complex executing CVO graph is selecting video streams from the cameras as relevant to the observed situations, and, ahead of time, is making flow priority reservations in the network, as such obtaining smart, more optimal assignment of network resources to the most important video (and sensor data) streams. Figure 15 gives a high-level overview of the mechanism, where by the RWK-based situation predictions, the executing CVO graph derives Flow specifications and priorities according to anticipated relevance to the by themselves also prioritized service instances, so that at the time the stream is actually activated it can use the reserved Flow.

**SK** maintained at the CVO Level for the platform + infrastructure setup at hand, is mainly about the **network (and stream source) properties**:

- maximum (worst case) propagation time required for setting up a new Flow across the routers in the WSN
- capacity dimensioning indicators (maximum bandwidth of line-powered gateway node versus battery-powered node, and other dimensioning data)
- Video quality levels and corresponding resolutions, bandwidth requirement, etc.

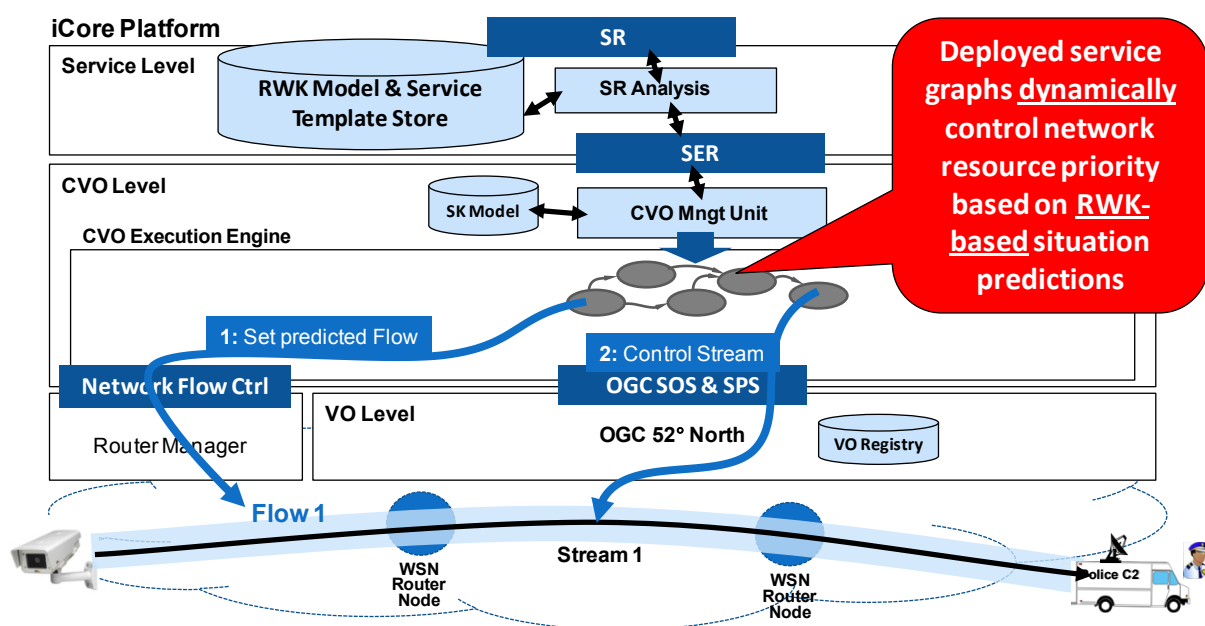


Figure 15: Network flow priority control leveraging RWK-based situation prediction in Service instances

#### 2.4.2.3. Service level

As they illustrate well the actual demo scenario, the **SRs** issued to the iCore platform (from the C2 application front end), the list of SRs and the **RWK** referenced from the corresponding Service Templates has been discussed at the beginning of section 2.4 (2.4.1 Overview).

As is generically the case in the iCore platform, SRs are matched with service Templates and the referenced RWK is used to parameterize the CVO Types in logical CVO graphs sent to the CVO Level as a SER.

Interesting to note for the Urban Security case, is that the RWK is derived from mainly two sources:

- operational domain expertise (common practices, do's and don'ts) for police surveillance operations and operation planning fixed before platform operation: static RW facts such as VIP planned waypoints and a dedicated Evacuation Plan, and
- the Se-Star simulated World (camera positioning and properties, people behaviour)

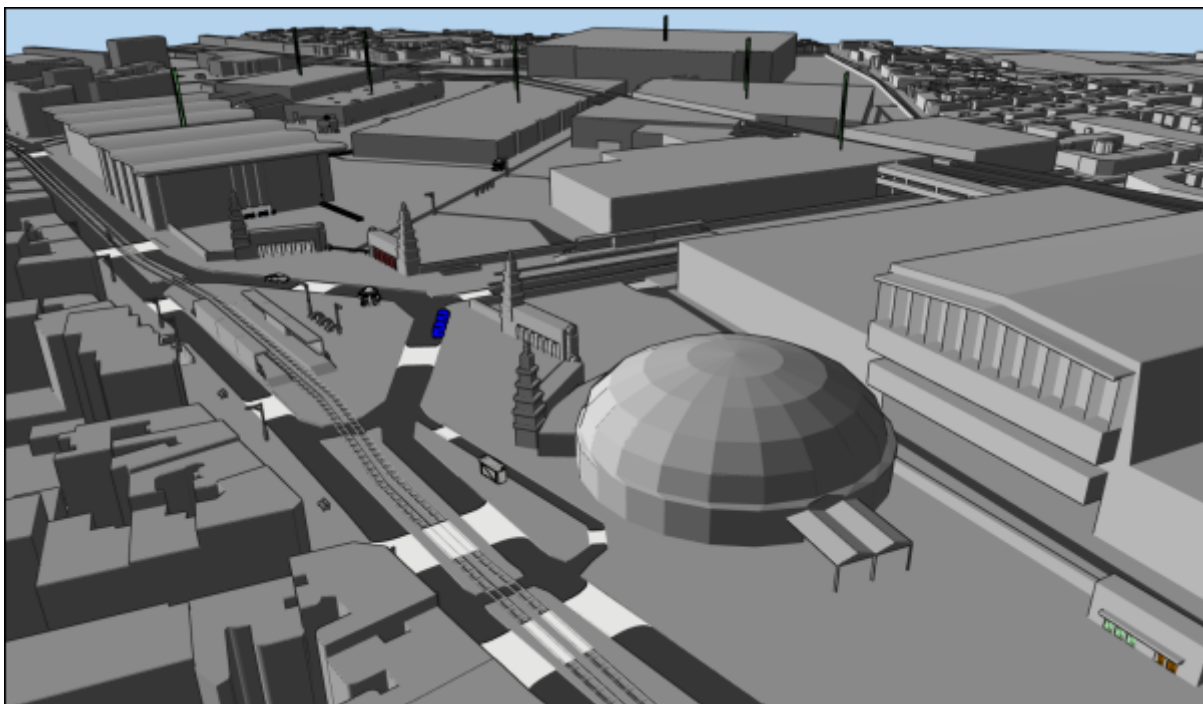
#### 2.4.2.4. C2 application front-end

Both for testing and demonstration purposes, a flexible HTML-based C2 user interface was built, based on simplified C2 operational team requirements (simply button control for SRs, suitable data visualisations and controls). In a post-project exploitation step, it could be substituted by a commercial Thales C2 console.

#### 2.4.2.5. Bio-inspired synthetic environment for system behaviour validation

We use a real-time simulator to run the scenario in a big area with many chemical and video sensors. The simulation takes place in the "Parc des Expositions". It is a big public exhibition site in Paris. The SE-Star simulator is able to animate a complex environment populated with thousands of virtual actors.

With this scenario, all these actors will visit the area and after the bomb alert, they will evacuate the area.



**Figure 16: Close view of the Parc des Exposition entrance, without any virtual actors**

With the simulator, we can change many settings easily, such the number and the position of the cameras, detectors, etc.

The software runs the scenario while being connected to the different components of the demonstrator. Thus, the simulator can send data to them and receive data as well:

- Sent data

- Inject sensor values for the chemical sensors into WSN nodes
- Inject virtual camera-rendered video streams into the WSN network
- Send GPS position of the VIP to the iCore platform (via the WSN)
- Send crowd detection helper data to iCore platform (replacing video recognition which would be used on real camera content)
- Received data
  - Receive PTZ command for the virtual cameras from the iCore platform via the network
  - Receive direction orders for the VIP evacuation from the C2 user (or autonomously) via the iCore platform

The simulator uses the H.264 norm for the encoding and the RTSP protocol for the video streaming.

### 2.4.3. Hardware

Except laptops and desktops running iCore platform, C2 applications and Se-Star simulation software, the main hardware PoC of interest is the ARAGO WiScore low power multi-radios platform dedicated to IoT applications with both indoor and outdoor deployment.

The WiScore platform comprises two key hardware units that allow running simultaneously a high and a low data rate network and integrates two commercial solutions namely WiSGate and WiSMote.

**WiSGate** embeds **Linux** free and open source operating system, fully customizable. It relies on a 32-bit RISC ARM based processor (ARM926EJ-S) and supports the high data rate network based on the 802.11s standard protocol; Figure 17 shows the key hardware components of the WiSGate. CCTV digital camera sensors were integrated on WiScore through Ethernet.

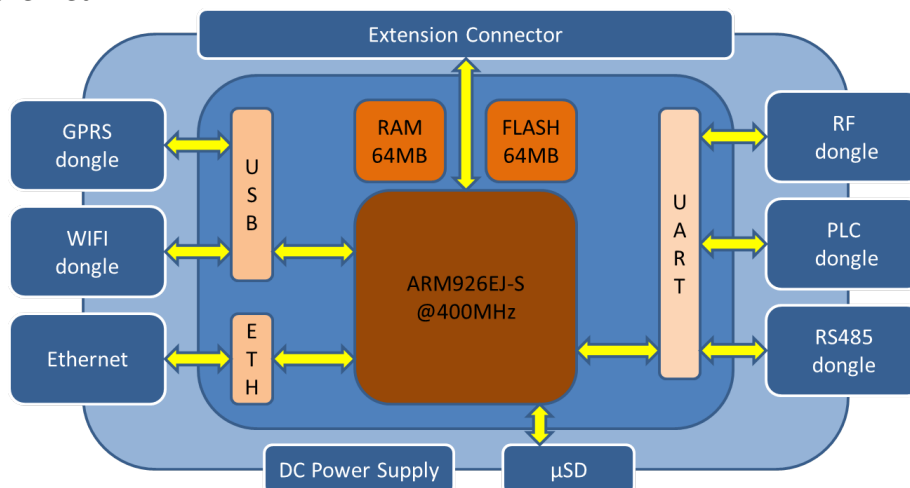
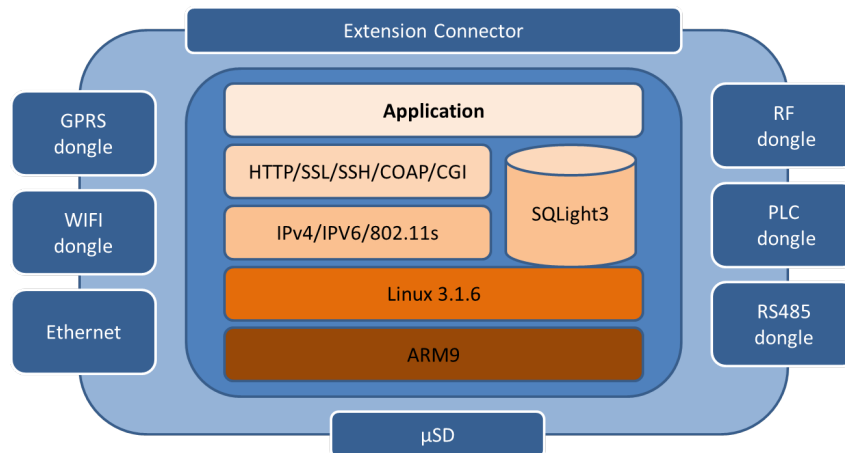


Figure 17: WiSGate hardware main components

Concerning the software, WiSGate comes with 6 key software functionalities such as: a light CGI web server, a SQLite3 data base, SSL/SSH, PHP, virtual-network kernel devices, IEEE 802.11s mesh driver, and IPv6; see Figure 18. As the platform is fully open, additional packages can easily be added. Thus, 802.11s protocol (54 Mbits/sec.) has been integrated to

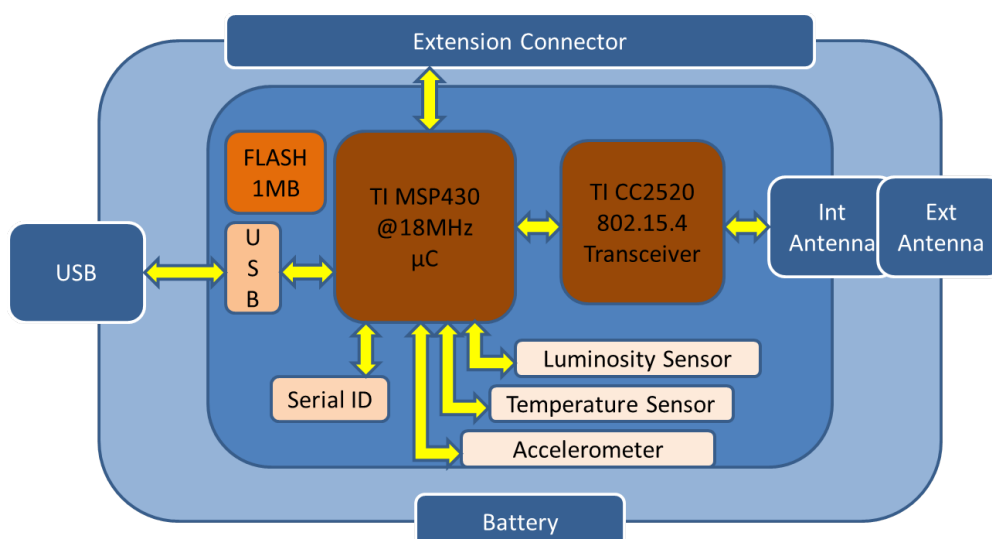
ensure on demand streaming over multi-hop network. Since the on demand video is issued mainly from a central point to multipoint, we decided to use the HWMP routing protocol that supports either Multi-Point-to-Point or Point-to-Point communication.



**Figure 18: WiSGate runs the customer applications under Linux OS**

Through HTTP, SSH, and CGI web server the gateway can be setup or configured. Although GPRS could be used for data dissemination, Wi-Fi (802.11s) USB dongle was benchmarked to guarantee the on demand video streaming.

**WiSMote** is the second key component of the WiScore with a TI MSP430 5 series (16 bits) microcontroller, it is a sensor module designed to meet the requirements of Wireless Sensor Network (WSN) applications, namely, physical measurements in fields like environment, healthcare, smart building, logistics or industrial. Main hardware components are depicted in Figure 19. The mote operates over 2.4GHz ISM (250 Kbits/sec.). It is equipped with a duty cycle radio and embeds some sensors, for instance: luminosity, temperature, accelerometer, or humidity.



**Figure 19: WiSMote RF - hardware architecture**



Figure 20 summarizes the WisCore functional architecture focussing on main network interfaces. Concretely, Ethernet (Eth0) incoming/outgoing packets are redirected by default to 802.11s mesh (mesh0) over the br0 interface (mainly for video streaming from SE-Star application). SE-Star application uses also for chemical sensors data the IEEE 802.15.4 network over a SLIP tunnel between a physical serial port and a virtual network interface (tun0). To debug and configure the board the UART 1 serial based interface is used.

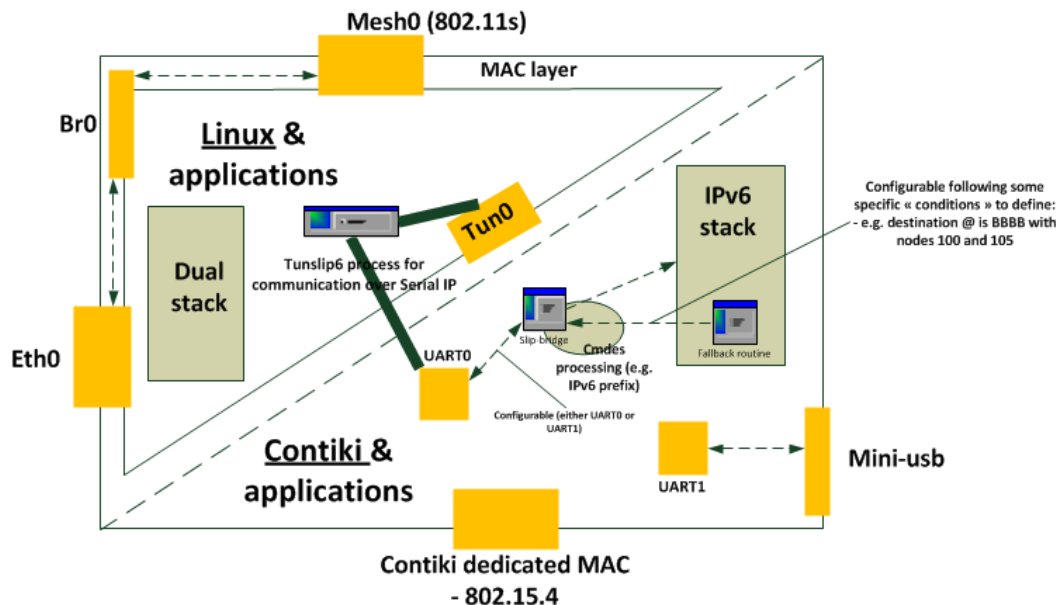


Figure 20: Urban Security use case – ARAGO wireless platform at functional level

Figure 21 illustrates the current release 1.0 of the hardware platform which is the integration of several commercial products depicted above.



Figure 21: Urban Security use case – ARAGO WisCore wireless platform picture

The second release of the platform provides higher integration and a more expendable solution with batteries included.

ARAGO also designed a Quality of Service (QoS) interface, router flow control, to handle multiple services on the platform, to improve situation awareness in critical decision



making, to proactively or dynamically select video streams taking into account the real-time network features.

The ARAGO Router Flow interface relies on HTTP CGI (*Common Gateway Interface*) executed by a web server. It automatically returns an HTML page. The advantage of using CGI is the usage of the dynamic pages in line with the user preference. It means that the data defined by the user is carried as parameters of the CGI script.

The main features of the CGI are:

- read data flow entries,
- process strings,
- write to the standard output flow,
- to be executed by the server.

Two main features are targeted by the remote control interface: **remote control** and **flow priority** configuration. The remote control interface gives the possibility to monitor and control fully the mesh network. Among the main features we mention: mesh topology initialization, node reset, retrieving node neighbours, obtaining the routing table, requesting particular routing entry, removing routing table entries, modifying the 802.11s channel or the transmission power.

Regarding the flow priority, it allows to configure a given priority on the nodes, to update a flow, and to remove a flow. More specifically, the remote control interface was dedicated to the VIP scenario that demands real time network monitoring and intelligent on-demand video streaming.

To conclude, ARAGO succeeded to bring a concrete solution for the IoT Urban Security Business based on CCTV systems, wearable sensors, and Chemical, Biological, Radiological, Nuclear, and Explosives (*CBRNE*) *sensing systems*. Overall, WiSCore provides a valuable answer for two open and hot topics in IoT, namely, on board multi-radio system coupled with security and control enhancements.

## 2.5 Smart Business: Supply Chain Management and Logistics

### 2.5.1. Overview

The Smart Business use case demonstrates the following:

- Acceptance reports: the iCore system provides the end user of perishable goods (e.g. a retailer or pharmacist) a report at delivery of the products showing its conditions during transport and storage. Based on these reports, the end user is able to decide whether to accept the shipment or not. At the same time, the reports can be used by transporters to avert claims for spoiled goods. The prototype is focused on temperature sensitive products.
- Real-time alerts/early warnings: the iCore system provides (early) warnings to transporters when the products are outside their optimum storage conditions. The system provides these warning to guarantee the quality of products throughout the supply chain and enables transporters to act before damage occurs.
- Automatic control of storage and transport conditions: Since the iCore system has knowledge of what conditions are required for products within a warehouse/truck. Therefore, it can provide functionality to automatically control environmental

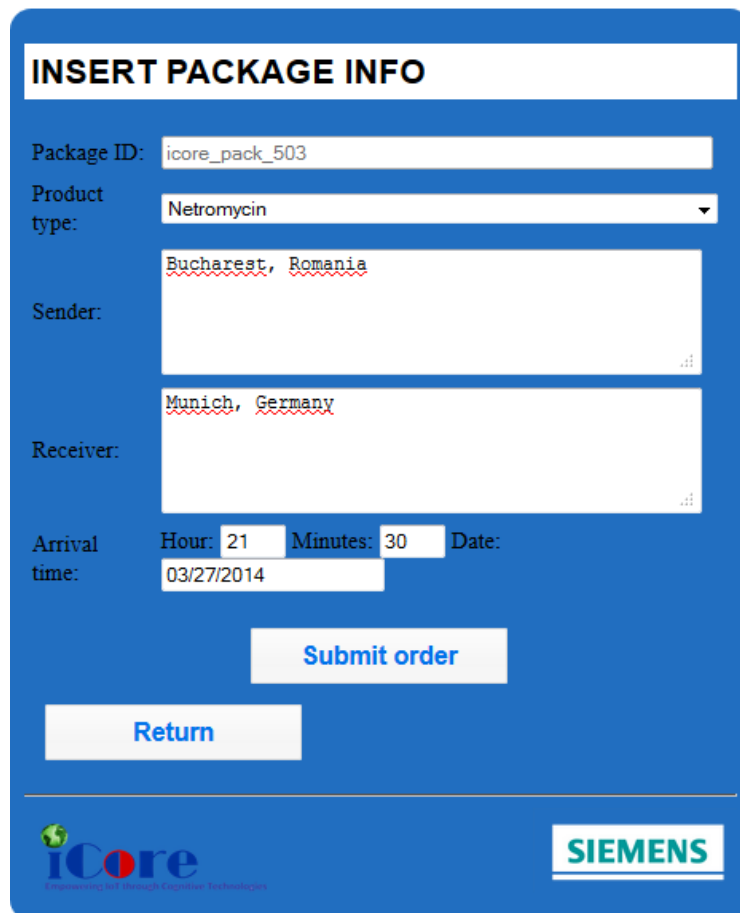
conditions based on the requirements of products stored in (compartments of) warehouses or trucks by directly controlling coolers/HVAC systems.

See [http://www.youtube.com/watch?v=a2wkpz7FkYk&feature=player\\_embedded](http://www.youtube.com/watch?v=a2wkpz7FkYk&feature=player_embedded) for an impression of the intended story. In the following sections, the prototype is further detailed.

## 2.5.2. Components

### 2.5.2.1. Service Request GUI

This component is used by the transport company clients for generating the transportation requests (see Figure 22). The client has to specify the product type, the addresses of the sender/receiver and the arrival time. The acceptance criteria for the products are included by the domain expert in the RWK database.



**Figure 22: The GUI used by the transport company clients for registering a transportation request.**

The Service request GUI component is used by the transport company employee to manage the sensors and to register the binding when the sensors are attached to the parcels (see Figure 23).

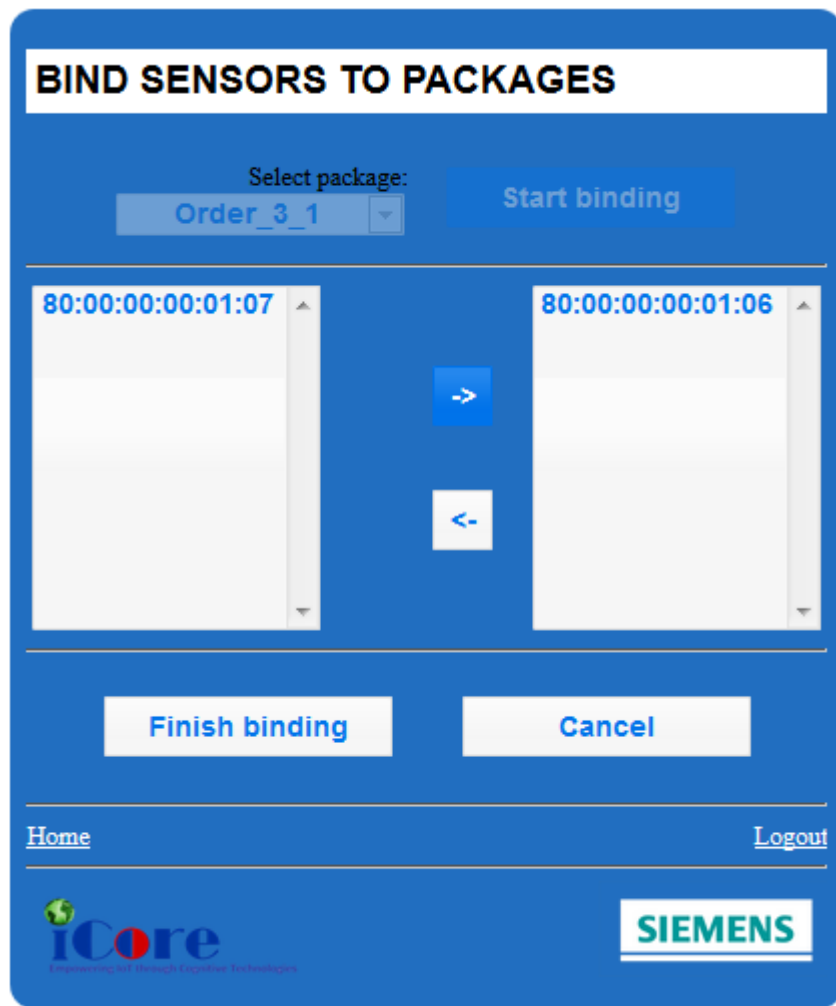


Figure 23: The GUI used by the transport company employees for registering the binding of a sensor to a package.

#### 2.5.2.2. Real world knowledge model

The domain expert includes in the RWK the acceptance criteria for different types of products and the prediction models, which are used for forecasting the state of the products during the transportation process.

More details about this component can be found in D6.5 Final proof of concept prototype for Smart Business.

#### 2.5.2.3. Real world information

The transportation process requires continuous accumulation of historic data for generating the reports for end users (transport company employees and clients) and for detecting when the prediction models are outdated.

More details about this component can be found in D6.5 Final proof of concept prototype for Smart Business.

#### 2.5.2.4. Service request analyzer

This component is used for generating the appropriate service execution request, depending on the job type specified by the transport company client. SRA identifies from the real world knowledge model the transportation criteria for the product and selects a list of prediction models which can be used for estimating the transportation condition of a package. The list of prediction models contains various prediction models for the available transportation means (by truck, by minivan, by plane, etc.).

#### 2.5.2.5. Situation modelling

As mentioned above, historic data (raw events, complex events) is collected in the RWI database. Once a transportation job is finished, the situation modelling analyses the data and calculates the prediction error of the model. If the prediction error is larger than a threshold for several transportations jobs, then the prediction model is considered outdated. In this case the situation modelling retrains the prediction model using the accumulated historical data. The new model is stored in the RWK database as the replacement for the existing one.

The prediction problem falls into the time series prediction area. There is a multivariate series, with data internal and external temperature, humidity, current clamp, and parcel temperature sensors. These values were the ones received by associated VOs and stored as real world information. The Weka forecasting plugin was used for building a predictor. The inner regression model needed as “base forecaster” is chosen among linear regression, multilayer perceptron, M5P and RBF – see Figure 24. Supplementary, any of these models’ hyper parameters can be further customized – see Figure 25.

Comparison between the predicted and actual values can be seen in Figure 96. The behaviour of the model can thus be investigated by a Machine Learning expert.

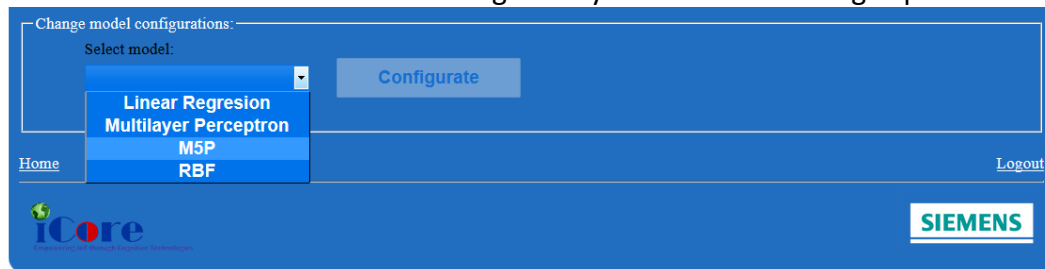


Figure 24: Choosing the base forecaster for the time series forecasting model

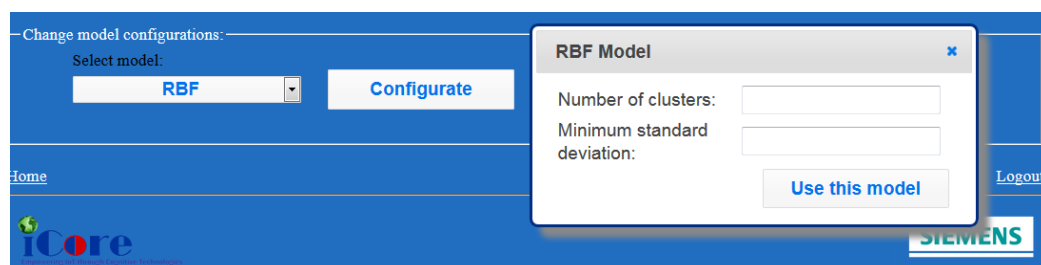


Figure 25: Choosing models' hyperparameters

#### 2.5.2.6. CVO Management Unit

The CVO Management unit receives an XML document from the Service Level’s Service Request Analyzer. Among service details, some candidate predictive models stored by the

RWK are handled to CVO management unit. The effective predictive model is selected based on the CVOs selected for execution.

After analysing the request details, the CVO Management unit interrogates the CVO registry to find already running CVOs that might be reused for the task at hand. If necessary, it invokes CVO factory's services to appropriately instantiate a CVO, based on some predefined templates.

Based on the CVOs which collaborate for solving the current task, the CVO management unit selects the appropriate predictive model from the list of candidate models passed through the Service Request Analyzer. The selection of the predictive model cannot be done before this stage, due to lack of complete knowledge of which CVOs are employed for the current task. The predictive model itself will be embedded in a dedicated CVO.

#### **2.5.2.7. CVO Container (Execution)**

The CVO Container is continuously analysing and aggregating the VO event data. The continuously processing is described in queries using the VO events and the conditions passed by CVO management Unit.

More details about this component can be found in D6.5 Final proof of concept prototype for Smart Business.

#### **2.5.2.8. CVO registry**

At a service request, the CVO Management Unit has to find out whether some of the needed CVOs are already running. This question is answered by the CVO registry, which maintains ids and semantic descriptions of the running CVOs.

More details about this component can be found in D6.5 Final proof of concept prototype for Smart Business.

#### **2.5.2.9. CVO Factory**

The CVO factory is able to instantiate a CVO based on a CVO template. It answers to CVO Management Unit requests, which handles a description of the CVO functionalities. The CVO factory creates a SPARQL query for the CVO template repository; based on the retrieved templates, it makes a decision and instantiates a CVO which is further handled to the CVO Management Unit.

#### **2.5.2.10. CVO template repository**

The templates which are used to instantiate CVOs are maintained by the CVO Template Repository. In our case, we use Sesame triple store to maintain the semantic description of the templates. This repository is queryable through an API and provides a template, which is further used by the CVO factory to instantiate a CVO.

#### **2.5.2.11. CVO template**

The CVO templates are used to instantiate CVO. The instantiate CVO are started and executed in the CVO Container (Execution).

Additional to the component described in D6.5 "Final proof of concept prototype for Smart Business" the integration of the CVO prediction resulted in new findings.

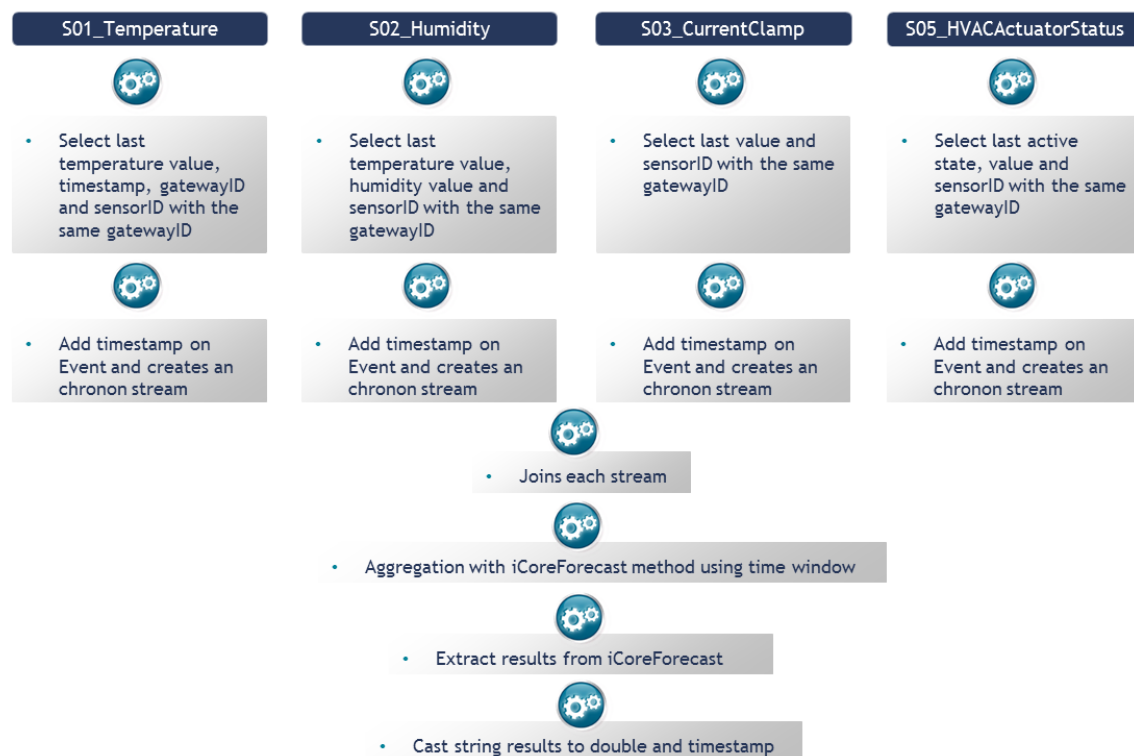
A CVO template can use additional CVO templates inside to hide the complexity to higher level.

The CVO template for templates for predicting temperature in a container uses several VO data.

The instantiating of the template needs the container temperature VOs, the surveyed PackageTemperature VO, the Humidity VO, the CurrentClamp VO and the HVACActuatorStatus and a time window for gathering the VO values within the range.

This CVO template uses a set of CVO templates for finally predicting the package temperature value 10 min, 1 hour and 2 hours in the future.

## Temperature Prediction Template within a container



**Figure 26: Set of CVO templates for predicting temperature**

In the figure above the CVO Template block “Select last Temperature value ..” and “Add timestamp on Event” reduces the amount of events to the values the iCoreForecast module is interested. The next CVO Template joins all events within a container.

After the CVO Pattern “Aggregation with iCoreForecast module using the time window” the results are prepared for further processing.

### 2.5.2.12. CVO prediction

The RWK contains a set of predictive models (neural network models, support vector machine models, decision trees models, see Figure 24 and Figure 25) which are in charge for triggering early warnings, based on previous experience – on which the predictive model was developed – and on the current input. The predictive model stored by the RWK is handled to CVO management unit in form of a serialized JVM object.

The predictive model is deserialized at the CVO level and wrapped by a CVO which further connects to appropriate datasets. The embedding CVO may include the workflow traditionally found in a ML process: selection of the data, pre-processing (other than CEP, it might include here some customized data conversion, e.g. discretization or casting from categorical to numerical data). From all the CEP primitive operators, time alignment is one of the most useful tools.

The values provided by VOs associated internal and external temperature, humidity, current clamp, and parcel temperature sensors are used as prime data for the forecasting model; then the model is asked to provide the values in the forecasting horizon. More specifically, the predictions are made for 10 minutes, one and two hours onwards. The predictions are consumed by the automatic control of transport conditions and for early warning functionality of the use case.

### 2.5.2.13. Dispatcher Dashboard

In the logistic chain several stakeholders are involved. One stakeholder is the dispatcher responsible for the transport. The dispatcher gets informed, when a transport may run in bad condition or is in bad condition. Figure 27 shows, beneath the historical value and the current value, the truck position.



Figure 27: Dispatcher Dashboard showing truck position with temperature values

### 2.5.2.14. CVO controller for AC units

In this CVO is implemented the logic for controlling the state of the AC unit based on the temperature inside the container.



#### 2.5.2.15. VO Registry

The VO registry contains a contextual enriched description of ICT and non-ICT objects. The VO registry is a key component in iCore to lookup devices (e.g. search on location, available resources on objects and description of how to interact with an object), but also to manage devices and selection of best fitting data sources for a particular request.

The VO registry in the Smart Business use case contains entries for each device, however, also abstracts to “shipments” and contains information about device ownership.

#### 2.5.2.16. VO Factory / VO Template Repository / VO Management Unit

The VO factory, VO template repository and VO management unit interact with ICT-objects and keep the VO registry up-to-date. The initial VO registry entry is created according an RDF template description of (known) devices.

#### 2.5.2.17. VO Container / VO Back-end / VO Front-end

The main functionality of these components is to provide a runtime environment for VOs. E.g. data produced by devices is published in XML format via MQTT and the data is augmented with device ownership information etc.

#### 2.5.2.18. Security at VO Level

The security at VO level is based on policies<sup>6</sup> e.g. an owner of a device may always interact with a device and also stakeholders in the logistics process (e.g. transportation subcontractors) who are currently responsible for the cargo, however, others not. The security features are interwoven in many VO level components, but the main policy enforcement point is the MQTT broker.

#### 2.5.2.19. VO monitoring interfaces

VO monitoring functions have been implemented for debugging purposes i.e. any information published by the VO level to other iCore levels is stored in a database and a web application has been developed to interpret the published XML snippets and connectivity statistics are collected.

### 2.5.3. Hardware

This proof of concept is mainly focused on the handling of temperature sensitive products in the supply chain. The hardware used are mainly wireless sensor network devices from Ambient Systems.

<sup>6</sup> More policies are defined using SecKit implementation of JRC.



**Figure 28: Wireless temperature sensor deployed close to temperature sensitive product**

The wireless sensors are designed for mobile use and to travel with the produce from manufacturing site to the end-user. The sensors communicate wirelessly in the 2.4GHz-band and have enough memory to store months of sensor readings in the case they are outside the coverage of WSN gateways (e.g. in an ship/airplane). The sensors can be configured remotely and in general push temperature information once per 5 minutes (which is common practice in cold chain monitoring). Optionally, the sensors can provide remaining shelf-lifetime of the products they monitor, based on historical temperature conditions.

The following additional sensors are integrated in the Smart Business Proof-of-Concept:

- Open/close sensors are used to detect if doors in warehouses or trailers are opened or closed;
- Relative air humidity sensors are used in the cold chain, not to monitor the quality of the effective substances in pharmaceutical products, but to determine condition of (carton) packages (condense). If the packaging is not in pristine condition, it may not be accepted by end-users.

All the wireless sensors above have been developed by Ambient Systems.

Wireless sensor network infrastructure (such as routers and gateways) is “see-through” concerning sensor data. The gateway, developed by Ambient Systems, communicates periodical via socket connections with the VO Back-end using 3G connection. The gateways used in this proof of concept also provide location information (GPS) e.g. to determine the location of the trailer it is mounted on. Also a gateway based on an embedded PC has been used in the proof of concept.



**Figure 29: A wireless sensor network gateway device**

## 2.6 Smart Tour in the City trial

### 2.6.1. Overview

This trial aims to showcase aspects of cognitive management for IoT self-management in the scope of a Smart tour in the city application, particularly addressing scalability issues (in terms of VOs, real world information data, numbers of users). In terms of iCore features and components the trial highlights the following:

- **Dynamic (on demand) service provisioning:** Travel agencies request an application from IoT application providers.
- **Automated analysis of user requirements:** A tourist registers himself (prior to the utilization of the service) and gives information about his/her profile. Based on this input, user requirements are derived.
- **Service personalization:** The service takes into account user preferences.
- **Cognitive capabilities:** Through exploitation of real world knowledge, cognitive management mechanisms allow the dynamic reaction (or proactive action) corresponding to changes in the application context.
- **Minimum human intervention:** No intervention from the tourist is needed for the creation of a service that respects his preferences.
- **Abstraction of heterogeneity:** Exploitation of virtual objects that correspond to heterogeneous sources of information.

The Smart tour in the city trial offers information on traffic, points of interest and weather to tourists, dynamically suggesting optimal routes to reach desired destinations, based on user preferences and situation (e.g. location). It should be noted that the value and novelty of the Smart tour in the city trial does not come so much from the services offered to users (tourists) but rather to stakeholders such as IoT application providers and travel agencies that can exploit the features of the iCore architecture for facilitating the development of services/applications that can be provided to tourists. Table 1 summarizes the advantages of a Smart tour in the city application developed by exploiting the iCore framework as opposed to other similar solutions.

Part of the trial has focused on Athens addressing tourists visiting different sites around the city. Another major part of the trial concerns the exploitation of the SmartSantander infrastructure for conducting experiments for the large scale evaluation and validation of the integrated iCore architecture and concepts.

The stakeholders addressed in this use trial are IoT application developers, Travel agencies and Tourists. Further stakeholders, which were not addressed but could be targeted include Municipalities (e.g. the City of Athens or other cities), Tourism related open communities (e.g. <http://opentourism.gr>).

**Table 1: Advantages of a Smart tour in the city application developed by exploiting the iCore framework as opposed to other similar solutions**

With iCore	Without iCore
Service requests in <b>natural language</b>	<b>Predefined interaction</b> with the system through customized Human – Machine Interfaces (HMIs) that correspond to system functional requirements
<b>Dynamic (on demand) creation of personalized services</b>	Design and development of services even based on user preferences, but in a <b>custom</b> and <b>manual way</b> of deployment
<b>Self-management capabilities, Self-awareness of CVOs</b>	<b>No cognitive capabilities</b>
<b>Autonomous</b> discovery, control and interoperability of virtual objects corresponding to heterogeneous sources of information	<b>Customized</b> deployment of devices and exploitation of their information

### Trial storyline

A travel agency (namely CLIPPER travel and SALINA travel) requests an IoT application from WINGS ICT Solutions (“I want a Smart Tourism application”). WINGS ICT Solutions therefore develops the application (hereafter called “Smart Tour in the City application”) using the iCore platform and the Service Templates to match the functions of the requested application; indicatively: “Fastest route to destination”, “Scenic route to destination” and “Avoid traffic”. The application is then sold to CLIPPER travel and SALINA travel who distribute it to their customers (tourists). The tourist is then required to complete a simple registration in order to start using the application (an e-mail will be sent to confirm the registration). The tourist (or tourist group) then arrives in Athens/Santander and starts using the application to help him navigate/explore the city. Upon first use of the application his credentials will be requested (from the registration process mentioned above). He can then select the service he needs from a predefined set (“Fastest route to destination” and “Avoid traffic”) or input his own request in natural language and a destination to be guided to. After the address is entered the Smart Tour in the City application will calculate all the alternate routes from the current location to the target address. According to the requested service(s) the next steps of the process vary. If the user specified that he wants to avoid traffic, the Smart Tour in the City application will then look up the traffic value (indicatively: 1 for low traffic, 2 for medium traffic, 3 for high traffic) for every street in the routes that have been calculated. At this point, the Hadoop service will be used which will provide the traffic values taking into account statistics taken over time as well as the real time traffic value as reported by the Greek police or the Smart Santander facility according to the service request. The Hadoop service focuses on the exploitation of data coming from smart cities so as to build knowledge on traffic information (to facilitate traffic diagnosis and prediction). Afterwards, the traffic values for every street in each route will be added to produce a list and find the one with the least traffic. The route with the lowest value assigned, and therefore the smallest amount of traffic, is presented to the tourist/tourist group. If the user specifies that the “fastest route to destination” is needed then, as

described above, the routes/traffic will be calculated. In this case however, the Smart Tour in the City application will take an additional factor under consideration; the route distance. It will find the best combination of low traffic and short distance and present it to the tourist/tourist group. In all cases, weather information as well as “points of interest” (i.e. shops, gas stations, restaurants, etc.) will be displayed to the user according to his location and profile preferences. Moreover, the tourist (or tourist group) can specify the means of transport they wish to use. The choice made will change the routes generated accordingly. Additionally, as long as the Smart Tour in the City application is operating it sends data (indicatively: requested services, often visited points of interest, preferred means of transport) to the User Preferences and Profiling mechanism in order to be able to further parameterize the service offered to the user. Lastly, the users are asked to provide feedback (rating) concerning their experience with the application upon closing it.

## 2.6.2. Components

For the Smart Tour in the City trial various components of the iCore architecture have been exploited. It should be noted that components developed also for the Smart Home proof-of-concept prototype have been utilised while some additional components have been developed. The list of components includes at the Service level: Service Requestor GUI, Service Template Repository, User Profiling and Preferences, Natural Language Processing, Service Request Analysis, Hadoop, Smart Tour Android application.

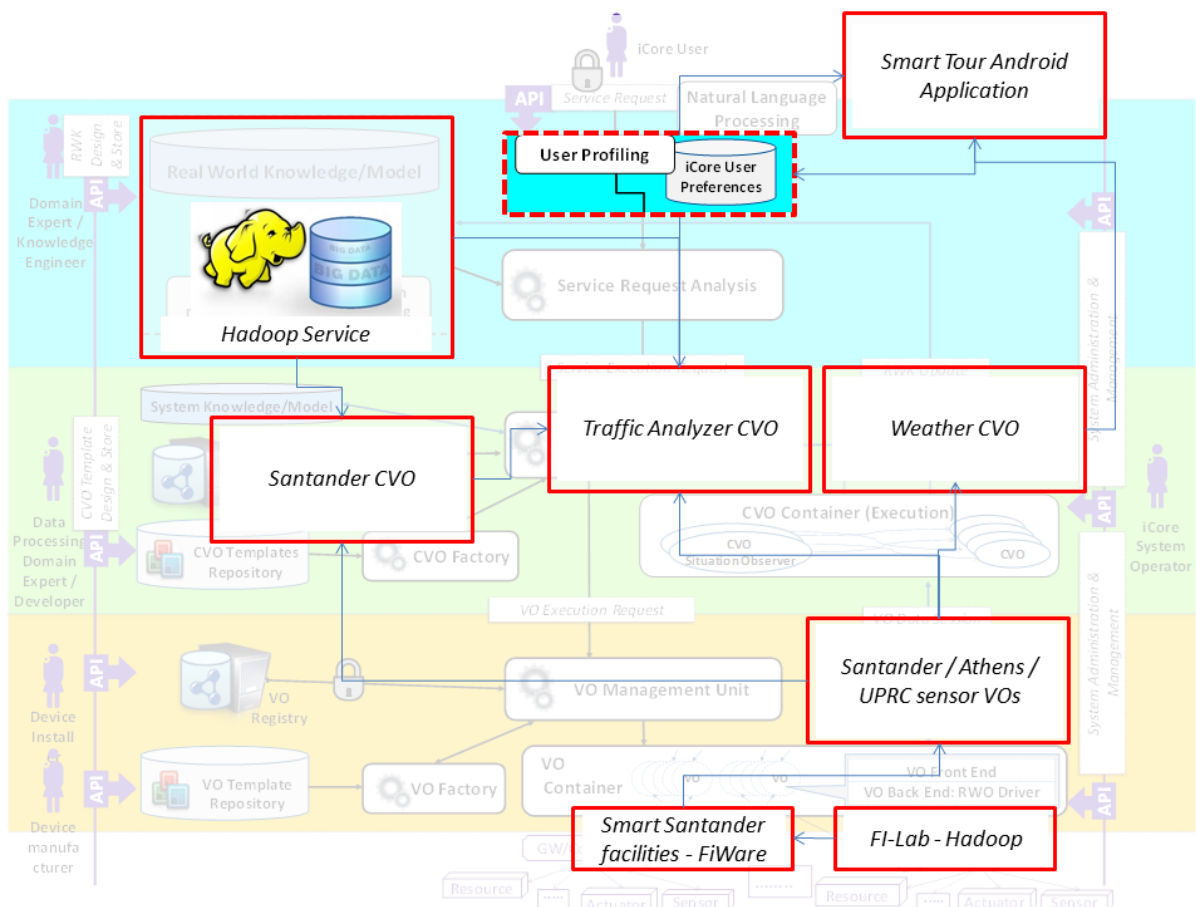


Figure 30: View of additional/enhanced components for Smart Tour in the City trial

The components at the Composite Virtual Object (CVO) Level include: CVO Registry, CVO Factory, CVO Management Unit: Approximation and Re-use Opportunity Detection, CVO Management Unit: CVO Composition Engine, CVO Management Unit: CVO Dynamic Workflow (Planner), CVO Management Unit: CVO Coordination, Traffic Analyzer CVO, Weather CVO, SmartSantander CVO. Finally at the VO level the following components have been developed and used: VO Registry, VO Template Repository, VO Creation Tool (VO Factory), VO Management Unit, VO Container, VO Front-End Software and Back-End REST Modules, Security Agent. More details on the components also developed for the Smart Home proof of concept prototype can be found in section 2.1 as well as in [2]. The additional components, developed specifically for the trial are described in more details in the following. Figure 30 shows the mapping on the iCore architecture of the additional components developed for the trial. Figure 31 depicts a high-level view of the interactions between the additional/enhanced components in the Smart Tour in the City trial during service execution.

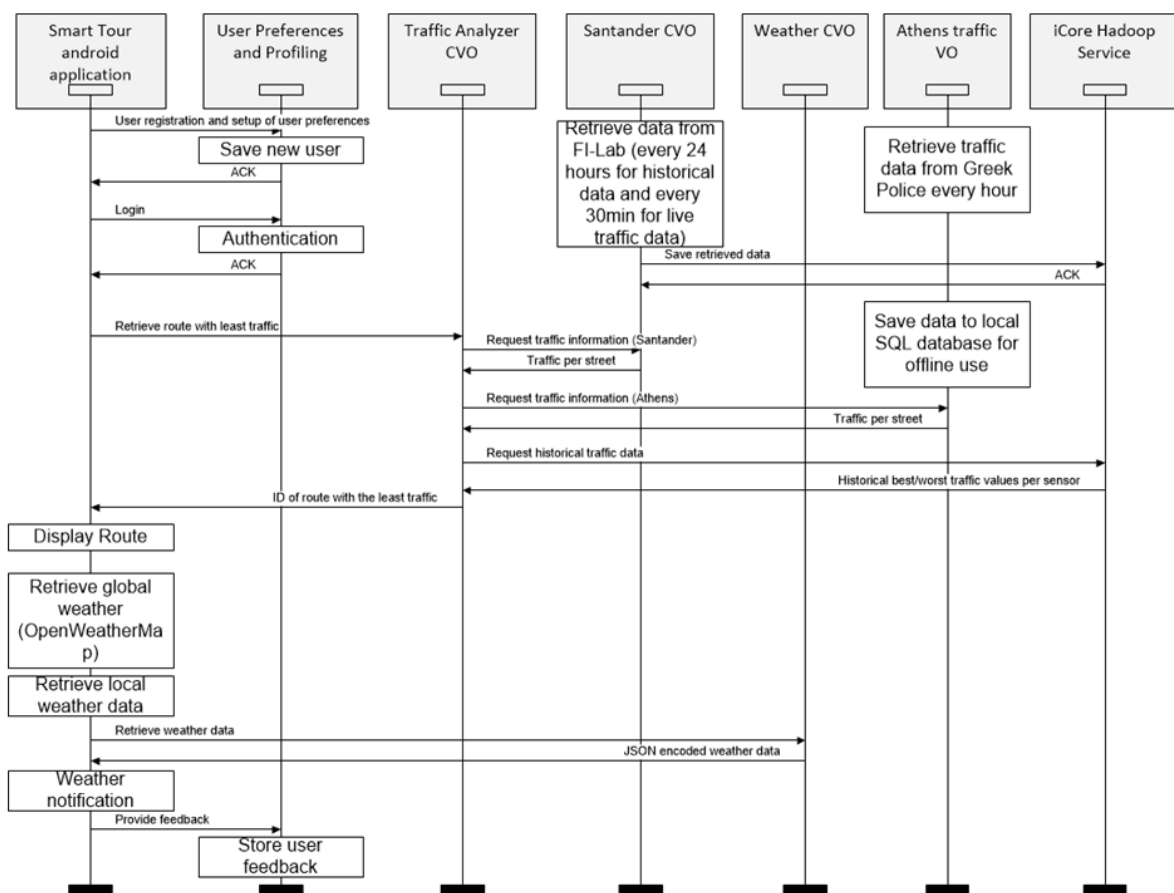


Figure 31: Interactions between additional components in Smart Tour in the City trial

### 2.6.2.1. Smart Tour Android application

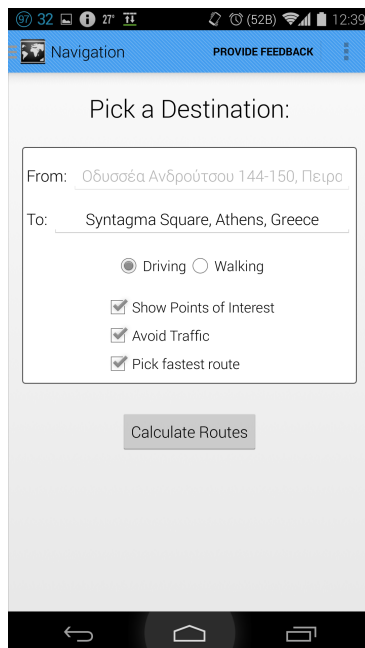
In the context of the Smart Tour in the City trial an android application has been developed with the purpose of guiding tourists in the cities of Athens and Santander. The application can be divided in 4 sections:

1. Navigation
2. Map

### 3. Instructions

### 4. Weather

Starting with the navigation section, the user's location is automatically calculated using the cellular and/or the Wi-Fi network and/or the GPS (if available). The destination field is filled by the user and possible addresses are presented to him after geocoding ([using the Google Geocoding API](#)) the user input.

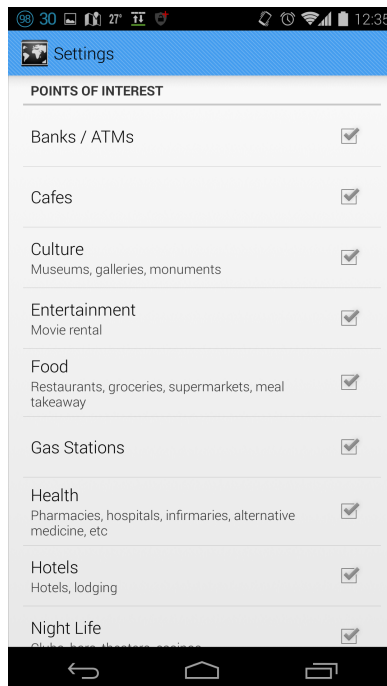


**Figure 32: Smart Tour in the City Android application - Navigation Section**

The possible routes are calculated through the [Google Directions API](#). The user can choose the mode of transport (Driving, Walking) as well as whether to show points of interest on the map, avoid traffic when possible and pick the fastest route:

- **Points of interest:** To present points of interest to the user the [Google Places API](#) has been used. Up to 20 points of interest will be displayed to the user with a radius of 2 km (the center being the user's current location)
- **Avoid Traffic:** This choice is only enabled if the user has selected "Driving" mode. To pick the fastest route the Traffic Analyzer CVO is used.
- **Pick Fastest Route:** This option takes into account the duration/distance of every route (as returned by the [Google Directions API](#)) as well as the route with the least traffic (if "Avoid Traffic" has been selected). The "Fastest" route is then calculated using the combination of these variables.

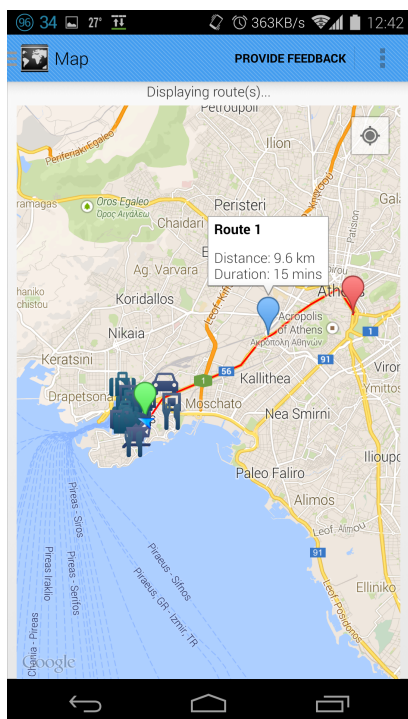




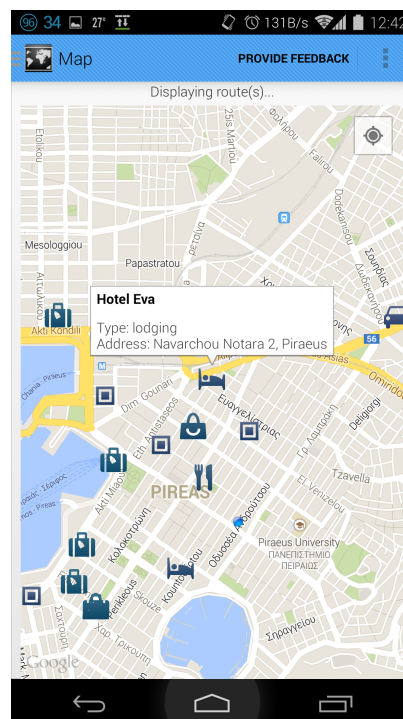
**Figure 33: Smart Tour in the City Android application - Points of Interest Settings**

After the settings are set and the user has set a destination the routes can be calculated by clicking the button labelled “Calculate Routes”. This will automatically switch the user to the Map section to display the route(s) found.

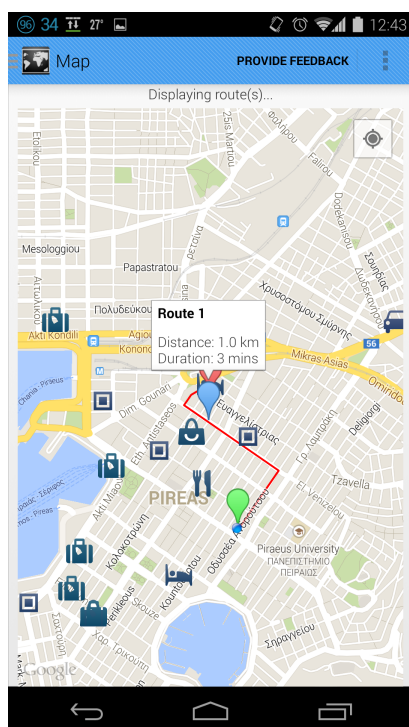
**In the Map section**, as the name suggests, there is a Google map view centred at the user location. If any routes have been found they will be displayed to the user in this section. The points of interest are also displayed on this map. This section offers a second way to calculate routes to a destination. Specifically, the user can click 2 points on the map to calculate a route between them (the first click for the origin and the second for the destination). Additionally, clicking on an information window of a point of interest will calculate a route from the user’s location (or any other point if he has set the origin location by clicking once on the map) to that point. Long-clicking the map results in the deletion of all routes and points drawn on it. In this manner of calculating the routes the settings set in the Navigation section are respected.



**Figure 34: Smart Tour in the City Android application - Map Section**

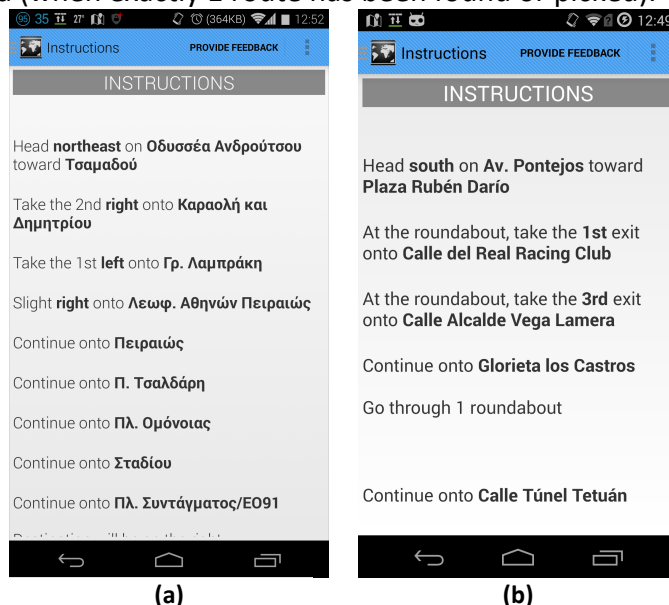


**Figure 35: Smart Tour in the City Android application - Point of Interest information window**



**Figure 36: Smart Tour in the City Android application - Route calculation after clicking on the information window of a point of interest**

In the Instructions section, the user is presented with step-by-step instruction on the route that has been picked (when exactly 1 route has been found or picked).



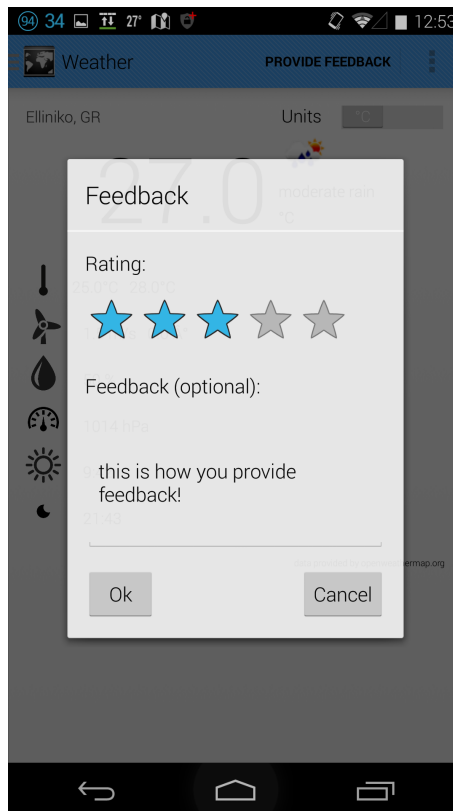
**Figure 37: Smart Tour in the City Android application –  
(a) Instructions Section (Santander); (b) Instructions Section (Athens)**

In the Weather section, the user is presented with weather information based on his current location (using the Openweathermap API).



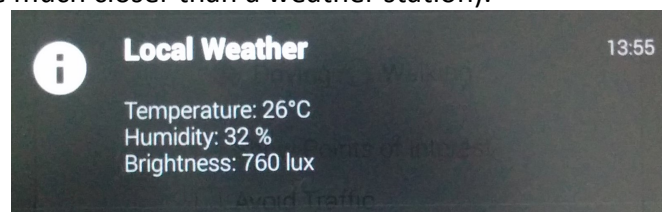
**Figure 38: Smart Tour in the City Android application -Weather Section**

In all the sections presented the user has the option to provide a rating and feedback (using the iCore Profiler CVO) as well as change settings concerning which points of interest will be shown.



**Figure 39: Smart Tour in the City Android application -Feedback Dialog**

Additionally, when the user is inside the sensor range of the weather CVO a notification message will be displayed that contains information about humidity, brightness and temperature (more accurate than the Weather section information as the sensors of the weather CVO will be much closer than a weather station).



**Figure 40: Smart Tour in the City Android application – Weather information display**

### 2.6.2.2. iCore Preferences and Profiling

This component's purpose is to store and retrieve user profiles to/from the iCore system. The user profiles are stored in an SQL database. A profile holds the username, password, email that is given by the user through the registration process on the Smart Tourist application. Additionally, the profile holds the rating and feedback that are optionally given by the user to indicate his satisfaction with the service provided by iCore.

### 2.6.2.3. Traffic Analyzer CVO

This component has been developed using Java, SQL and a RESTful interface. This CVO receives a set of routes and country code and performs the required operations to return the route with the least traffic.

The country codes supported for the trial are “GR” and “ES” (Greece and Spain respectively). Depending on the country code received by the device will perform a different operation (as the traffic information originates from different sources).

In both cases the traffic information is received through a set of VOs.

For the case of Greece the traffic information is received from the Greek police via their web page. It is then stored at regular intervals in a local SQL database (1 hour intervals), so that it can be accessible in case the site is offline, as “Street – Traffic” pairs. The Traffic Analyzer CVO in this case will perform geocoding on the points inside the set of routes received and try to match the street names to those in the database (for performance reasons the geocoding results are cached so that the geocoding only needs to happen once for any given street).

Furthermore, enhancements and restrictions were implemented to increase performance:

- **Caching:** The most expensive operation this program performs is geocoding which takes approximately 1 second every time it’s ran. For this reason a caching mechanism was implemented that saves the points that have already been geocoded so that they will not be geocoded again for the CVO instance. Every point (latitude - longitude pair) in a route is then first checked against the cache before a geocoding operation occurs (a point is considered “cached” when at least one of the points in the cache has a distance of 40 meters at maximum with it).
- **Distance between points:** Each route received by the Traffic Analyzer CVO may have thousands of points. It is therefore a very expensive and even unnecessary operation to check every point. For this reason a minimum distance of 50 meters between points was implemented.

For the case of Spain the Smart Santander CVO is used to retrieve the traffic information from the Smart Santander facilities. Additionally, in the case of Spain, historical data are used via the Hadoop service to perform a more accurate prediction of the traffic conditions of each route.

Each route in the set received consists of points. Each point is a set of coordinates (Latitude and Longitude).

For both cases (Greece and Spain), a score needs to be calculated for each route (the lower the traffic the lower the score). To that end, the CVO loops through every route and considers each point. If there is a difference of at least 50 m (thus reducing execution time considering a route may consist of more than a thousand points) between the current and last point the traffic for the current point is calculated.

At this point the manner of the calculation performed differs for each country. For Greece, the point is geocoded ([using the Google Geocoding API](#)) in order to be checked against the database of the streets monitored by the Greek police. For Spain, the traffic is monitored using a set of sensors (from the Smart Santander facilities). For this reason, the point’s distance from the sensors is checked. The traffic value is considered only if a traffic sensor is near enough the point to be accurate (50 m).

The score of each route is its average traffic.

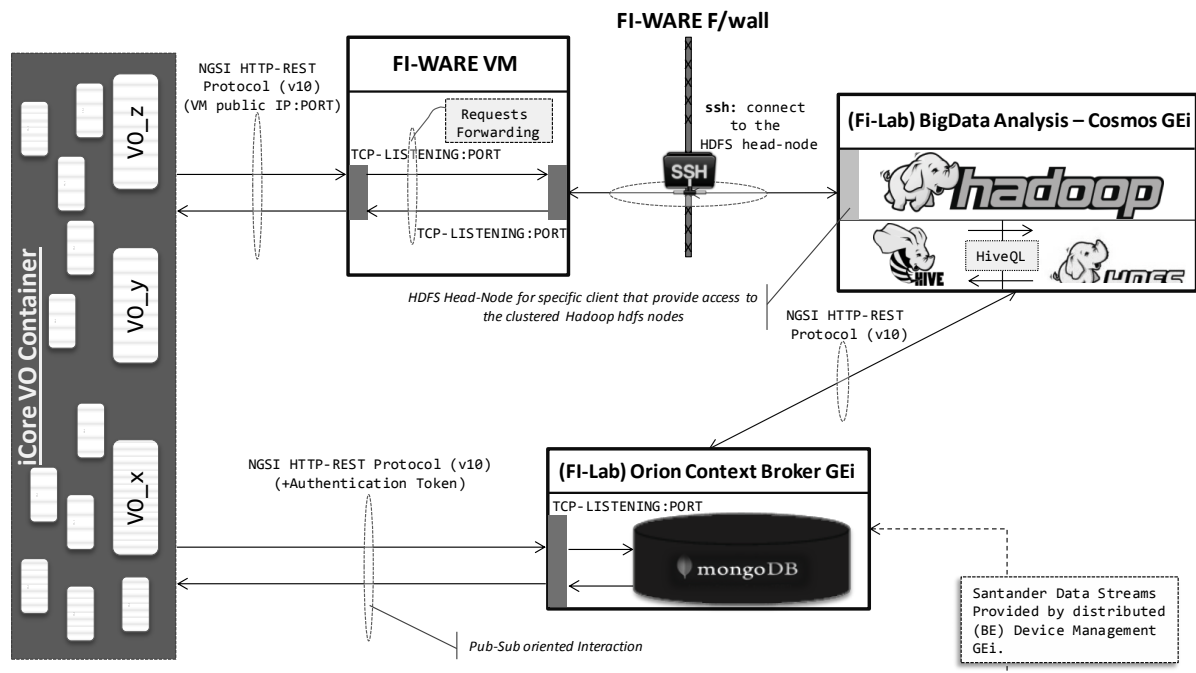
#### 2.6.2.4. Weather CVO

The Weather CVO (hereafter wCVO) is comprised by three different VOs, which correspond to three environmental monitoring sensors that are installed in the TNS Laboratory

headquarters, in Piraeus, Greece. These VOs retrieve measurements for the Temperature, the Humidity and the Luminosity of the ambient, by enabling the CVO to provide each 30 seconds the real-time value of each measurement. As already introduced above, the wCVO is used by the Smart Tourist application, by allowing the user to be aware anytime about a most accurate presentation of the environmental conditions in the area where she/he is moving. For the trial purposes we considered that the user is moving around the TNS Lab building, while the application detects a local weather station and asks the user if he/she would join the weather station information. Since the user accept the joining request, the CVO starts to transfer the measurements of the environmental conditions and the user gets the data on the screen of its smart phone. In this way the Trial concepts could be enriched by distributing several such weather stations, managed by corresponding wCVOs that will take over to provide real-time measurements for the Temperature, Humidity and Luminosity, from different areas / locations in the city of Athens.

#### 2.6.2.5. Smart Santander CVO

The Smart Santander CVO (hereafter ssCVO) constitutes the entity, which bridges the iCore platform with the Smart Santander platform. Specifically, a wide range of separate VOs are composed under the ssCVO and they are governed and manipulated by a common CVO Logic, which aims to the data acquisition from heterogeneous Traffic Management sensors that are distributed in the city of Santander, Spain. The FI-WARE Open-APIs have been used for the development of the VOs that have been deployed in the iCore platform and perform the interoperable connection with the available sensor in Santander. In particular, two (2) different FI-WARE Generic Enablers (GEs) have been exploited so as to structure the main architecture that supports the overall functionality of the ssCVO. The Figure 41 depicts the overall architecture on which the Smart Santander has based its functionality. The VOs get access via a FI-WARE Virtual Machine (VM), which is associated with an iCore account in the FI-WARE cloud. The VM, essentially, offers authenticated open-access to each entity performs any action through its network. Thus, each VO is authenticated so as to use the FI-WARE GEs that will offer access to the Santander data. The first GE that is used directly from the VOs is the Big Data Analysis – Cosmos GE that allows the access to aggregated contextual data of different and diverse sensors from the Smart Santander. The aggregation of the sensor data is performed in the Orion Context Broker global instance, which is hosted in the FI-WARE cloud. Each Wireless Sensor Network (WSN) in Smart Santander infrastructure publishes its data into a local instance of the Orion Context Broker GE, and then each local instance publishes in an automatic way the sensor data into the global instance.



**Figure 41: Bridging iCore with Smart Santander Architecture Overview - Smart Santander CVO**

In this way all data from Santander is aggregated into the global instance, and each VO can get specific information by performing requests to the Cosmos GE, (e.g. to get the last measurement values from a sensor, by using its unique id). Following the above process the ssCVO, through the VOs that belong in its composition, is able to get the last, as well as historical measurements for the each available traffic sensors that is deployed in Santander and connected to FI-WARE cloud.

In the Smart Santander platform, Apache Hadoop is used for distributed processing of the HiveQL queries in order to retrieve real-time and historical measurements. Afterwards, since the ssCVO has acquired these data, sends them to the Hadoop platform that is deployed in the iCore trial so as to infer conclusions on the corresponding situations, e.g. high/medium/low traffic. This process is further explained in the next section.

#### 2.6.2.6. Hadoop

As described in the previous section, the Smart Santander CVO retrieves historical measurements from the deployed sensors in Santander. The output of this function is rough data of no specific meaning for the user. For example, measurements regarding traffic intensity of a specific road may state its traffic condition at a specific moment; however if these measurements are considered individually, no inference can be derived for the actual traffic status of this road. Furthermore, a user would be interesting in the comparison of the traffic status of different roads, in order to avoid the congested ones and select the most suitable routes. This information cannot be derived from the observation of rough data. In addition, it should be highlighted that the magnitude of the retrieved measurements is extremely high (millions of records in the simplest case). It is evident that there is a need for processing of the Smart Santander's data, in order to create knowledge and moreover, this



processing should be done in an efficient way in order to tackle the raised challenges (e.g. size, execution time).

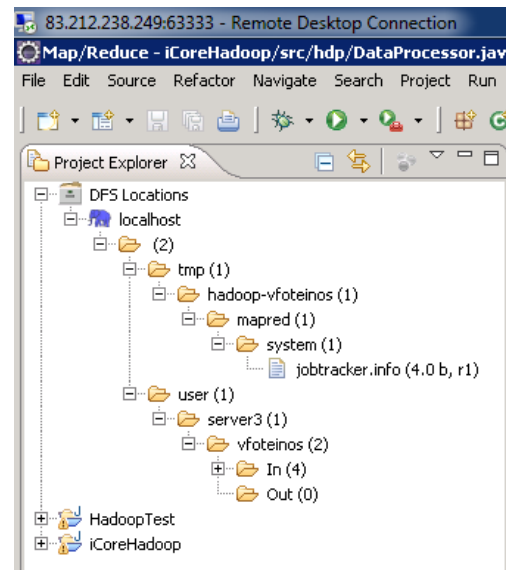
For this purpose, we developed a mechanism that lies at the Service level and is responsible for processing the rough data and deriving the real world information. It can be accessed through two distinct REST interfaces, one for receiving the measurements from the Smart Santander CVO and the second for sending the derived statistics (to the Traffic Analyzer CVO). Apache Hadoop framework version 0.20.2 (Figure 42, Figure 43) was exploited for the actual processing of the data. Hadoop is considered as the most suitable solution for large-scale processing of datasets.

```

vfoneinos@SERVER3 ~/hadoop-0.20.2
$ bin/hadoop namenode -format
cygwin warning:
MS-DOS style path detected: C:\cygwin\home\vfoneinos\hadoop-0.20.2\build\native
ve
Preferred POSIX equivalent is: /home/vfoneinos/hadoop-0.20.2/build/native
CYGWIN environment variable option "nodosfilewarning" turns off this warning.
Consult the user's guide for more details about POSIX paths:
  http://cygwin.com/cygwin-ug-net/using.html#using-pathnames
14/09/28 16:41:46 INFO namenode.NameNode: STARTUP_MSG:
=====
STARTUP_MSG: Starting NameNode
STARTUP_MSG: host = SERVER3/83.212.238.249
STARTUP_MSG: args = [-format]
STARTUP_MSG: version = 0.20.2
STARTUP_MSG: build = https://svn.apache.org/repos/asf/hadoop/common/branches/b
ranch-0.20 -r 911707; compiled by 'chrisdo' on Fri Feb 19 08:07:34 UTC 2010
14/09/28 16:41:47 INFO namenode.FSNamesystem: fsOwner=server3\vfoneinos,root=
t,Administrators,Users
14/09/28 16:41:47 INFO namenode.FSNamesystem: supergroup=supergroup
14/09/28 16:41:47 INFO namenode.FSNamesystem: isPermissionEnabled=true
14/09/28 16:41:47 INFO common.Storage: Image file of size 107 saved in 0 seconds
14/09/28 16:41:47 INFO common.Storage: Storage directory \tmp\hadoop-vfoneinos\d
fs\name has been successfully formatted.
14/09/28 16:41:47 INFO namenode.NameNode: SHUTDOWN_MSG:
=====
SHUTDOWN_MSG: Shutting down NameNode at SERVER3/83.212.238.249
=====
vfoneinos@SERVER3 ~/hadoop-0.20.2
$ bin/start-all.sh
starting namenode, logging to /home/vfoneinos/hadoop-0.20.2/bin/../logs/hadoop-v
foneinos-namenode-SERVER3.out
localhost: starting datanode, logging to /home/vfoneinos/hadoop-0.20.2/bin/../lo
gs/hadoop-vfoneinos-datanode-SERVER3.out
localhost: starting secondarynamenode, logging to /home/vfoneinos/hadoop-0.20.2/
bin/../logs/hadoop-vfoneinos-secondarynamenode-SERVER3.out
starting jobtracker, logging to /home/vfoneinos/hadoop-0.20.2/bin/../logs/hadoop
-vfoneinos-jobtracker-SERVER3.out
localhost: starting tasktracker, logging to /home/vfoneinos/hadoop-0.20.2/bin/../
logs/hadoop-vfoneinos-tasktracker-SERVER3.out
vfoneinos@SERVER3 ~/hadoop-0.20.2

```

**Figure 42: Formatting HDFS and starting Hadoop daemons**



**Figure 43: Browsing HDFS location (through Eclipse)**

At first, a dedicated web service (REST-based) was developed for receiving requests. A request consists of the necessary info (e.g. request ID, type of data) and an indication of where data (measurements) are stored. Based on the request, two processes are considered, a short-scale process which includes data belonging to a specific period (e.g. last month) and a large-scale one where all available historical data are examined. From the Smart Santander CVO, the following data can be retrieved: i) Traffic intensity, ii) Humidity, iii) Luminosity, iv) Presence. Upon reception of the request, data are forwarded to the Hadoop file system and processed based on the developed map/reduce functions. The output of this operation is a set of (non-)optimal IDs (corresponding to sensors in roads) for different cases (e.g. traffic intensity, luminosity, etc.). Therefore, it is possible to identify the most congested roads or the less congested ones for different time periods. In this manner, real world information is derived and can be used from the end-users.

Through another web service (REST-based), the Traffic Analyzer CVO issues its requests. An indicative request is: {"traffic intensity", "overall", "high"} and the response is a set of sensor IDs and their values. For this request, taking into account the available historical measurements, these sensors were found to be the most congested ones. The corresponding roads will be avoided, if the user has requested for an optimal route. Furthermore, the rest of the environmental-oriented data (humidity, luminosity, presence) are used in a similar way offering exploitable real world information to the users.

### 2.6.3. Hardware

The hardware used for demonstration purposes is a Google Nexus 5 device (2.3 GHz quad-core CPU, 2 GB RAM, android version 4.4.4). In practise however the hardware requirement from an iCore user is any android device (version 4.1 and above) that supports Google location services.

## 2.7 Smart Hospital Trial

### 2.7.1. Overview

Modern hospitals are complex logistic systems composed of a number of departments and rooms, medical equipment (asset) including portable ones and personnel. The assets are often migrate from one department to another or are misplaced. Accurate positioning of assets ensures their quick localization in the case of emergency and makes them available for periodic maintenance.

The Smart Hospital trial is done at the Santa Chiara Hospital in Trento, Italy, in the Newborn (Neonatologia) department. There are many medical devices in the department to guarantee the daily work and ensure high-quality medical service. Currently roughly 240 devices are present inside the department and have to be monitored. The list of these devices includes monitor, personal computer, medical equipment and all the electronics devices available inside the department. In order to guarantee good healthcare services, it is also necessary to ensure a well-managed environment, able to track, monitor and maintain all the available technologies and devices that are present inside the department.

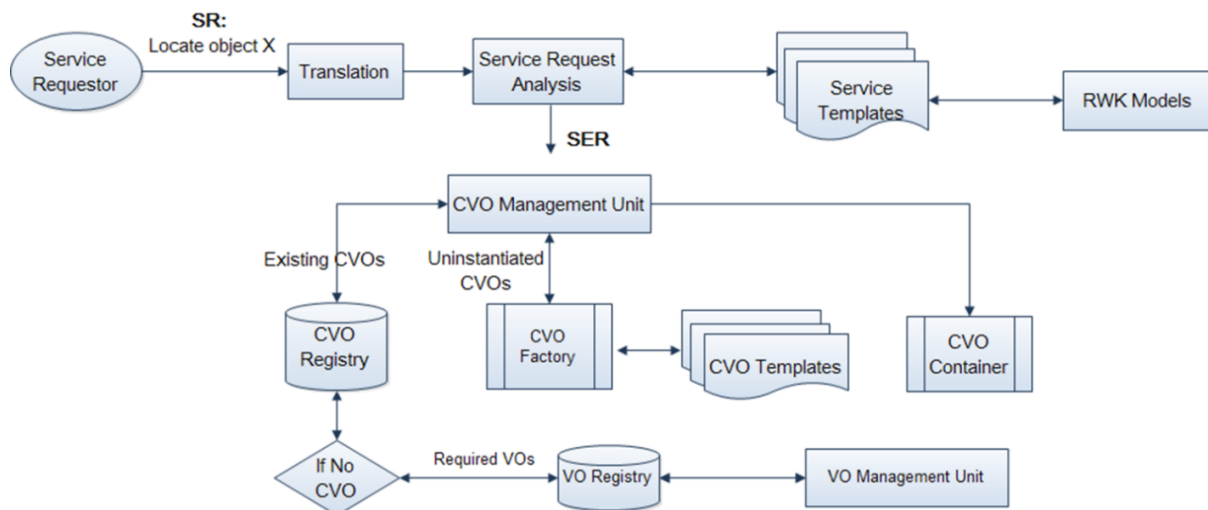
The trial deployment comprises of tracking and monitoring devices realized by the real time location system solutions provided by ZIGPOS (eeRTLS) and the software engine 'Treelogis' provided by Trilogis. This technology enables the ability to geo-fence objects and provides the interfaces to translate the local coordinates of the devices into global coordinates. Both core technologies interact with iCore to provide a flexible, easy to use solution for the various requirements in the asset management and additionally ensuring energy conservation. Each of the medical devices to be tracked is tagged with sensors from ZIGPOS. In addition to which sensors will be deployed in the rooms, which enables the triangulation of the position of a tagged object. More details on the ZIGPOS solutions are provided in Section 2.7.4.

The iCore framework implemented in the described scenario is able to monitor the location of the various devices, set geo-fencing areas and time-fencing. This generates alerts when the device enters/exits the defined geo-fence area and alerts when the device is not returned to the location within a specified time-fence. It also monitors the maintenance schedule of the devices and sends alerts to the administrators/ maintenance personnel if a device is due for service. The idea of using this iCore solution is designed to monitor how many times and where a specific device is used. Information about the usage time is not procured directly for now. For this reason we correlate the movement of the object with the fact that it is used. Another important feature of the trial is the use of the Real World Knowledge. Real world information is captured and is used to derive prediction models based on movement patterns of the objects. The purpose of the real world knowledge model is two-fold. The first is to ensure the continuance of service, i.e. when the real time location information of the objects is unavailable; the prediction model provides the

estimated location of the devices. The iCore framework functions as the decision making engine to select the available prediction models at real time ensuring they satisfy the defined performance criteria. A global RWK model capturing the relationships, selection criteria, thresholds has been implemented which facilitates in this decision making process. A relational database model in MySQL is the design implementation choice for the same. Further prediction models based on the occupancy patterns at both the device level and at a room level provide predicted occupancy patterns. This is then used to provide recommendations.

## 2.7.2. Components

Components of Smart Hospital trial are described in D5.4 in details. In this section, we summarize what components have been used with respect to the trial. Figure 44 represents the iCore components used in the trial.



**Figure 44: Architectural Overview of the iCore components used in the Smart Hospital Trial**

The major components can be summarized as follows:

### 2.7.2.1. Service Level Components

The service level receives the SR in the form of natural language. A translation module extracts the functional tokens from the service request based on natural language libraries. The Open NLP library serves the purpose of extracting the token. While in the implementation, only features of token extraction have been utilized, the library can be easily extended to implement more features of the natural language processing. The translated SR is then provided to the service request analysis block which generates the Service Execution Request (SER) that comprises of the list of VOs and CVOs that are required to fulfil the SR. The service templates are workflow based models implemented using the Drools library. In addition a real world model exists, capturing the properties and relationships between the various devices, which is exploited to decide on the best execution path on the basis of the current parameters in the real world.

### 2.7.2.2. CVO Level Components

The CVO level components include the CVO management unit which receives the SER, looks up for the required VOs and CVOs. The CVO factory and CVO repository together provides this information. While the CVO registry maintains all the information of the currently deployed CVOs, the CVO factory holds information about all the available CVOs. In the implementation, the CVO registry is limited to looking up for the available CVO templates. A CVO template interface is defined and implemented by various CVOs, which are defined in WP4 deliverables. In short, the available CVO templates are (i) Locate\_CVO – on execution; it provides the location of the device. This CVO interacts with the software components of Trilogis via MQTT protocol to retrieve the location information (ii) Trace\_CVO – on execution, it helps to trace an object and provides geo-fence and time-fence updates. The geo-fence events are triggered when an object leaves the defined geo-area and the time-fence events are triggered when an object does not return to the go-area in the defined time-frame (iii) Maintenance\_CVO- provides all the maintenance related information i.e. scheduling the devices for maintenance, checking the status of the devices and informing the system administrators and maintenance personnel of any requirements (iv) Recommendation CVO – This is used to provide recommendations of power plans to the indoor positioning system. The recommendation CVO relies on the RWInformation and the prediction models derived therewith. In the trial, the prediction CVOs are developed in collaboration with partners TU Delft leveraging their expertise in machine learning and statistical models. The RWK models capture the RWInfo, in this case the movement patterns of the objects and the occupancy patterns in the rooms in build a prediction model. The Autoregressive models and Markov models are implemented. The models are developed on the basis of the test data collected from the test-bed set up in the premises Trilogis and the movement of the employees are modelled. For the deployment in the hospital, it is presumed that we will have similar data patterns and the models are to be tweaked according to the patterns available in the hospital. These models are incorporated as CVO templates which will be called upon by the CVO management unit if predicted information is required. The predicted values are utilized by the recommendation CVO along with a rule base of the energy model of the ZIGPOS positioning system in order to provide recommendations.

The modelling process includes a data collection, data pre-processing and a model development phase. The data collection phase included collecting data of the movement patterns from the positioning tags. Data sample was collected every 30 seconds with timestamp and location information. In the data processing step, aggregation of two consecutive samples is done in order to eliminate transitions and missing information in the dataset is eliminated by appending the previous location if found. A Markov chain is used to represent our RWK model. Markov chain is a mathematical system that undergoes transitions from one state to another on a state space. It is a random process usually characterized as memory-less: the next state depends only on the current state and not on the sequence of events that preceded it. A stochastic matrix (Transition Probability Matrix) TPM, describes a Markov chain  $X$ .

In general, if the probability of moving from  $i$  to  $j$  in one time step is  $P_r(j|i) = P_{i,j}$ , the stochastic matrix  $P$  is given by

$$P = \begin{pmatrix} p_{1,1} & p_{1,2} & \dots & p_{1,j} & \dots \\ p_{2,1} & p_{2,2} & \dots & p_{2,j} & \dots \\ \vdots & \vdots & \ddots & \vdots & \ddots \\ p_{i,1} & p_{i,2} & \dots & p_{i,j} & \dots \\ \vdots & \vdots & \ddots & \vdots & \ddots \end{pmatrix}. \quad \sum_j P_{i,j} = 1. \quad (1)$$

The TPM captures the probability of an object to move from one location to another. TPM can be calculated in several ways a) time-invariant b) time-dependent. Here the RWK model calculates TPM for each and every object with both time-invariant and time-dependent features. These are envisaged to be used in the prediction of the next location of object.

Two prediction models namely Auto Regression (AR) and Markov Model (MM) are considered. The goal of prediction mechanism is to accurately predict the next location of the object so that the sensor on the device being tagged can be put to sleep mode to save energy.

(i) Auto Regressive Model

An autocorrelation based transversal filter can be described as follows:

$$u(t) = \sum_{i=1}^p u(t-i)\alpha(i), \quad (2)$$

where  $u(t)$  is the predicted time-series value at time instance  $t$ ;  $[u(t-1), u(t-2), \dots, u(t-p)]$  are the previous  $p$  values of time-series;  $[\alpha(1), \alpha(2), \dots, \alpha(p)]$  are the filter coefficients; and  $p$  is the order of the autocorrelation model. Once the filter coefficients are determined, the current value of the series can be estimated using the past values. To find the filter coefficients, a set of training data is required from the time series. Suppose, there are  $T_p$  training data ( $u(1), u(2), \dots, u(T_p)$ ) available from the series. Then, the filter coefficients can be found by solving the following linear equations –

$$U\alpha = \underline{u}, \quad (3)$$

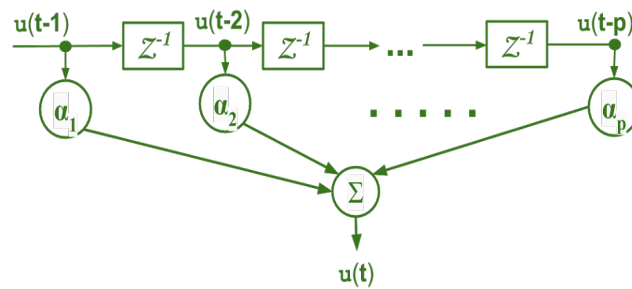
$$U = \begin{bmatrix} u(p) & u(p-1) & \dots & u(1) \\ u(p+1) & u(p) & \dots & u(2) \\ \vdots & \vdots & \ddots & \vdots \\ u(T_p-1) & u(T_p-2) & \dots & u(T_p-p) \end{bmatrix}, \quad (4)$$

$$\alpha = [\alpha(1), \alpha(2), \dots, \alpha(p)]^T, \quad (5)$$

$$\underline{u} = [u(p+1), u(p+2), \dots, u(T_p)]^T. \quad (6)$$

is an over determined system of linear equations, as the number of equations ( $T_p - p$ ) are more than the number of variables ( $p$ ). As  $U$  is not a square matrix, a unique solution is not possible, and an approximate solution can be found using least-square method. The least-square solution of the above equation can be found using the following equation –

$$\underline{\alpha} = (U^T U)^{-1} U^T \underline{u}. \quad (7)$$



**Figure 45: AR Model**

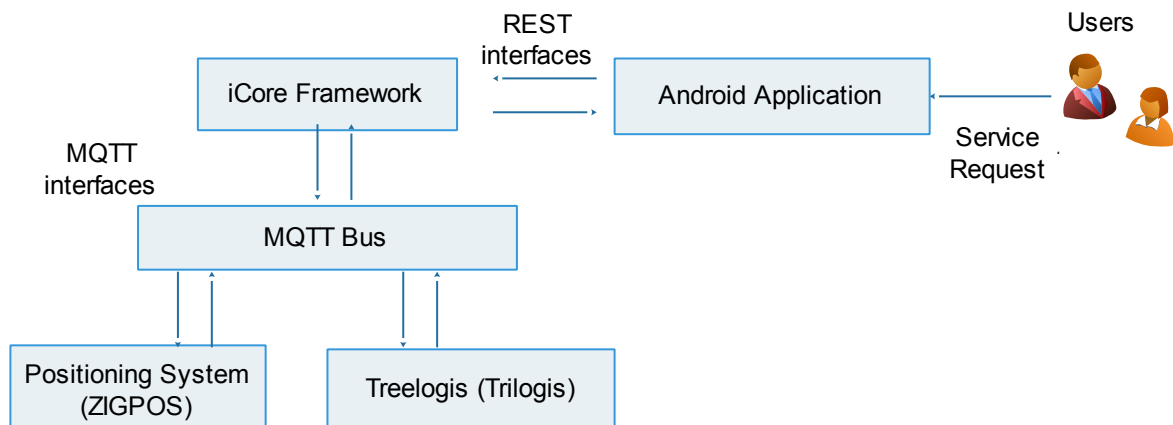
In a nutshell (as shown in Figure 45), autoregressive (AR) process models the conditional mean of  $u(t)$  as a function of past observations,  $u(t-1), u(t-2) \dots u(t-p)$ . An AR process that depends on  $p$  past observations is called an AR model of degree  $p$ , denoted by  $AR(p)$ . Experimental results on the model are demonstrated in Section 3.7.1.7

### 2.7.2.3. VO Level Components

The VO level is responsible for creating the virtual objects, in this case, the medical objects that are tagged with the ZIGPOS positioning tags. The virtual objects are stored in the Virtual Object Registry as soon as these are connected to the positioning system. The virtual objects are enhanced with additional information e.g. co-ordinates of a “geo-fence” associated with the real-world object (to enable notifications of “out-of-fence” besides change of location), as well as “time-fence” for all purposes related to managing maintenance, objects out of place for too long etc.

As far as fostering interoperability is concerned, the communication layer used by the Virtual Objects has been implemented using a publish/subscribe bus (the MQTT broker). This is used for all the notifications such as change location and out-of-fence (worth noticing here that this is all agnostic of the positioning technology used and hence largely reusable). Upper layer queries are on the other hand answered using a REST-based client/server communication with the VO Registry. In the specifics of this trial the flexibility given by the VO Registry consisted in accommodating a data model where each Virtual Object has a location, as well as a fencing perimeter and a time-fence for notification of various messages related to object (medical device) status.

#### 2.7.2.4. Android Application



**Figure 46: Interactions between the Android application and other software components**

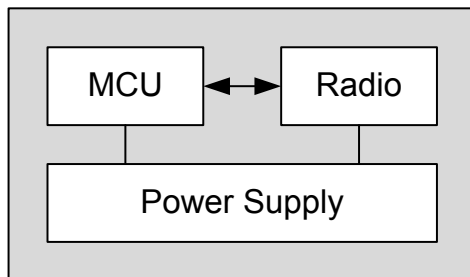
In addition to the above mentioned blocks, an Android application would be available for the users to issue service request. The android application is being developed in co-ordination with Innotec21 in order to leverage their expertise in the mobile application development. The input service request (SR) which is fed into the iCore framework represented in Figure 44, thus comes from the android application. The interactions between the android application and the other partners/components in the trial are represented in Figure 46. The user issues the SR via the Android application. This SR is provided to the iCore framework which interacts with the positioning system provided by ZIGPOS and also the geo-location services provided by Trilogis in order to provide a response to the SR. The iCore framework runs on a Jersey based REST serve and the interactions with the mobile application is based on RESTful interfaces. In addition to the represented components, there exists a database for the iCore framework, which stores the outcome requested by the SR. This facilitates faster response back to the android application. The database is MySQL based. There are three types of service requests:

- (i) Location requests for which the location of the devices are represented on a map
- (ii) Trace requests for which the geo-fence and time-fence events, both of which are events generated when an object enters or leaves a geo/time fence, is represented as an alert in the mobile application and finally
- (iii) Maintenance service requests which is also represented as alerts in the mobile application.

#### 2.7.3. Hardware

In this trial we use wireless sensor nodes by ZIGPOS Figure 47 presents the architecture and picture of the node. It can be used and configured in the trial as ‘anchor’ or ‘tag’. The anchor nodes have fixed positions with hard-coded coordinates. A tag node is physically connected with a medical equipment to be tracked. The sensor nodes are featured by low power consumption and accurate localization.





a)

b)

**Figure 47: Wireless sensor node used in the Smart Hospital trial: (a) hardware architecture and (b) photo of the node.**

There are several different modes within the sensor node acting as a tag. The minimum current consumption (as for theory / data sheets) for the radio devices is:

Modes	Current Consumption <sup>7</sup>
Ido (always on)	0.8μA
Flash deep sleep	1.0μA
μC deep sleep	0.7μA
bma deep sleep	0.5μA
voltage divider for batty monitoring (always on):	0.15 μA
sum theory	3.15 μA
sum measured	5 μA

#### 2.7.4. ZIGPOS Localization Technology

ZIGPOS eeRTLS real-time location and tracking system operates over wireless sensor and actuator networks and provide the possibility to estimate accurate position to any of its members. The communication is based on the global standard IEEE 802.15.4, 2.4 GHz frequency band enabling interoperability with existing wireless solutions. The system consists of following devices:

**MOBILE Tags:** The mobile tags will be located in the wireless networks. People, devices will be equipped with those tags

**ANCHOR nodes:** The anchor nodes are fixed at a steady position in the localization network.

**Gateway:** The Gateway is the interface between the localization networks and the IT-Infrastructure. At least one Gateway is required for the whole networks. The Gateway itself provides the localization server, that evaluates the positions, handles requests via dedicated

<sup>7</sup> Current consumption valid at 3,6V, 25 degree Celsius

interfaces and provide the ability to the management interface.

Basic properties:

- Low cost and low-power solution
- Supporting different networking topologies (Mesh, tree, etc.)
- Supporting reliable and secure protocol stack
- Operating complete independent mode, no satellite components are required
- Easy integration to existing/third party systems with standardized interface
- Offer customized interface to third party tools and systems
- System consists of MOBILE Tags, ANCHOR nodes, ROUTER devices, GATEWAY devices
- System include ZIGPOS RTLS software components

Applications:

- Indoor, outdoor and mixed scenarios
- Real time tracking and positioning of objects and person in various scenario
- Tracking assert, objects and people in medical hospital for hospital management
- Tracking assert, objects and people in warehouse, shopping mall, retailer
- Facility guidance of visitor in hospital, conference, exhibition centres, etc.

Technical Data:

- Accuracy: up to 30 cm depending on the environment and deployed infrastructure
- Communication: Communication based on IEEE802.15.4, energy efficient low rate 250kBit/s (up to 2 MBits/s in proprietary modes)
- Lifecycle: Current consumption: active < 16mA/sleep < 2µA, e.g. 1Hz [2.000mAh], up to years
- Update-cycle: System offer different response time from milliseconds to seconds
- Scalability: The system is scalable to monitors hundreds of the MOBILE Tags
- Path: Support Non-light of sight
- Coexistence: Coexistence with Bluetooth and Wi-Fi (offer to select different communication channel)
- Battery of mobile nodes: Rechargeable

### 2.7.5. Energy Consumption Data

To estimate the energy consumption of unique devices inside the network we take the Beacon Interval (1.22s) and their static Timeslots (15,36ms). Depending on the current tasks for each device one can simply add the used Blocks to solve the tasks and estimate the power consumption.

#### 2.7.5.1. Block Overview

The eeRTLS system is based on a time-synchronized communication and distance estimation approach. Therefore the smallest period in time (15,36ms) can be used to simplify the energy consumption for each hardware device depending on the used block.

In total we have 6 blocks to fulfil all tasks:

Beacon Rx  
(Receive of the beacon to keep time synchronization), valid for ALL devices (anchors and mobiles)

**1**

Time Synchronisation and information transportation:  
(Difference between empty beacon and filled beacon with information (who has to do what)

Energy consumption empty Beacon (ESB @3.6V):

**149,6021 mWms (ca. 4,15562E-05 mWh)**

Energy consumption full Beacon (ESB @3.6V): **311,5373 mWms**

EMPTY Slot  
Equal for Anchor or Mobile

**2**

Empty slot, where device does nothing and is in sleep mode!"

Energy consumption empty Slot (ESB @3.6V):

**0,331776 mWms**

PMU ranging  
Equal for Anchor or Mobile

**3**

PMU high accurate,

The device will do an accurate measurement

Energy consumption empty Slot (ESB @3.6V):

**413,373096 mWms**

RSSI  
Mobile

**4.1**

RSSI Mobile

The mobile tag is sending a packet that can be received by various anchors

Energy consumption empty Slot (ESB @3.6V):

**106,0229 mWms**

RSSI  
Anchor

**4.2**

RSSI Anchor

The anchors have to wait for receiving the expected packet from a mobile tag and forward the packet with given RSSI to the Router/Coordinator

Energy consumption empty Slot (ESB @3.6V):

**392,8574 mWms**

Communication

**5**

Communication (transmit packet with information), for now just use full packet transportation

Energy consumption empty Slot (ESB @3.6V):

**194,6669 mWms**

#### 2.7.5.2. Accuracy mapping

Level	Synonym	Accuracy	description	Needed blocks
0	Rough RSSI	X	Rssi based Position based on one Packet broadcasted from mobile to various anchors	Anchor:[1]+[4.2] Mobile:[1]+[4.1]
1	Simple PMU	X*4	PMU measurement with 4 anchors	Each Anchor:[1]+[3] Mobile:[1]+4*[3]
2	Accurate PMU	X*6	PMU measurement with 6 anchors	Each Anchor:[1]+[3] Mobile:[1]+6*[3]
3	Averaged PMU	X*10	2 Measurements with 6 anchors each (over 2 Beacon Periods)	Each Anchor:[1]+[3] Mobile:[1]+6*[3] → 2 times
4	High Accurate	X*20	5 Measurements with	Each Anchor:[1]+[3]

	PMU		each time 6 anchors (over 5 beacon Periods)	Mobile:[1]+6*[3] → 5 times
--	-----	--	--	-------------------------------

In total we have 79 Blocks per Beacon Interval, which means we have to add empty blocks until every BI is filled for every single device.

So it has to be added **(79-NumberOfUsedBlocks)\*BLOCK[2]**

In level 3 and 4 there is more than just 1 Beacon Interval needed, so this has to be taken into account as well.

### 2.7.5.3. Example Calculation

Requested position accuracy level of a mobile Tag [M1] is 3:

The mobile tag will try to estimate the position out of distances to 6 anchors. Therefore it has to receive the Beacon and make a PMU measurement 6 times:

$$E = E_{\text{beacon}} + 6 * E_{\text{pmu}} + (79-7) * E_{\text{sleep}}$$

$$E = [1] + 6 * [3] + 72 * [2]$$

$$E = 311,5373 + 6 * 413,373096 + 72 * 0,331776$$

$$E = 2.815,66 \text{ mWms} \rightarrow 7,82e-4 \text{ mWh}$$

### 2.7.6. OTAU

Update functionalities are very important in case a heavy change in firmware needs to be done. Due to the fact that the Stack on a single device is optimized in size and calculation resources it could happen, that a security update or functional update is only feasible with a total compiled new software stack. For easy upgrading an over-the-air firmware flash is implemented, that simply transmits new firmware via REST-interfaces to a specific device. Here a number of packets need to be transmitted. The radio node is kept waking up after a beacon is received and several 10s to 100s of packets are exchanged. In addition the power mode (c2) has to be activated, so this costs a lot of energy, but is happening only very rare.

## 2.8 Smart Theme Park Trial

### 2.8.1. Overview

This trial illustrates how the iCore functionalities can be utilized in entertainment attractions like theme parks. The motivation of this trial is to provide a personalized digital souvenir for theme park customers. This trial, on one hand, could capture the valuable experience of customers in the format of digital multimedia while they are enjoying the events. On the other hand, it creates a new product and theme parks could make profits from it.

The main idea is to deploy various sensors including digital cameras, microphones, RFID sensors, infrared sensors around the thrilling rides to record expressions and screams of riders, starting and stopping time of rides and identification of riders. All these multimodal information are analyzed and fused to generate personalized video clips for each customer.

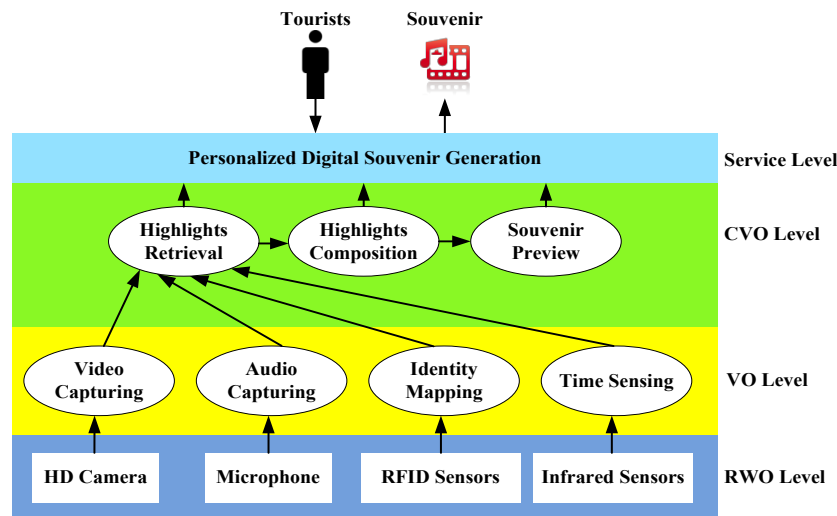


Figure 48: Architecture of Smart Theme Park trial based on iCore.

## 2.8.2. Components

The architecture of this trial is illustrated in Figure 48. The key components of the architecture are described in the following sections.

### 2.8.2.1. Digital Souvenir Generation (service level)

This component provides a simple GUI for the operator to offer service to customers and results in the dynamic creation/instantiation of the appropriate Composite Virtual Object (CVO) via interactions with other iCore components,. It has been implemented using a number of technologies including Java Servlets, AJAX, PostgreSQL, as well as a RESTful interface to facilitate the connection between the Servlets and the iCore components.

### 2.8.2.2. CVO Level

- CVO management unit  
At a service request, the CVO Management Unit tries to find the needed running CVO from CVO registry. If failed, it turns to the CVO factory to activate and instantiate a new one.
- CVO registry  
Maintain the id and semantic description of the running CVO.
- CVO factory  
The CVO factory is able to instantiate a CVO based on a CVO template. It answers to CVO Management Unit requests, which handles a description of the CVO functionalities.
- Highlights retrieval CVO  
This CVO searches personal video and audio data from the Postgres database using customer information like name, phone number, RFID EPC as key words.
- Highlights composition CVO  
This CVO edits and combines the personal video and audio data into a single digital souvenir in time order. It is implemented based on the open source Libav libraries from the FFmpeg project.
- Souvenir preview CVO

This CVO enables customers to preview their personal souvenirs. It is implemented as a streaming server using the free FFserver tool from FFmpeg project.

### 2.8.2.3. VO Level

- VO management unit

This component receive VO execution request from CVO level and search the VO registry for the required VO. If no VO is available, it interacts with the VO factory component to activate the VO and register it to VO registry.

- VO registry

The VO registry in Smart Theme Part Trial contains information about each sensor. The information describes the associations between VOs and sensors. Each VO is identified by a “Uniform Resource Identifier” (URI).

- VO factory.

This component is responsible for the creation of VOs. Once a VO is created, installed and deployed it is registered in the VO registry.

- Video capturing VO.

This VO maps to the real world digital cameras. Visual images are recorded and compressed by consumer digital cameras, which are installed on each seat of the ride and adjusted to be able to capture the detail expression of the rider. The recorded video streams from each seat are first saved to local storage and transmitted wirelessly to the backend servers when the ride vehicle returns to the station.

- Audio capturing VO

Similar to video capturing VO, audio capturing VO capture sounds including screams, laughs made by the riders during the ride using microphones installed on seats. The audio data are also transmitted through wireless links when the vehicle returns back.

- Identity mapping VO

This VO establishes the mapping between customer and the seat-mounted digital camera. This mapping information plays a key role for highlights retrieval CVO. The VO is implemented by attach a passive RFID tags to each seat-mounted camera. When the riders are seated, the security inspectors scan the wrist band of each rider and the tag on the camera installed on the same seat using a handheld smart terminal to set up the mapping, which is then transmitted wirelessly and stored in databases for further use.

- Time sensing VO

This VO records the time when ride vehicle departs from and returns to the station. Infrared sensors are deployed around the start and end of riding track to detect the passing vehicles.

### 2.8.3. Hardware

- RFID scanning terminal



BayNexus SDA (Smart Digital Assistant), running Android 4.0 system, equipped with 2.4GHz RFID reader and 802.11b/g/n wireless interface card.



Figure 49: RFID scanning terminal

- Data collection terminal

Cubietruck, running Ubuntu 12.10, equipped with Allwinner A20 CPU, 64GB SD card, 802.11g/n wireless interface, and Ethernet interface.

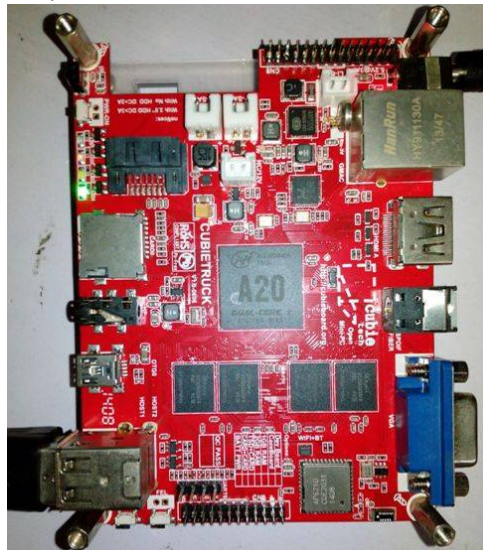


Figure 50: Data collection terminal

- Digital camera

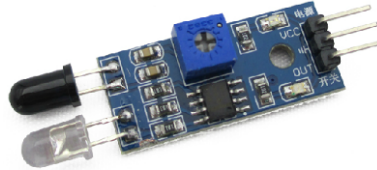
Logitech Webcam C920, with embedded microphone. This camera uses H.264 encoded compression to provide high quality video at low bit rates and is connected to the Cubietruck develop board via USB 2.0.



Figure 51: Digital camera

- Infrared sensor

This sensor could detect objects within the range of 30-60 cm. It is connected to the Cubietruck board via the GPIO pins and deployed around the track of ride vehicles. Sensor data are transmitted via LAN to backend servers.



**Figure 52: Infrared sensor**

## 3 Technical validation

### 3.1 Smart Home: Living assistant

This section presents results for various test cases for the Smart Home proof of concept prototype. It should be noted that these results have to a great extent already been presented in previous Deliverables [2],[4]. Here the results are presented in a more complete way, e.g. providing the association with functional requirements, etc.

#### 3.1.1. Test case #1 – Dynamic CVO creation

##### 3.1.1.1. Description

This test case addresses dynamic CVO creation, i.e. the creation of a new CVO “from scratch” when there is no prior knowledge or no suitable CVO is found that can be re-used.

##### 3.1.1.2. Entities involved

The entities involved at the service level are: Service Requestor GUI, Service Template Repository, User Profiling and Preferences, Natural Language Processing and Service Request Analysis. The components at the CVO level include: CVO Registry, CVO Factory, CVO Management Unit: Approximation and Re-use Opportunity Detection, CVO Management Unit: CVO Composition Engine, and CVO Management Unit: CVO Dynamic Workflow (Planner). Finally at the VO level the following components are used: VO Registry, VO Management Unit, VO Container, VO Front-End Software and Back-End REST Modules, Security Agent.

##### 3.1.1.3. Related requirements

Table 2 lists the functional requirements (according to deliverable D2.1) related to this test case.

**Table 2: Requirements related to Smart Home test case #1**

Identifier	Name	Description
SL-F-5 i	iCore external actors shall be able to instantiate new logic and reuse existing logic for dynamic iCore service composition by means of newly generated and existing iCore objects.	This is the systematic ‘programmability’ of an environment of iCore objects.

SL-F-9.	The iCore system shall enable the proper and efficient composition of iCore objects that fulfil the query from the service level.	The iCore system shall be able to configure/monitor/select iCore objects on the basis of what the services requested by external actors need.
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-12	The iCore system shall be able to aggregate and infer relevant situational dimensions (e.g. time, place, preference of users) to describe the current situation and anticipate changes to it.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-13	The iCore system shall be able to <b>infer</b> (reason) on user intentions/behaviour in order to support current and future services and service requests.	Importance of storing relationships between objects, people, applications. The iCore system must be adaptable to User intentions/behaviour for dynamic service composition.
SL-F-14	The iCore system may have an interface where users can explicitly describe or query his/her intentions/behaviour if necessary regarding his/her current situation (iCore is supposed to	Simple API to communicate user intentions, semantic interrogation (SPARQL) for current situation and personal history. Feature is addressed to the cases where “observation” capability at Service Level can’t perform or direct information from the user when demanded.

	be able to sense those changes).	
SL-F-15	The iCore system shall maintain and be able to store, classify, rank, and annotate a categorisation of different situations based on user intentions/behaviour per situation for future use	It is an essential element for the iCore system to 'grow' intelligence and foster reuse of iCore objects in multiple services and across multiple situations.
SL-F-17	The iCore system shall be able to identify an approximate CVO quickly to match requested services as best as possible.	It is not always possible to find exact services for a user all the time. Thus, if an exact CVO which was supposed to provide a particular service is not available, we have to identify an approximate service. The approximation is done based on the availability of alternatives and possibilities. Depending on the situation, there might be a threshold to decide and select an approximate service (i.e. alternative service).
CVO-F-1.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-2.	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters).
CVO-F-3	The CVO functionality shall be able to autonomically select and integrate the correct VOs that describe and match the current / potential service	A CVO must be able to evaluate the service function offered by VOs and reconfigure itself if necessary by selecting / maintaining the best VOs.

	requirements.	
CVO-F-5	The CVO registry shall be able to obtain Service Logic requirements either by a query (when they are stored somewhere) or by a trigger event (when they are constructed on the fly).	The CVO registry must be able to interface with the upper service level and be able to receive service logic requirements (request parameters).
CVO-F-6	The CVO registry shall be able to address and name stored VOs and CVOs.	Stored VOs and CVOs must have an address and name in order to be retrieved, e.g. as web services.
CVO-F-7	The CVO registry shall be able to store VO interaction history based on the user situation description framework or Category.	A CVO must store information about the CVO components and correlate them with the situation.
CVO-F-8	The CVO registry shall be able to maintain a set of past relationships with other CVO's or VO's in order to create an environment/CVO space for Service composition.	The CVO registry must keep a history of activated CVOs and their components.
CVO-F-9	The CVO shall be enabled with an auto construction of a "scheduler".	A CVO must somehow evolve a schedule to use each VO (and its functionalities) in a sequence (some in parallel) in order to achieve a required service.
CVO-F-11	A CVO shall be created based on a template that is	Templates are necessary for a semi-automatic generation of CVOs from data acquisition and events processing.

	produced at design time and intends to define the CVO and determine his capabilities.	
CVO-F-12	A CVO shall be validated before the execution, based on the received, complete and accepted non-conflict situation with a clear lifespan for execution.	Validation is necessary in order to add in valid CVOs against SLA rules (we need to insure that the decomposition of a SLA will result in a potential CVO discovery) and to insure that CVO execution will not generate bad results (e.g. circular loops or system crashes).
CVO-F-17	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters).
VO-F-1	The VO Registry shall be able to maintain and record the links with each RWO (i.e. send notification (signalling) when coupled and when decoupled).	Handle to the real world is maintained in a best possible way. Awareness regarding the RWO.
VO-F-2	The VO shall be able to name, type (Metadata) and address each RWOs that becomes available (i.e. at the point of discovery).	RWOs are appropriately named and addressed within their respective proximity domains.
VO-F-3	A VO shall enable the representation of proximity. It must also represent the relationship of proximity between multiple RWOs.	A VO must be able to publish its natural relation with another VO and represent these as two related RWOs.



VO-F-4	A VO shall be located easily within the proximity domain.	VO discovery for entities within the same proximity domain. Registry of the VO shall be located within the proximity domain.
VO-F-5	The VO functionality shall be able to dynamically describe RWOs and match (tag) the correct RWOs to current CVO / Service / Application requirements.	New RWOs can be added to the iCore system during run-time. Once VO functionalities tag with new RWOs then it also becomes a new VO and the old VO may survive for a certain time.
VO-F-6	A VO shall describe the available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe-threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
VO-F-7	A VO shall be able to know the current location (i.e. Be location aware) of the RWO it is representing.	The location of the RWO shall be abstracted at the VO level.
VO-F-8	A VO registry shall notify a CVO Registry of new VOs that become available for service composition and re-use.	The VO discovery interface between VO and CVO levels. A VO has the capability to advertise/notify itself for CVO composition.
VO-F-10	A VO shall be able to represent a single RWO or many RWOs.	A VO may be formed out of multiple RWOs.
VO-F-12	A VO shall provide a semantic description of a RWO including its control and configuration interfaces.	Description of features provided by the RWO, how to access these features (e.g. call a web service, send an event, etc.) and how to configure parameters. The relationship of the RWO and its resources shall be described at the VO level.

#### **3.1.1.4. Pre-conditions**

Service templates, CVO templates and VO templates must have been defined. VOs and related Real World Objects need to be installed.

#### **3.1.1.5. Execution steps**

Initially a user request and the policies are captured through the User Interface at the Service level and they are forwarded to the NLP entity, which executes the natural language processing task by exploiting the NLP. Then, the SRA entity derives the required logic, CVO and VO Types to fulfil the requested service. This information along with specified policies is forwarded in the form of a so called Service Execution Request to the CVO Management Unit. At first, the AROD searches for an already existing CVO that can fulfil the request, in the CVO registry. If the AROD infers that the reuse of a CVO is not possible, then it forwards the relevant information to the CCE. This latter entity proceeds to an optimal composition of VOs, according to requested functions and policies. For this purpose, the VO Registry provides the CCE, through the VO Management Unit, with information on available VO features and functions. As soon as the CVO composition has been decided the CVO creation is performed by the CVO Factory and the new CVO is deployed in the CVO Container. The CVO composition, as well as parameters related to the service request and associated situation are stored in the CVO registry. Finally, the created CVO provides the requested service to the user.

#### **3.1.1.6. Success criteria/ Expected output**

A CVO is created that is in line with the requirements provided at the service level. The created CVO works as intended. The involved mechanisms perform well in terms of producing the expected outcome as well as in terms of time required to fulfil their operation.

#### **3.1.1.7. Test Case Result**

Figure 53- Figure 56 provide a high-level view of the operation of the main components involved in the dynamic CVO creation.



Figure 53: Service request triggers dynamic CVO creation process

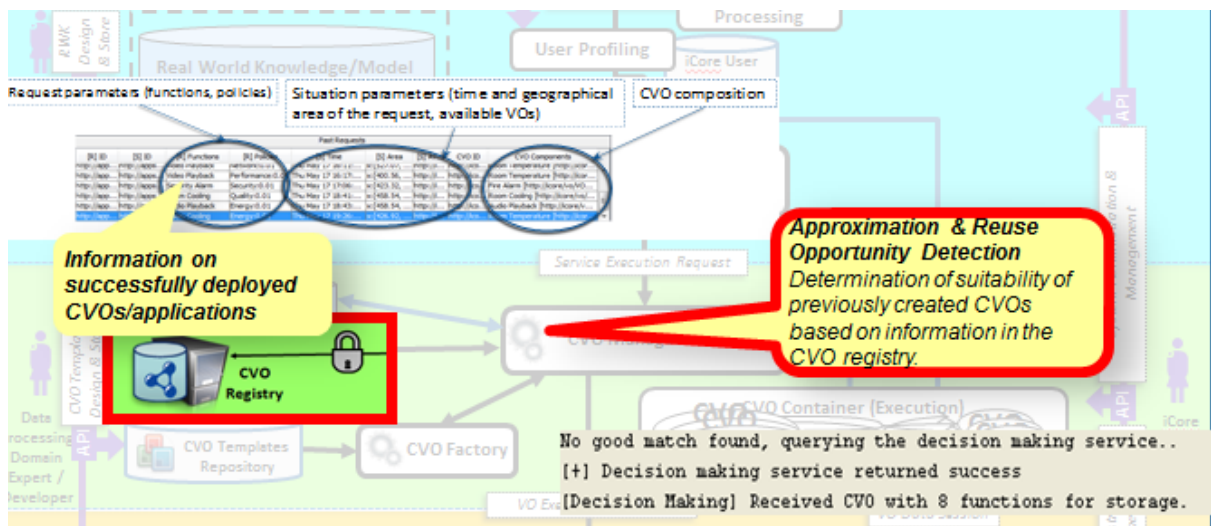


Figure 54: Search for an existing CVO that can fulfil the request

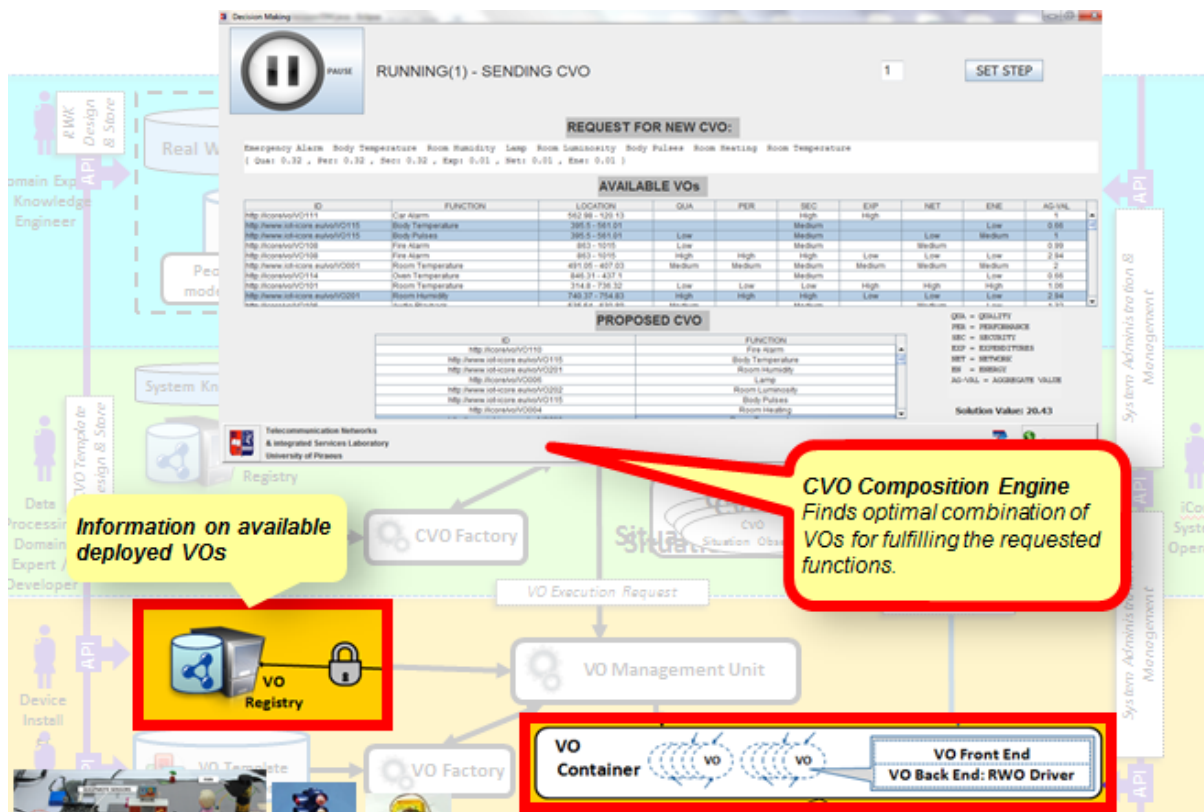


Figure 55: CVO optimal composition of VOs, according to requested functions and policies

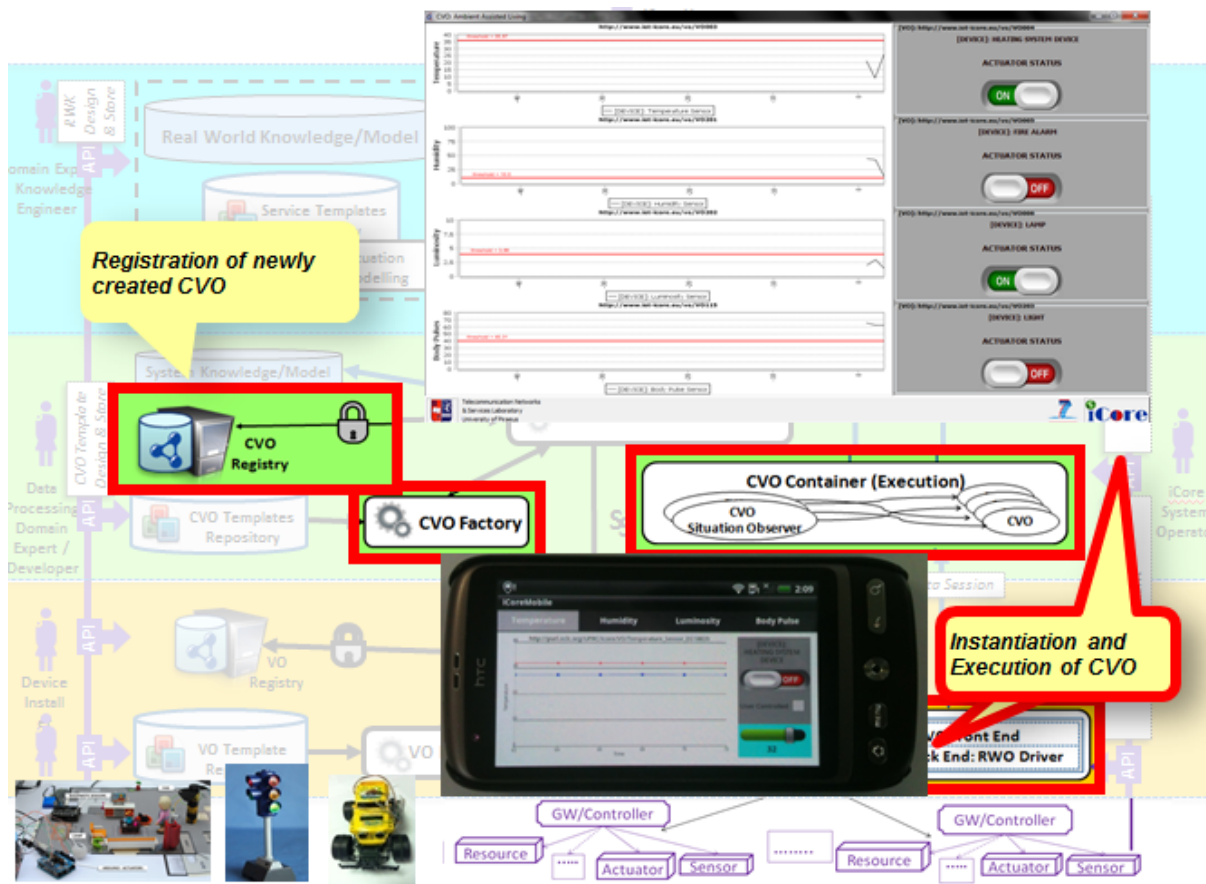
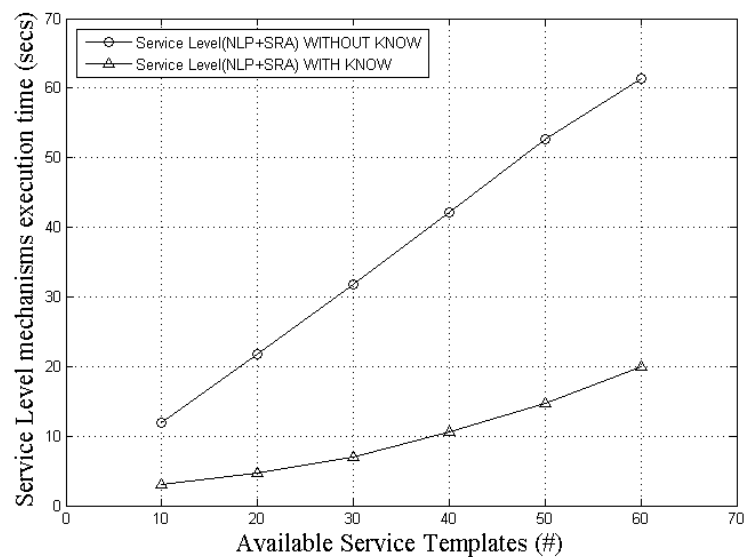


Figure 56: CVO registration and instantiation

Figure 57 presents results on the performance of Service level mechanisms with and without prior knowledge about a given user request. In the case that no prior knowledge is available all available Service Templates have to be searched in order to select the most appropriate one and derive corresponding VO and CVO information. Thus, this approach is dominated by the amount of available Service Templates and the amount of nodes comprised in each of the Service Templates. In the case of available knowledge on a particular user request, Bayesian statistics are exploited for the derivation of the optimal Service Template, thus eliminating the need of looking into each node of each Service Template. Therefore, the overall process of selecting the most appropriate service template in this case depends only on the number of Service Templates. As can be observed in Figure 57, when prior knowledge is available less than the half time is required for processing a user request by the Service level mechanisms. Generally, the goal of automated users' requirements analysis and matching for the dynamic creation and configuration of CVOs with the aim of providing smart, personalized services through the selection of the most appropriate Service Template is fulfilled in a realistic time-frame considering that the Service level mechanisms execution take place during an initial, design-time phase. Further optimization of the Service level mechanisms execution, is scheduled for future work.

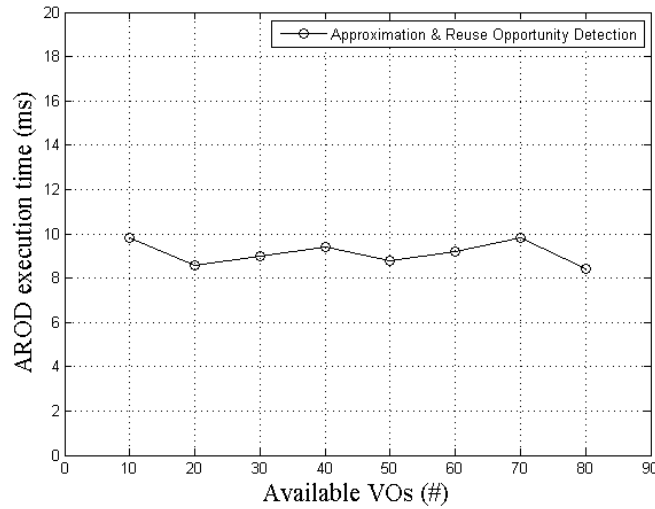


**Figure 57: Service Level execution time vs. Number of Available Service Templates**

In the following results on the performance of CVO and VO level mechanisms are presented. In particular, the time required for the execution of the AROD and the CCE entities as well as the VO discovery processes is presented. More specifically, results for eight different cases of the number of available VOs that could possibly be part of the created CVO are presented. In addition impact of the diversity of the structure/nature of each VO in terms of its representation data is investigated. The eight different cases correspond to increasing numbers of the available VOs ranging from 10 up to 80 and the diversity of the amount of information that correspond to each VO. The information for each VO is represented as a meta-data graph, which has a specific size, depending on the VO features. The graph includes associated meta-data containers with specific meta-data properties and each container corresponds to a specific VO feature, such as the VO functions, function inputs, outputs, etc. Each graph has its own size that is expressed in terms of the number of statements, where each statement represents a meta-data property. It should be noted that some of the VOs were emulated, i.e. did not correspond to the actual RWOs, while in each case, the average number of statements was between 130 and 150 statements per VO. The diversity of the VO information size and the different number of VOs, affect the overall execution time of the related processes in a linear increase of time. The experiments have been repeated five times for each case. Although the outcomes were quite similar in each case, the provided results correspond to the mean times, in order to be more accurate. In all cases, the requested CVO was instantiated successfully.

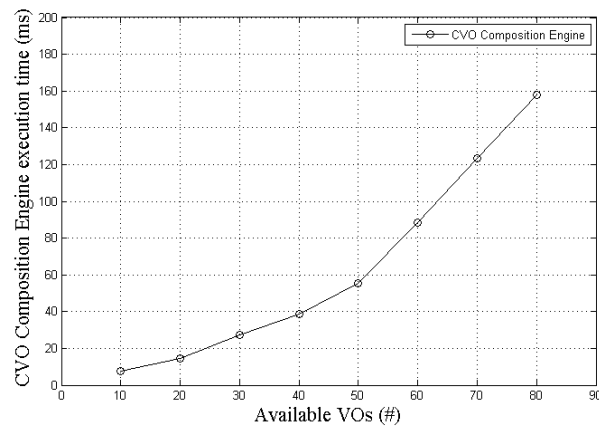
Figure 58 depicts the time required for the operation of the AROD entity, which searches for appropriate CVOs based on specific criteria, which correspond to situation and request parameters. This execution time is quite constant and very low (around 9 msec). It should be noted that only a small number of previously recorded CVOs are considered, i.e. two. Furthermore, it is observed that this operation is not directly affected by the number of available VOs in the area of interest, but by the number of the available CVOs that match the search criteria and include a specific number of VOs in their composition.





**Figure 58: Approximation & Reuse Opportunity Detection execution time vs. Number of Available VOs**

Figure 59 presents the time required for the CCE entity to select the optimal CVO composition according to the available VOs that were suggested by the AROD entity, while information about these VOs is retrieved from the VO registry. This time corresponds to the optimization process of selecting the appropriate VOs for the instantiation of the optimal CVO. The operation of CCE is based on a heuristic algorithm that aims at identifying a suitable composition in minimum time. The algorithm splits the problem into smaller separate ones (Divide and Conquer approach). Its objective is to find the optimal solution for each requested function separately and afterwards, to compose them into an optimal composition of functions. In particular, each function of the available VOs is evaluated, taking into account the defined policies and the features. For each requested function, one optimal function of a VO is found. At the end of this process, a number of functions of VOs are selected for providing the requested functions. The same VO may be exploited for more than one requested function. As can be observed (and as expected) the time required for the selection of appropriate VOs increases as the number of available VOs increases, as there are more VOs to be considered. However, it remains within realistic boundaries, especially considering that the CCE process for the selection of VOs will take place initially during the (automated) creation of the CVO.



**Figure 59: CVO Composition Engine execution time vs. Number of Available VOs**



Figure 60 and Figure 61 depict the increase in the execution time of the Dynamic Workflow (Planner) component when either the number of goals or the *Levels* of the plan were increased. However, only a few dozens of milliseconds were needed in each case.

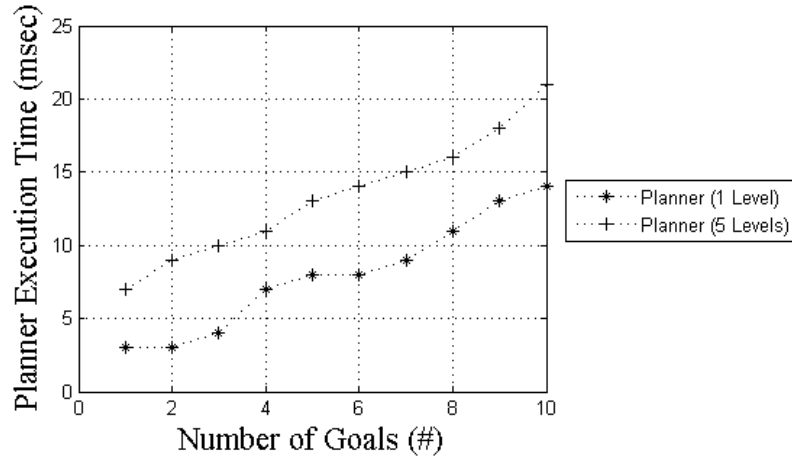


Figure 60: Planner execution time vs. Number of goals

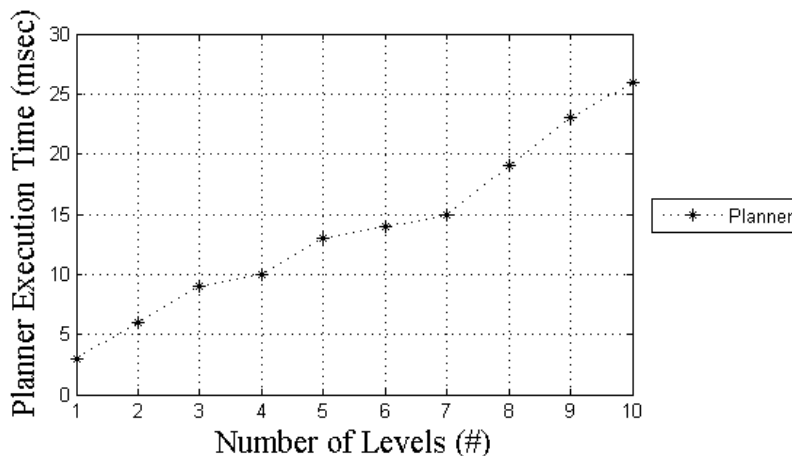
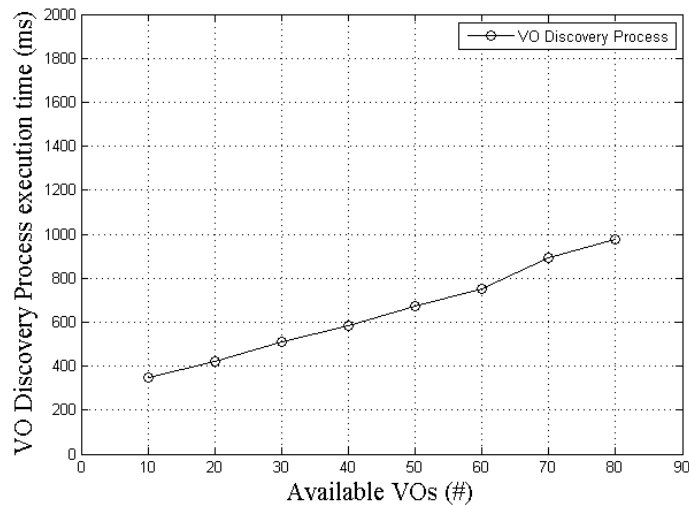


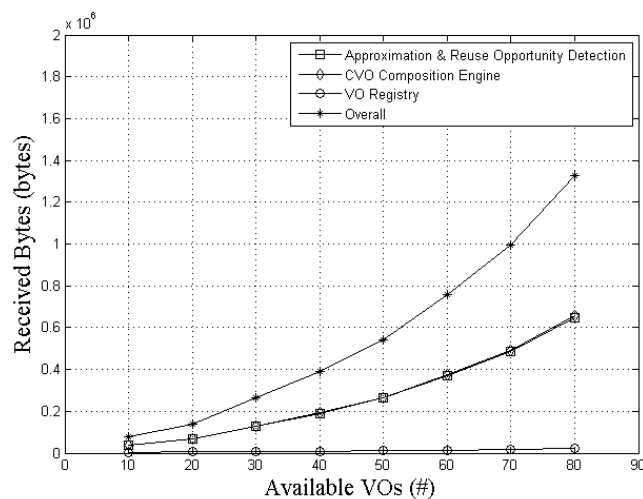
Figure 61: Planner execution time vs. Levels of plan

Figure 62 presents the time required for the VO discovery process, which actually includes the time for the discovery and acquisition of information of relevant available VOs. As can be observed, the time required for retrieving information on the available VOs, linearly increases as the number of relevant VOs in a certain area of interest increases. However it remains realistic especially when considering that this time is related to the initial discovery of VOs for the automated creation of a CVO, and not during run-time of a particular service/application. Further mitigation of the VO discovery time could be achieved by using distributed VO registries and parallelization of the VO search.



**Figure 62: VO Discovery Process execution time vs. Number of Available VOs**

Finally, the amount of bytes that were received from the main entities involved has been measured and is depicted in Figure 63. The number of the received bytes is similar for the AROD and the CCE mechanisms. This is anticipated as the major portion of this data corresponds to the information that is retrieved from the VO registry for the requested VOs. As expected this amount of bytes increases as the number of available relevant VOs (in a particular area of interest) increases. On the contrary, the number of bytes that are received by the VO registry is quite stable, low and is due to the VO IDs only sent by the AROD and/or the CCE mechanisms. Again this amount of bytes increases as the number of available relevant VOs (in a particular area of interest) increases.



**Figure 63: Received Bytes vs. Number of Available VOs**

### 3.1.2. Test case #2 – CVO Self-healing

#### 3.1.2.1. Description

The self-healing process enables the detection of a problem in the operation of a used VO and enables the dynamic reconfiguration of the corresponding CVO to overcome the problem.

### 3.1.2.2. Entities involved

The entities involved are: CVO Management Unit: Approximation and Re-use Opportunity Detection, CVO Management Unit: CVO Composition Engine. Finally at the VO level the following components are used: VO Registry, VO Management Unit, VO Container, VO Front-End Software and Back-End REST Modules.

### 3.1.2.3. Related requirements

Table 2 lists the functional requirements (according to deliverable D2.1) related to this test case.

**Table 3: Requirements related to Smart Home test case #2**

Identifier	Name	Description
SL-F-4	The iCore system shall offer a way to handle dynamic iCore Objects/RWOs connections and disconnections gracefully, impacting service execution as minimally as possible.	Applications and services must be able to “carry-on” with minimal features rather than “hang” if objects are disconnected.
SL-F-8	The iCore system shall be aware of changes when they occur on RWOs during the service runtime.	The iCore system must be able to review RWOs in accordance with current system status (i.e. availability of RWO resource, how RWO's are performing and behaving, whether new RWO's of the same type are required, to prioritise RWOs according to identified service requirements, etc.).
CVO-F-13	A CVO shall be validated during the execution when possible as required by the Application.	This requirement will be possible when a change in SLA is sensed, validated and the described situation for the currently executed CVO to be replaced/corrected <b>OR</b> , when the execution part of iCore fails at a certain stage during the lifespan of the current situation and demands for a repair/replace action of the whole or parts of the CVO including possible service level communication. Since Service Level is only allowed to manage the SLA, this requirement can impose transactional capabilities.

CVO-F-14	A CVO shall be enabled with functionality to support service recovery after an internal or external failure or due to mobility.	Composition rules are needed in order to insure a “repair” capability for CVOs (e.g. the replacing of VO/CVO segments with an equivalent VO/CVO when part of the execution chain is compromised).
CVO-F-15	CVOs shall be enabled to adapt to the Mobility of the iCore system (or part of it) or user.	This requirement deals with the changes that are manifested because of mobility of actors and devices of the iCore system.
VO-F-1	The VO Registry shall be able to maintain and record the links with each RWO (i.e. send notification (signalling) when coupled and when decoupled).	Handle to the real world is maintained in a best possible way. Awareness regarding the RWO.
VO-F-9	A VO shall notify CVO registries of its current status, availability and reliability for re-use.	The VO discovery interface between the VO and CVO levels. A VO shall have capability to update/modify itself and then notify the CVO for composition.

#### 3.1.2.4. Pre-conditions

VOs and related Real World Objects need to be installed. A CVO needs to be running.

#### 3.1.2.5. Execution steps

The self-healing process supports the dynamic detection of a problem in the operation of a VO that is included in the CVO composition and realizes its dynamic reconfiguration. Specifically, when a VO failure is identified, e.g. due to a RWO becoming unreachable, because of loss of connectivity of the VO with the RWO, RWO hardware failures, etc., the CVO level is informed by the VO Management Unit and a reconfiguration request is forwarded to the CCE entity for finding an appropriate replacement for the problematic VO. The CCE performs the discovery and the selection of the most suitable alternative VO, in

order to replace the problematic VO. Finally, information for the reconfiguration of the CVO is recorded in the CVO registry.

### 3.1.2.6. Success criteria/ Expected output

A running CVO continues to operate well after a problem with one or more of the VOs it comprises occurs. . The involved mechanisms perform well in terms of producing the expected outcome as well as in terms of time required to fulfil their operation.

### 3.1.2.7. Test Case Result

Figure 64- Figure 66 provide a high-level view of the operation of the main components involved in the CVO self-healing process.

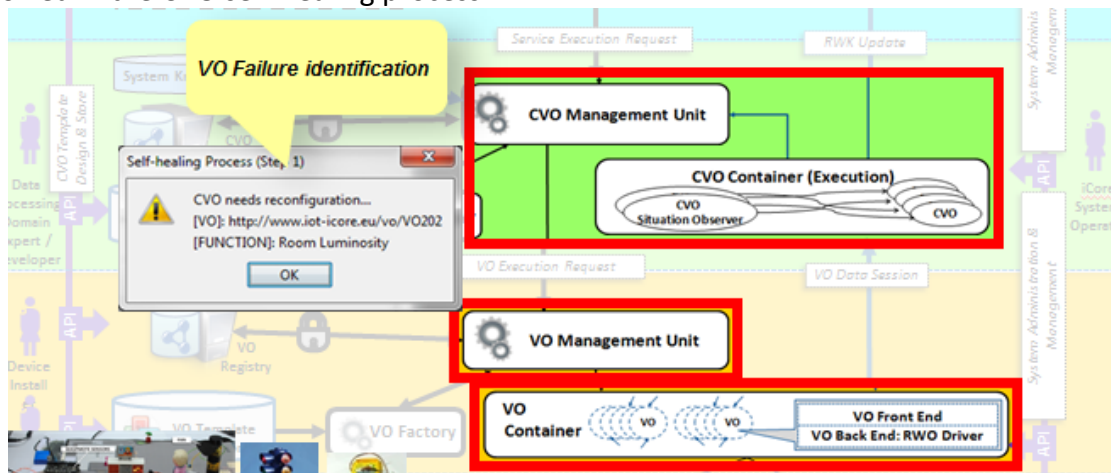


Figure 64: VO Failure identification

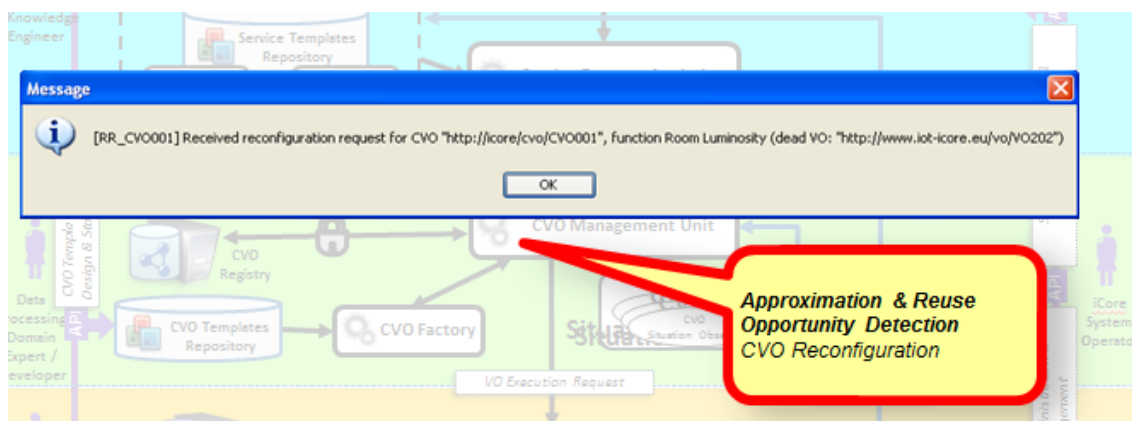


Figure 65: Triggering of reconfiguration process

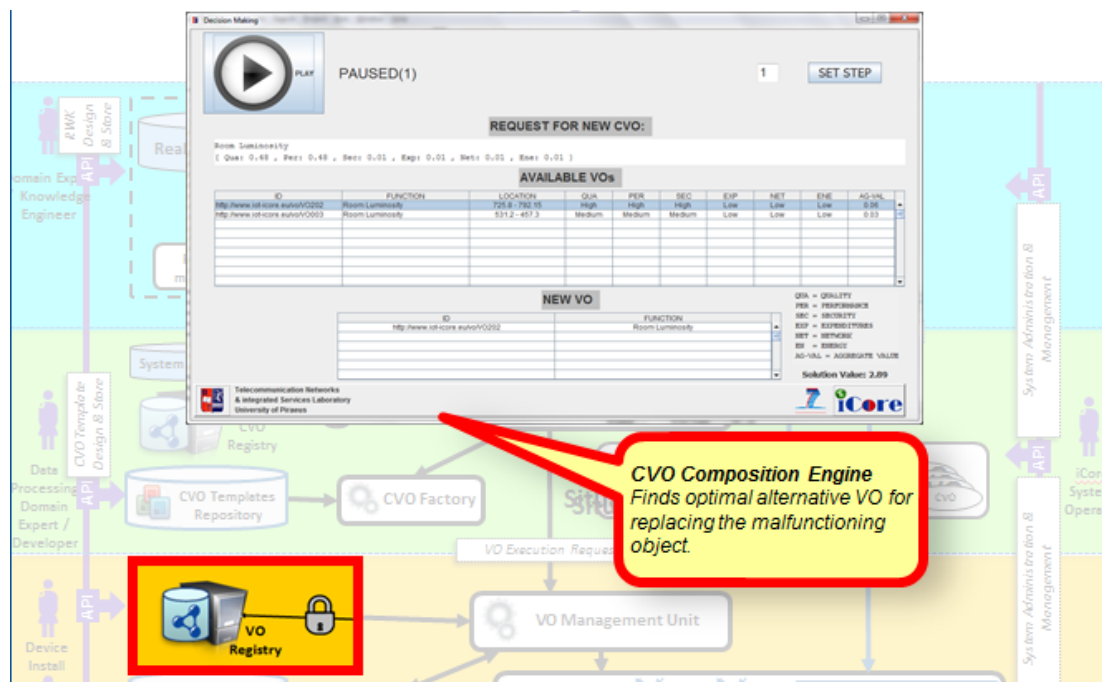


Figure 66: Replacement of problematic VO

Figure 67 depicts the evolution of the time required for the CVO self-healing process as the number of available VOs (to be considered as “replacements” for the problematic VO) increases.

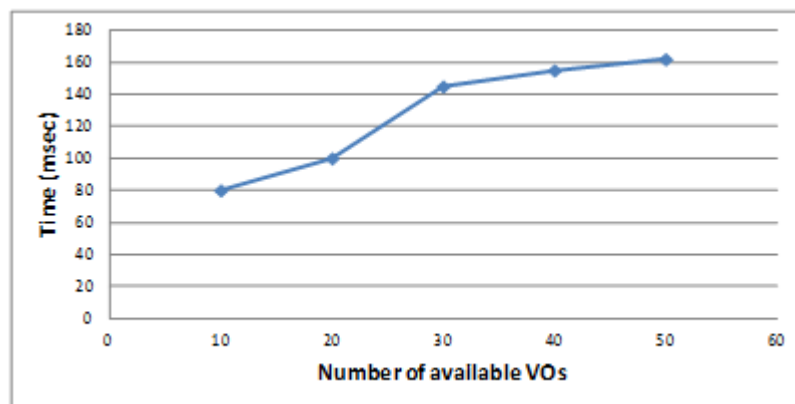


Figure 67: Time required for the CVO Self-healing process

### 3.1.3. Test case #3 – Knowledge-based CVO creation

#### 3.1.3.1. Description

The knowledge-based CVO instantiation takes place when an existing CVO can be directly deployed.

#### 3.1.3.2. Entities involved

The entities involved at the service level are: Service Requestor GUI, Service Template Repository, User Profiling and Preferences, Natural Language Processing and Service Request Analysis. The components at the CVO level include: CVO Registry, CVO Factory, CVO Management Unit: Approximation and Re-use Opportunity Detection, CVO Management

Unit: CVO Composition Engine, and CVO Management Unit: CVO Dynamic Workflow (Planner). Finally at the VO level the following components are used: VO Registry, VO Management Unit, VO Container, VO Front-End Software and Back-End REST Modules.

### 3.1.3.3. Related requirements

Table 2 lists the functional requirements (according to deliverable D2.1) related to this test case.

**Table 4: Requirements related to Smart Home test case #3**

Identifier	Name	Description
SL-F-1	The iCore platform requires a service model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-2	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-5	iCore external actors shall be able to instantiate new logic and reuse existing logic for dynamic iCore service composition by means of newly generated and existing iCore objects.	This is the systematic 'programmability' of an environment of iCore objects.
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-12	The iCore system shall be able to <b>aggregate</b> and infer relevant situational dimensions (e.g.	See above



	time, place, preference of users) to describe the current situation and anticipate changes to it.	
SL-F-13	The iCore system shall be able to <b>infer</b> (reason) on user intentions/behaviour in order to support current and future services and service requests.	Importance of storing relationships between objects, people, applications. The iCore system must be adaptable to User intentions/behaviour for dynamic service composition.
SL-F-15	The iCore system shall maintain and be able to store, classify, rank, and annotate a categorisation of different situations based on user intentions/behaviour per situation for future use.	It is an essential element for the iCore system to 'grow' intelligence and foster reuse of iCore objects in multiple services and across multiple situations.
SL-F-17	The iCore system shall be able to identify an approximate CVO quickly to match requested services as best as possible.	It is not always possible to find exact services for a user all the time. Thus, if an exact CVO which was supposed to provide a particular service is not available, we have to identify an approximate service. The approximation is done based on the availability of alternatives and possibilities. Depending on the situation, there might be a threshold to decide and select an approximate service (i.e. alternative service).
CVO-F-2	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters).

CVO-F-3	The CVO functionality shall be able to autonomously select and integrate the correct VOs that describe and match the current / potential service requirements.	A CVO must be able to evaluate the service function offered by VOs and reconfigure itself if necessary by selecting / maintaining the best VOs.
CVO-F-4	The CVO registry shall actively query information on available VOs and CVOs.	The CVO registry must be able to dynamically retrieve information about available VOs and CVOs.
CVO-F-5	The CVO registry shall be able to obtain Service Logic requirements either by a query (when they are stored somewhere) or by a trigger event (when they are constructed on the fly).	The CVO registry must be able to interface with the upper service level and be able to receive service logic requirements (request parameters).
CVO-F-6	The CVO registry shall be able to address and name stored VOs and CVOs.	Stored VOs and CVOs must have an address and name in order to be retrieved, e.g. as web services.
CVO-F-7	The CVO registry shall be able to store VO interaction history based on the user situation description framework or Category.	A CVO must store information about the CVO components and correlate them with the situation.
CVO-F-8	The CVO registry shall be able to maintain a set of past relationships	The CVO registry must keep a history of activated CVOs and their components.

	with other CVO's or VO's in order to create an environment/CVO space for Service composition.	
VO-F-8	A VO registry shall notify a CVO Registry of new VOs that become available for service composition and re-use.	The VO discovery interface between VO and CVO levels. A VO has the capability to advertise/notify itself for CVO composition.
VO-F-9	A VO shall notify CVO registries of its current status, availability and reliability for re-use.	The VO discovery interface between the VO and CVO levels. A VO shall have capability to update/modify itself and then notify the CVO for composition.

#### 3.1.3.4. Pre-conditions

Service templates, CVO templates and VO templates must have been defined. VOs and related Real World Objects need to be installed.

#### 3.1.3.5. Execution steps

This process enables the reuse of an already existing CVO. The first part of the process is similar to the steps that are described previously in 3.1.1 for the dynamic CVO creation until the triggering of the AROD entity of the CVO Management Unit. At this point the service request and situation information can be compared with records in the CVO registry for an adequate match. Past records corresponding to CVO components (VOs) with functions that are unavailable in the current situation (either exact or approximate ones) are filtered out, as they cannot fulfil the service goals. For the remaining records a satisfaction-rate similarity metric is calculated. This similarity metric depends on the quantity of total requested functions which are available as well as their correlations. It is calculated as a score (i.e. sum) of these correlations between the set of the requested and the required CVO functions. For the calculation of the overall similarity metric, the rest of the situation and request parameters are taken into account as well, such as requested policies, time of request, area of interest, available VOs, as well. If the overall similarity metric for an existing CVO equals or exceeds a specific threshold, then this existing CVO can be considered as suitable for a forthcoming request. Essentially, as already mentioned CVO level components can exploit known solutions as a response to a service request. In this way, the time needed for handling of a Service Execution Requests from the Service Level is reduced, since the CVO is not created from scratch.

### 3.1.3.6. Success criteria/ Expected output

A CVO is instantiated based on knowledge existing in the CVO registry. The creation/instantiation process is faster than in the case of the dynamic CVO creation. The involved mechanisms perform well in terms of producing the expected outcome as well as in terms of time required to fulfil their operation.

### 3.1.3.7. Test Case Result

Figure 68- Figure 70 provide a high-level view of the operation of the main components involved in the knowledge-based CVO creation.

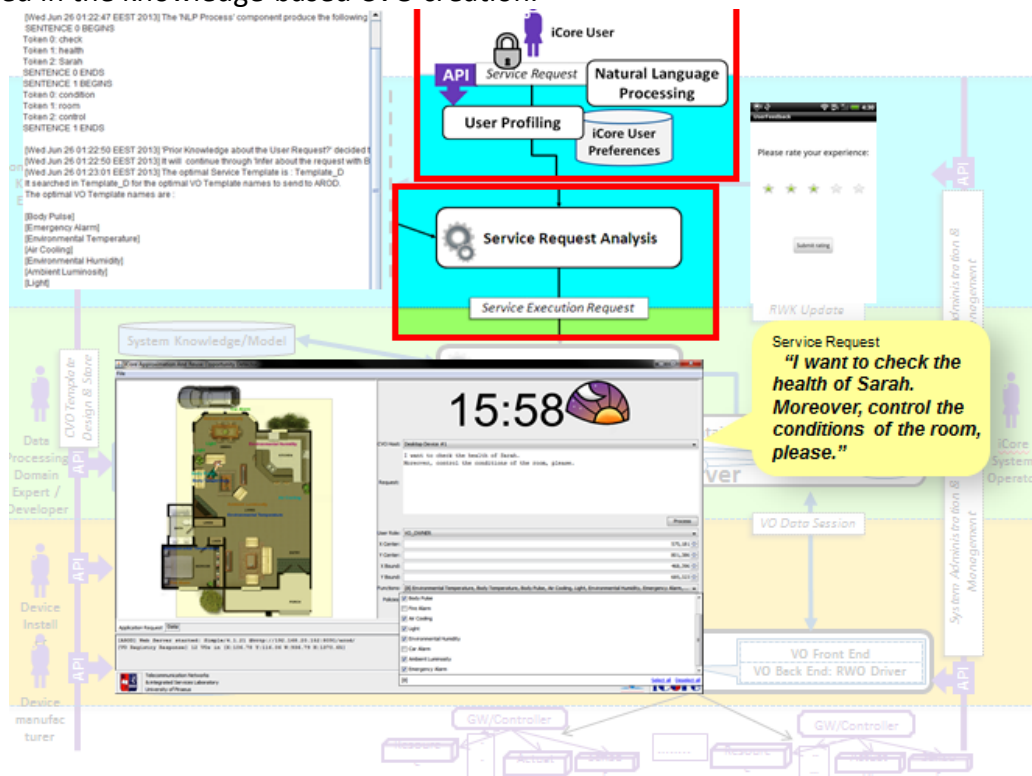


Figure 68: Service request triggers dynamic CVO creation process

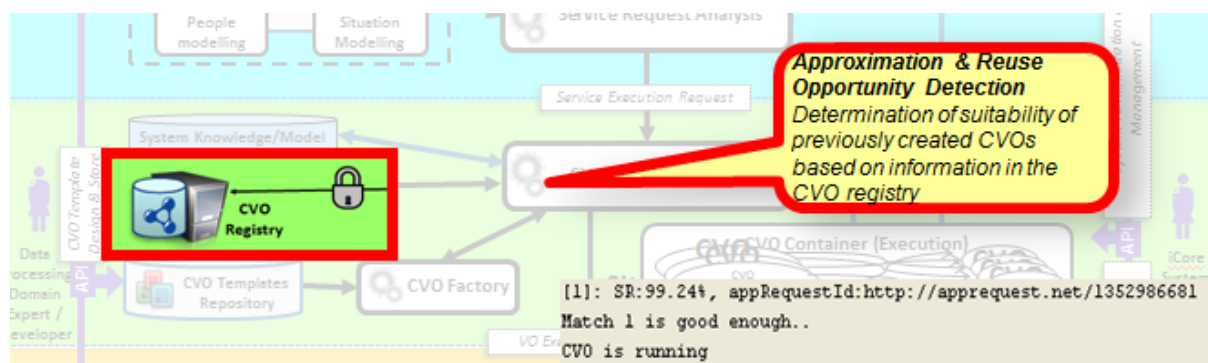
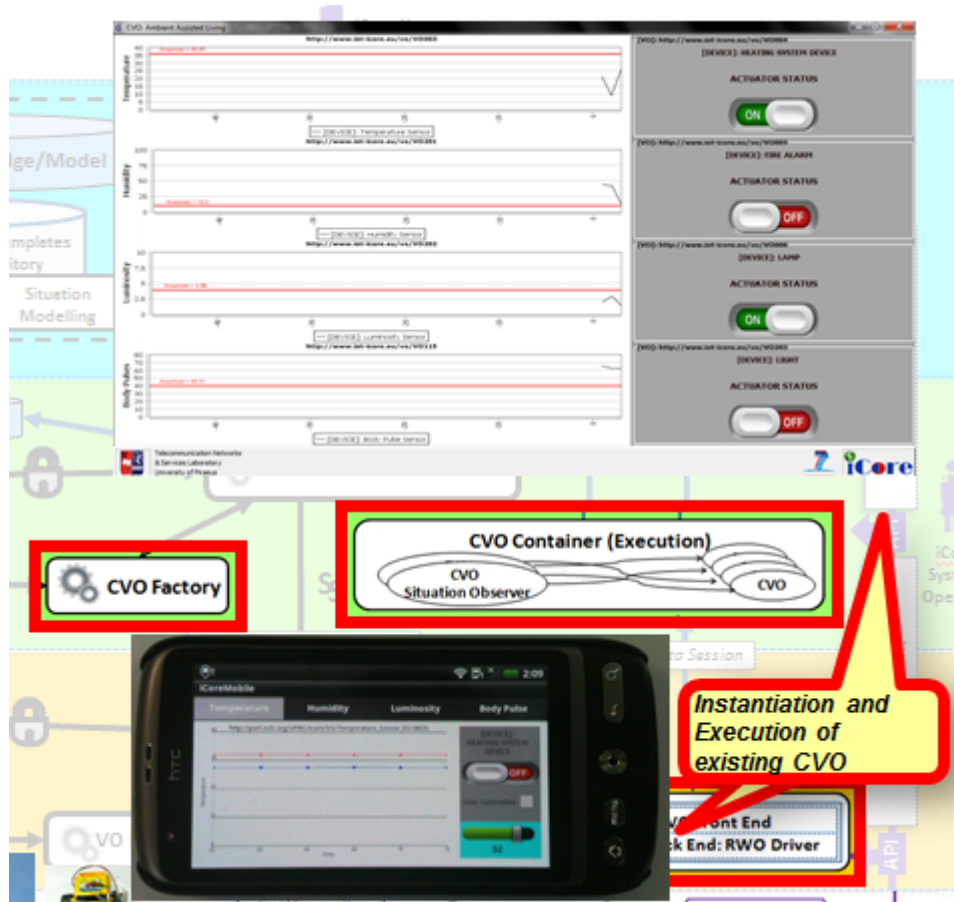
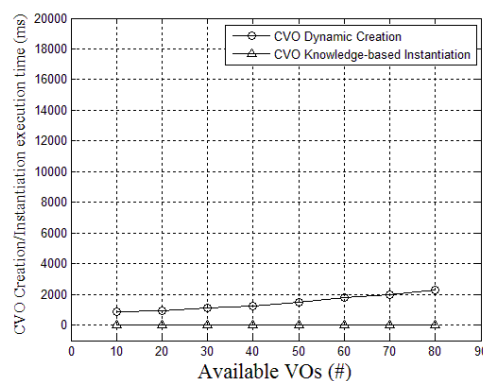


Figure 69: Search for an existing CVO that can fulfil the request



**Figure 70: CVO instantiation and execution**

Figure 71 presents the overall time, which is needed for the creation of a (new) CVO and the knowledge-based instantiation of an existing (previously recorded) CVO. The time for the creation of a new CVO (“from scratch”) comprises the time for; (i) the AROD, (ii) the CCE and (iii) the VO discovery. The time for the knowledge-based instantiation of an existing CVO on the other hand corresponds to the time required for the AROD. The time for the knowledge-based instantiation of an existing CVO remains much lower than the time required for the more complex process of a new CVO.



**Figure 71: Overall CVO Creation/Instantiation execution time vs. Number of Available VOs**

### 3.1.4. Test case #4 – CVO Coordination

#### 3.1.4.1. Description

CVO Coordination enables automated dynamic coordination of the simultaneous use of CVOs.

#### 3.1.4.2. Entities involved

The entities involved are the CVO Registry, CVO Management Unit: Coordination and various CVOs (Smart Home, Smart Car and Smart Traffic Light).

#### 3.1.4.3. Related requirements

Table 2 lists the functional requirements (according to deliverable D2.1) related to this test case.

Table 5: Requirements related to Smart Home test case #4

Identifier	Name	Description
SL-F-1	The iCore platform requires a service model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-2	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-12	The iCore system shall be able to <b>aggregate</b> and infer relevant situational dimensions (e.g. time, place,	See above

	preference of users) to describe the current situation and anticipate changes to it.	
CVO-F-4	The CVO registry shall actively query information on available VOs and CVOs.	The CVO registry must be able to dynamically retrieve information about available VOs and CVOs.
CVO-F-6	The CVO registry shall be able to address and name stored VOs and CVOs.	Stored VOs and CVOs must have an address and name in order to be retrieved, e.g. as web services.
CVO-F-7	The CVO registry shall be able to store VO interaction history based on the user situation description framework or Category.	A CVO must store information about the CVO components and correlate them with the situation.
CVO-F-8	The CVO registry shall be able to maintain a set of past relationships with other CVO's or VO's in order to create an environment/CVO space for Service composition.	The CVO registry must keep a history of activated CVOs and their components.
CVO-F-12	A CVO shall be validated before the execution, based on the received, complete and accepted non-conflict situation	Validation is necessary in order to add in valid CVOs against SLA rules (we need to insure that the decomposition of a SLA will result in a potential CVO discovery) and to insure that CVO execution will not generate bad results (e.g. circular loops or system crashes).



	with a clear lifespan for execution.	
CVO-F-13	A CVO shall be validated during the execution when possible as required by the Application.	This requirement will be possible when a change in SLA is sensed, validated and the described situation for the currently executed CVO to be replaced/corrected <b>OR</b> , when the execution part of iCore fails at a certain stage during the lifespan of the current situation and demands for a repair/replace action of the whole or parts of the CVO including possible service level communication. Since Service Level is only allowed to manage the SLA, this requirement can impose transactional capabilities.
VO-F-5	The VO functionality shall be able to dynamically describe RWOs and match (tag) the correct RWOs to current CVO / Service / Application requirements.	New RWOs can be added to the iCore system during run-time. Once VO functionalities tag with new RWOs then it also becomes a new VO and the old VO may survive for a certain time.
VO-F-8	A VO registry shall notify a CVO Registry of new VO's that become available for service composition and re-use.	The VO discovery interface between VO and CVO levels. A VO has the capability to advertise/notify itself for CVO composition.
VO-F-9	A VO shall notify CVO registries of its current status, availability and reliability for re-use.	The VO discovery interface between the VO and CVO levels. A VO shall have capability to update/modify itself and then notify the CVO for composition.

#### 3.1.4.4. Pre-conditions

Service templates, CVO templates and VO templates must have been defined. Multiple CVOs using the same CVO/VO are running simultaneously.

### 3.1.4.5. Execution steps

As already introduced, this process enables dynamic management of the simultaneous access to and use of available CVOs from other entities, such as other CVOs. Specifically, the CVO Coordination entity observes the use of available running CVOs by other entities and based on specific priority indicators combined with the current situation, decides on the sequence for the use of the CVO from each user-entity. Consequently, when more than one entity simultaneously using/accessing the same CVO, this is monitored by the CVO Coordination entity of the CVO Management Unit. The CVO coordination entity collects the available information with respect to the current situation, as well as the priority indicators for each entity, aiming to derive a priority list for the CVO use. In this way the best possible management of the concurrently running, inter-related CVOs is achieved, in terms of the avoidance of potential conflicts arising from their simultaneous access and use by other entities.

### 3.1.4.6. Success criteria/ Expected output

Resolving conflicting commands/request on one CVO/VO from multiple CVOs.

### 3.1.4.7. Test Case Result

Figure 72 provides a high-level view of the CVO coordination.



Figure 72: CVO Coordination overview

## 3.2 Smart Office: Easy meeting

### 3.2.1. Test case #1 – Smart Meeting Organisation

#### 3.2.1.1. Description

When the iCore user launches the application on the smart mobile device, it is connected to the MQTT broker and can organise and create a new meeting. Meeting participants are informed via automated personalised emails.

#### 3.2.1.2. Entities involved

CVO Container, Real World Knowledge, Service Request Front End

#### 3.2.1.3. Related requirements

Table 6: Requirements related to Smart Office test case #1

Identifier	Name	Description
SL-F-2.	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-11.	Situations and user properties shall be Appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.

#### 3.2.1.4. Pre-conditions

- The smart mobile device is on.
- CVO container is up and running, waiting for input.
- Database containing RWK/RWI is accessible.

#### 3.2.1.5. Execution steps

- Access the Smart Organiser new meeting application
- Select “Create New Meeting”
- Fill the meeting details
- Add all the participants
- Select the “Save” button
- Wait for the system to check the input data.

- Wait for the system to generate the QR-Codes
- Wait for the system to generate and send the personalised emails

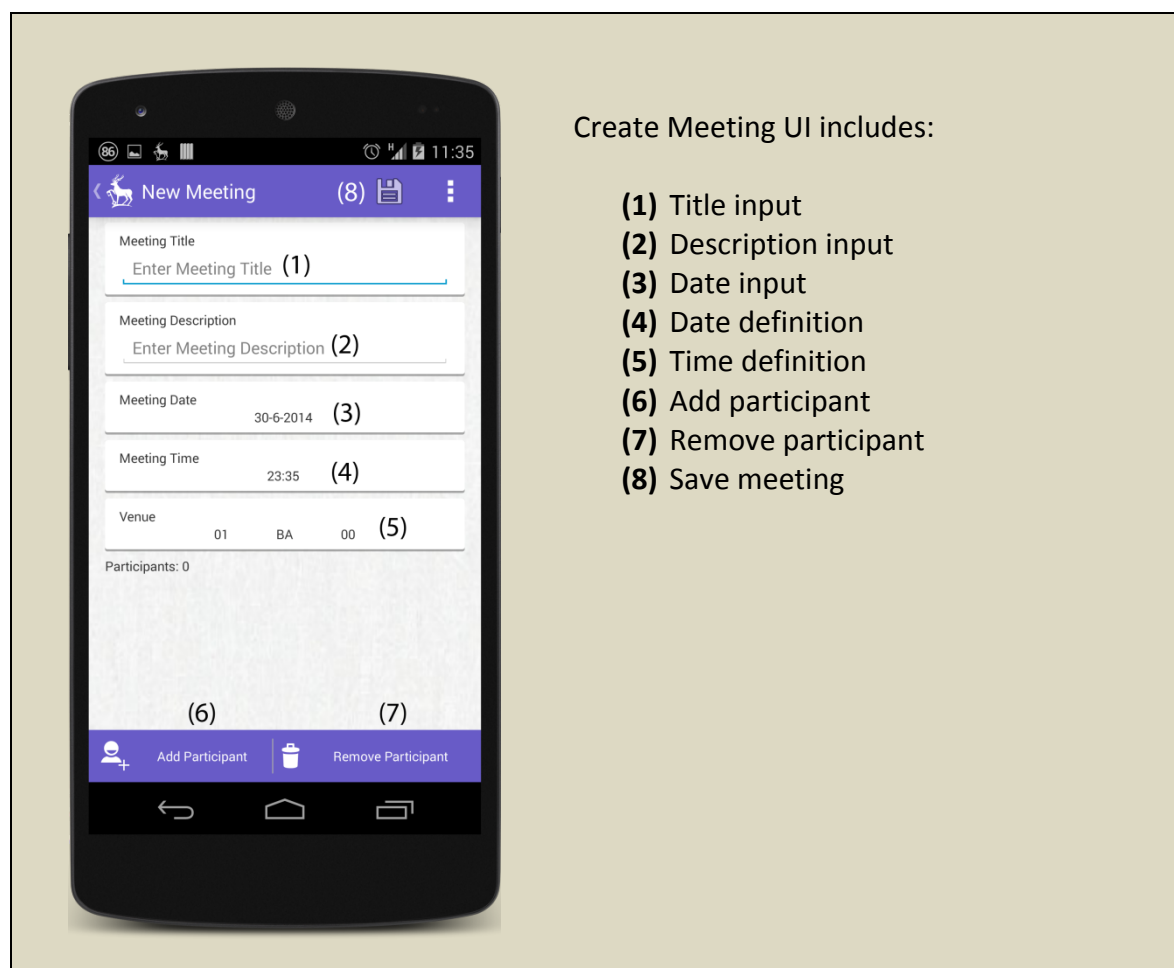
### 3.2.1.6. Success criteria/ Expected output

The system displays a success message only when the incoming data concerning meeting venue and organiser are matched to the RWK. Furthermore, additional feedback is presented when the QR codes are generated and when the emails are sent to the visitors. In case one of the previous messages do not appear in the system output it is because one of the following reasons:

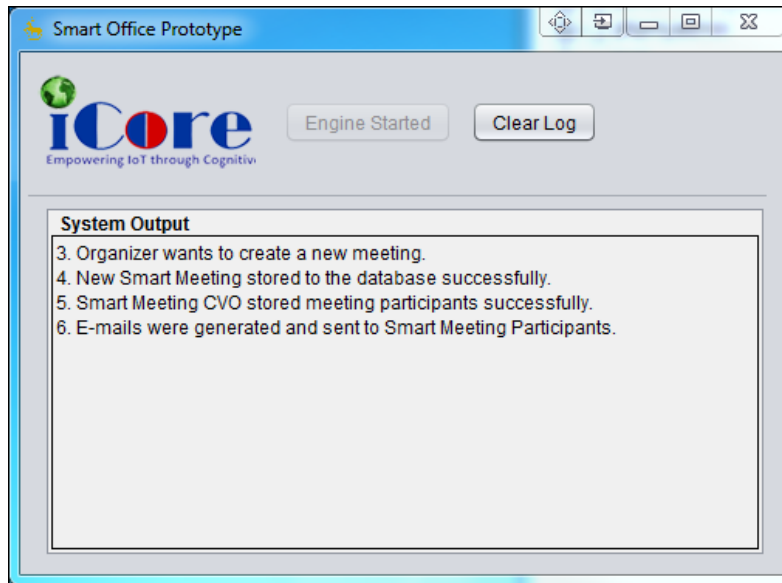
- Meeting organiser is not logged in to the app
- Meeting organiser is no longer eligible to create a smart meeting
- Meeting venue provided is not existing in the RKW
- Attendant's email is not in the proper format

### 3.2.1.7. Test Case Result

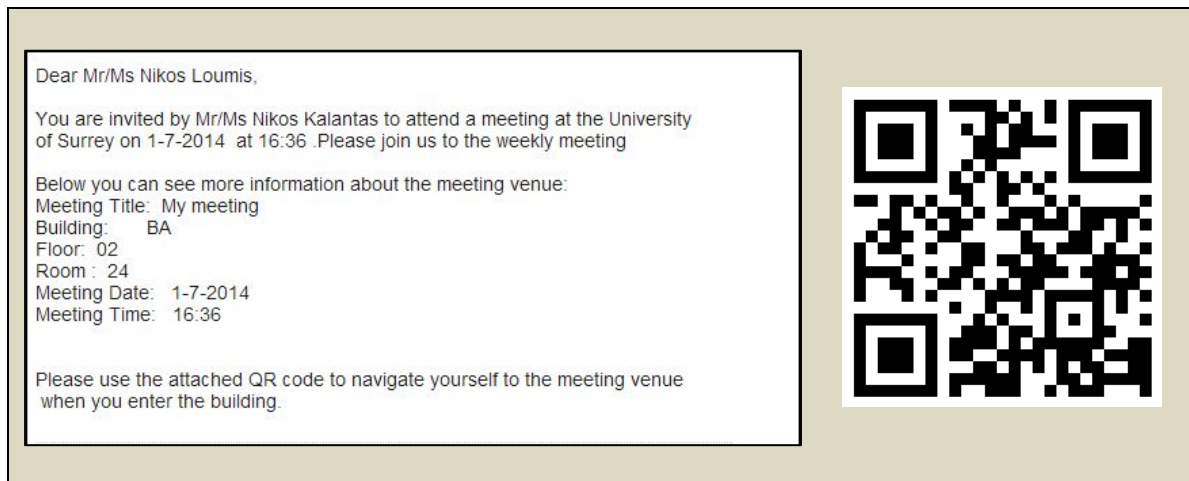
The following figures, taken from a live execution of the test case, illustrate the result: Successful new meeting creation and storage, as well as, generated emails.



**Figure 73: Create meeting UI**



**Figure 74: New meeting CVO notification.**



**Figure 75: New meeting generated email.**

### 3.2.2. Test case #2 – Smart Meeting Guidance

#### 3.2.2.1. Description

When the iCore Smart Meeting Visitor enters the host building, he/she comes across a mounted smart device. Guidance instructions can be retrieved by scanning the aforementioned QR-Code received with the invitation email.

#### 3.2.2.2. Entities involved

Real World Knowledge, CVO Container, VO Container, VO Management Unit, VO Registry

### 3.2.2.3. Related requirements

Table 7: Requirements related to Smart Office test case #2

Identifier	Name	Description
SL-F-2.	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-11.	Situations and user properties shall be Appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
CVO-F-1.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-9.	The CVO shall be enabled with an auto construction of a “scheduler”.	A CVO must somehow evolve a schedule to use each VO (and its functionalities) in a sequence (some in parallel) in order to achieve a required service.
VO-F-1.	The VO Registry shall be able to maintain and record the links with each RWO (i.e. send notification (signalling) when coupled and when decoupled).	Handle to the real world is maintained in a best possible way. Awareness regarding the RWO.
VO-F-2.	The VO shall be able to name, type	RWOs are appropriately named and addressed within their respective proximity domains.

	(Metadata) and address each RWOs that becomes available (i.e. at the point of discovery).	
VO-F-6.	A VO shall describe the Available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
VO-F-7.	A VO shall be able to know the current location (i.e. Be location aware) of the RWO it is representing.	The location of the RWO shall be abstracted at the VO level.

#### 3.2.2.4. Pre-conditions

- The mounted smart guidance device is on.
- CVO container is up and running, waiting for input.
- Database containing RWK/RWI is accessible.
- Visitor has the QR-code handy

#### 3.2.2.5. Execution steps

- Select “Visitor Check-In” button at the Smart Meeting Guidance mobile application
- Scan the iCore compliant QR-Code using the front camera of the mounted device
- Wait for the system to check the QR- Code
- Wait for the system to match the data with the RWK
- Information message is displayed in the screen
- Select the “guide me there” button, on the dialog screen
- Follow the directions to the next hop

#### 3.2.2.6. Success criteria/ Expected output

During the Smart Meeting Guidance scenario, multiple tests are run in the back end of the system. Various things might go wrong leading to the equivalent message display.



On the other hand, if the QR-Code scanning is successful, the iCore visitor is able to see the information messages, as well as the directions to the meeting venue.

- The reasons why the Smart Meeting Guidance might not be completed are:
- Scanned QR-Code is not iCore compliant
- iCore QR-Code contains meeting not stored in the database

### 3.2.2.7. Test Case Result

This section provides screenshots captured in a real life scenario at the University of Surrey. Successful scanning, matching, and guidance to the meeting venue.



Figure 76: QR-Code scanning.

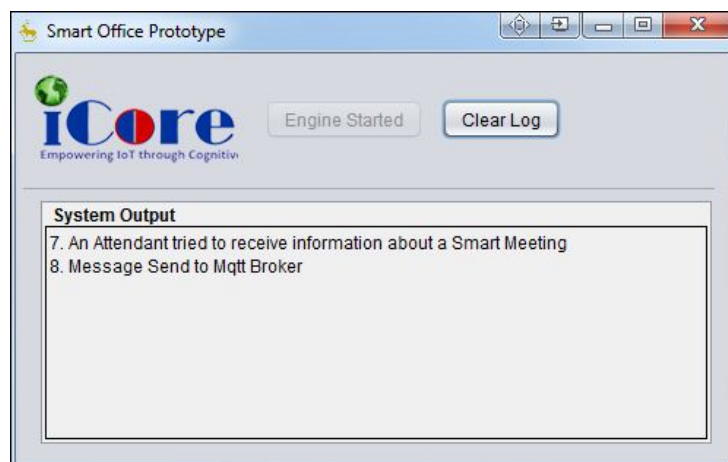
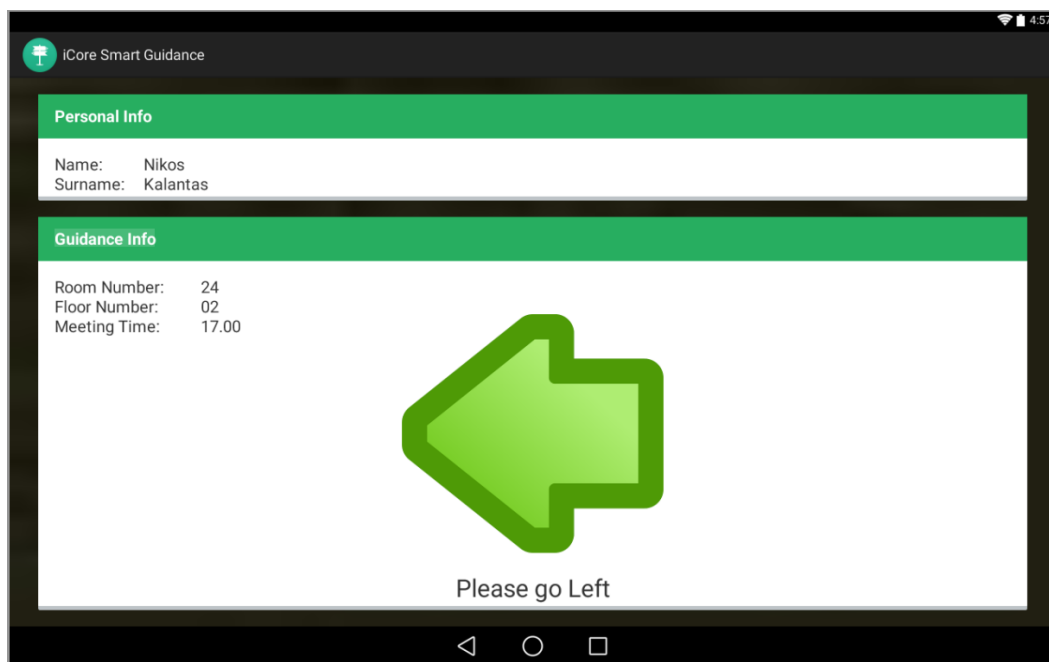


Figure 77: System Logging.



**Figure 78: Indoor Navigation.**

### 3.2.3. Test case #3 – Smart Meeting Break

#### 3.2.3.1. Description

Once the meeting organiser wants to check for a break possibility, he either triggers the “auto smart break meeting”, or the “manual smart break”. Both of the features share the same tangible target which is the calculation of the possibility having a break at any given time. The outcome is pushed to the Meeting organiser display VO.

#### 3.2.3.2. Entities involved

CVO Container, VO container, Service Request Front end

#### 3.2.3.3. Related requirements

**Table 8: Requirements related to Smart Office test case #3**

Identifier	Name	Description
SL-F-2.	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.

SL-F-11.	Situations and user properties shall be Appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
CVO-F-1.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-9.	The CVO shall be enabled with an auto construction of a “scheduler”.	A CVO must somehow evolve a schedule to use each VO (and its functionalities) in a sequence (some in parallel) in order to achieve a required service.
VO-F-2.	The VO shall be able to name, type (Metadata) and address each RWOs that becomes available (i.e. at the point of discovery).	RWOs are appropriately named and addressed within their respective proximity domains.
VO-F-6.	A VO shall describe the Available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
VO-F-7.	A VO shall be able to know the current location (i.e. Be location aware) of the RWO it is	The location of the RWO shall be abstracted at the VO level.

representing.

### 3.2.3.4. Pre-conditions

- CVO container is up and running, waiting for input.
- Meeting organiser Mobile Application is installed and running
- Meeting attendants mobile application is installed and running
- Esper Engine is up and running

### 3.2.3.5. Execution steps

#### *Automated*

- Turn the “auto break” switch on, at the mobile organiser application UI
- Wait for the system to gather sensor data
- Wait for the system to calculate the gathered data
- Information message is displayed in the notification area
- Break prediction is illustrated in the appropriate graph at the organiser’s mobile application UI

#### *Manual*

- Select the “Smart Break” button at the mobile organiser application UI
- Wait for the meeting participants to express their opinion via their mobile application
- Wait for the system to gather all the data
- Wait for the system to calculate the gathered data
- Information message is displayed in the notification area
- Break prediction is illustrated in the organiser’s mobile application UI

### 3.2.3.6. Success criteria/ Expected output

The Break information is shown to the meeting organiser by his mobile device’s display in a predefined amount of time. This amount of time is configurable and dynamic.

The Smart Break function suggests the following:

- The chance of having a break at any given time
- The estimated time remaining to the next needed break
- The calculated attendant opinion on the manual smart break

### 3.2.3.7. Test Case Result

The following screenshots are captured at a live execution of the test case. In both of the modes offered by the system, the reader can clearly see the outcome of the data processing.

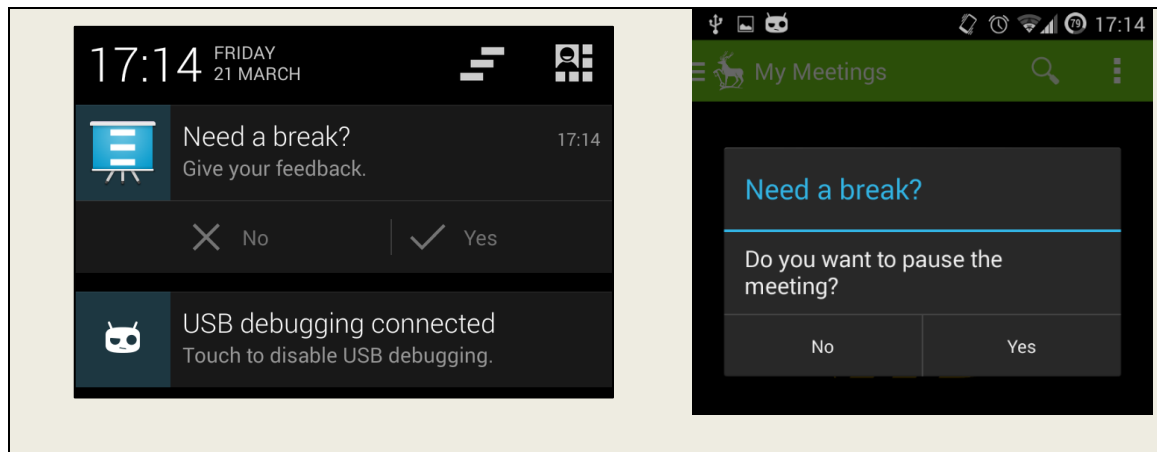


Figure 79: Meeting Organiser Feedback Provider

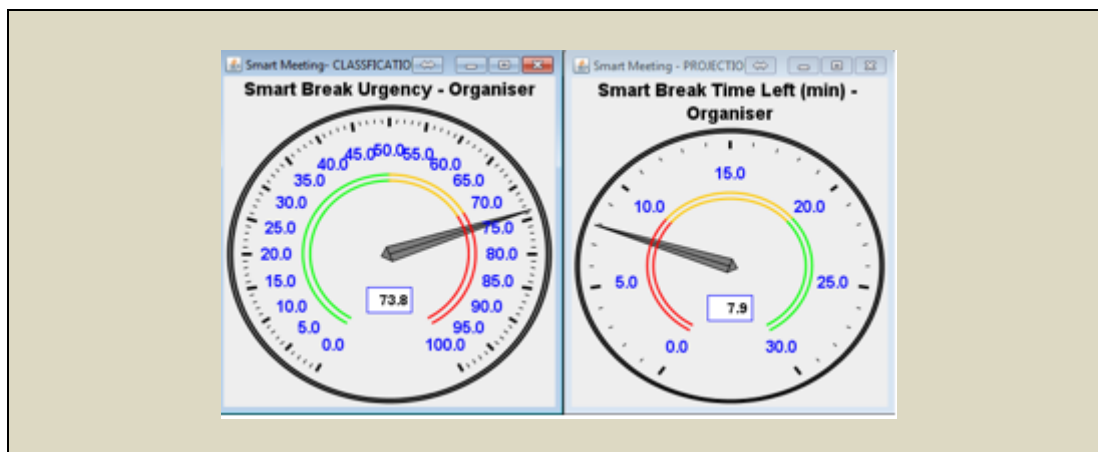


Figure 80: Smart Break Outcome

### 3.2.4. Test case #4 – Smart Meeting Recording

#### 3.2.4.1. Description

When the iCore Smart Meeting Organiser wants to record the meeting using the embedded microphones on the meeting participants' devices. The device to handle the recording is the best match to the criteria defined by the meeting organiser, at any given time.

#### 3.2.4.2. Entities involved

CVO Container, Service Request Front end, VO container, Service Request Analysis, Service Template Repository, Real World Knowledge, CVO Management Unit, CVO Registry, VO Registry

#### 3.2.4.3. Related requirements

Table 9: Requirements related to Smart Office test case #4

Identifier	Name	Description
------------	------	-------------

SL-F-2.	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-9.	The iCore system shall enable the proper and Efficient composition of iCore objects that fulfil the query from the service level.	The iCore system shall be able to configure/monitor/select iCore objects on the basis of what the services requested by external actors need.
SL-F-11.	Situations and user properties shall be Appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
CVO-F-1.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-4.	The CVO registry shall actively query information on available VOs and CVOs.	The CVO registry must be able to dynamically retrieve information about available VOs and CVOs.
CVO-F-5.	The CVO registry shall be able to obtain Service Logic requirements either by a query (when they are	The CVO registry must be able to interface with the upper service level and be able to receive service logic requirements (request parameters).

	stored somewhere) or by a trigger event (when they are constructed on the fly).	
CVO-F-6.	The CVO registry shall be able to address and name stored VOs and CVOs.	Stored VOs and CVOs must have an address and name in order to be retrieved, e.g. as web services.
CVO-F-9.	The CVO shall be enabled with an auto construction of a “scheduler”.	A CVO must somehow evolve a schedule to use each VO (and its functionalities) in a sequence (some in parallel) in order to achieve a required service.
VO-F-2.	The VO shall be able to name, type (Metadata) and address each RWOs that becomes available (i.e. at the point of discovery).	RWOs are appropriately named and addressed within their respective proximity domains.
VO-F-6.	A VO shall describe the Available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
VO-F-7.	A VO shall be able to know the current location (i.e. Be location aware) of the RWO it is representing.	The location of the RWO shall be abstracted at the VO level.



#### 3.2.4.4. Pre-conditions

- CVO container is up and running, waiting for input.
- Meeting organiser Mobile Application is installed and running
- Meeting attendants mobile application is installed and running

#### 3.2.4.5. Execution steps

- Select the “Smart Recording” button at the mobile organiser application UI
- Wait for the system to gather all sensor data from the in-meeting devices
- Wait for the system to calculate the gathered data
- Select the most appropriate device matching the given criteria
- Information message is displayed in the recording device

#### 3.2.4.6. Success criteria/ Expected output

The Recording information is shown to the meeting participant when his device is recording. All the recording fractures are stored locally in the host device.

#### 3.2.4.7. Test Case Result

After the data are gathered and processed, the recording notification can be displayed in the notification area of the attendant’s mobile device.

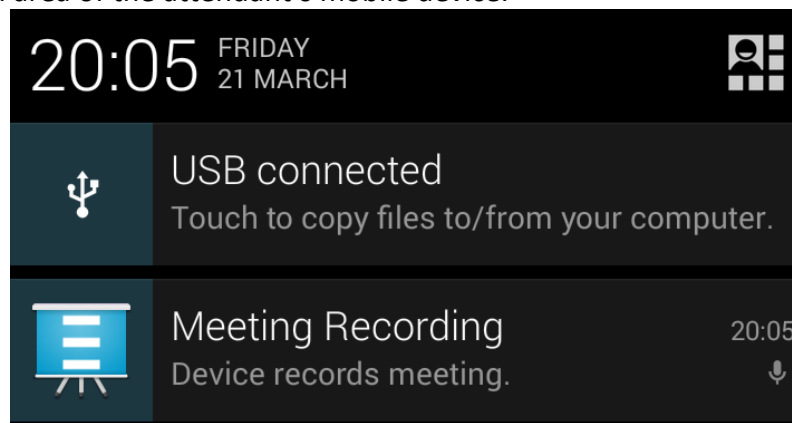


Figure 81: Meeting recording notification

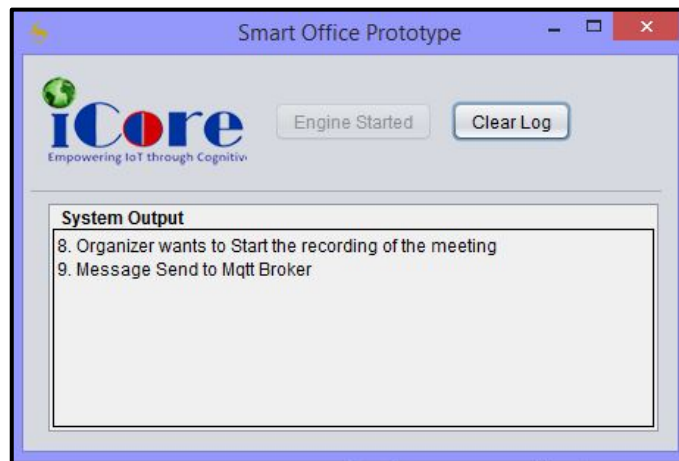


Figure 82: Smart Meeting Recording Service Request at Smart Office CVO

### 3.2.5. Test case #5 – Smart Meeting Wrap-Up

#### 3.2.5.1. Description

At the end of the Smart Meeting, the organiser decides to wrap-up. All the data generated throughout the meeting all gathered and uploaded to a remote location. Furthermore, the locally stored files are deleted from the mobiles devices.

#### 3.2.5.2. Entities involved

CVO Container, Service Request Front end, VO container

#### 3.2.5.3. Related requirements

Table 10: Requirements related to Smart Office test case #5

Identifier	Name	Description
SL-F-2.	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-11.	Situations and user properties shall be Appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.

CVO-F-1.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-9.	The CVO shall be enabled with an auto construction of a “scheduler”.	A CVO must somehow evolve a schedule to use each VO (and its functionalities) in a sequence (some in parallel) in order to achieve a required service.
VO-F-2.	The VO shall be able to name, type (Metadata) and address each RWOs that becomes available (i.e. at the point of discovery).	RWOs are appropriately named and addressed within their respective proximity domains.
VO-F-6.	A VO shall describe the Available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
VO-F-7.	A VO shall be able to know the current location (i.e. Be location aware) of the RWO it is representing.	The location of the RWO shall be abstracted at the VO level.

### 3.2.5.4. Pre-conditions

- CVO container is up and running, waiting for input.
- Meeting organiser Mobile Application is installed and running
- Meeting attendants mobile application is installed and running

### 3.2.5.5. Execution steps

- Select the “Smart Wrap-Up” button at the mobile organiser application UI
- Wait for the system to contact the meeting devices
- Wait for the devices to encapsulate the recordings and upload them
- Wait for the system to receive the data and store the recordings

### 3.2.5.6. Success criteria/ Expected output

The test is considered successful when all the data generated throughout the meeting are uploaded in a remote location. Data integrity is needed in order for the recordings to be played with no interruptions. Furthermore, the files need to be deleted from all the devices for privacy purposes.

### 3.2.5.7. Test Case Result

Figure 83 shows the system log when the wrap-up trigger message is received. and the information of the recordings uploaded while Figure 84 shows the notification on the participant device notifying of the ongoing operation

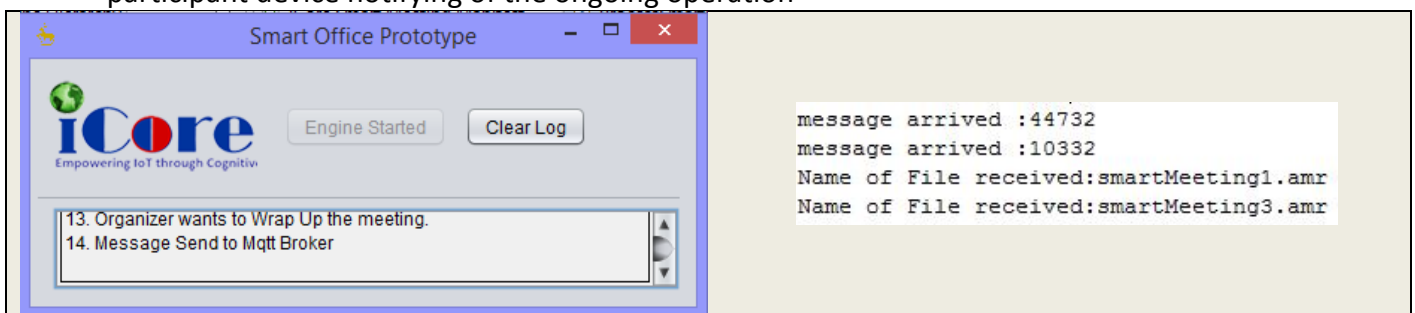


Figure 83: Smart Meeting Wrap-up CVO Handling Meeting Recordings.

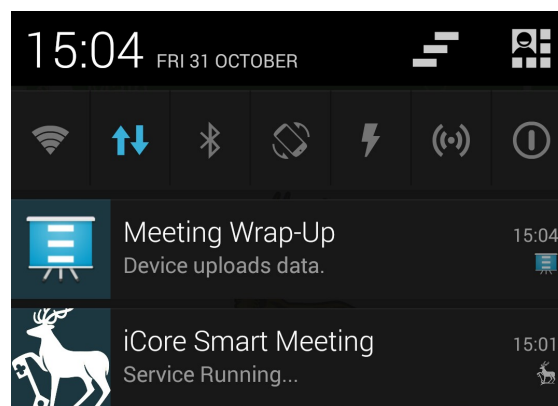


Figure 84: Meeting wrap-up notification.

## 3.3 Smart City: Transportation

### 3.3.1. Test case #1 – The iCore Services are available to the OnBoard Unit

#### 3.3.1.1. Description

When the iCore client application is launched on the Uconnect™ device, the connection with the VOs is established and the available iCore Smart City Transportation Services “Parking Around Me” and “Parking at Destination” are shown to the Driver.

#### 3.3.1.2. Entities involved

OnBoard Unit, Management Unit, CVO Container

#### 3.3.1.3. Related requirements

**Table 11: Requirements related to Smart City: transportation test case #1**

Identifier	Name	Description
SL-F-14	The iCore system may have an interface where users can explicitly describe or query his/her intentions/behaviour if necessary regarding his/her current situation (iCore is supposed to be able to sense those changes).	Simple API to communicate user intentions, semantic interrogation (SPARQL) for current situation and personal history. Feature is addressed to the cases where “observation” capability at Service Level can’t perform or direct information from the user when demanded.
SL-F-17	The iCore system shall be able to identify an approximate CVO quickly to match requested services as best as possible.	It is not always possible to find exact services for a user all the time. Thus, if an exact CVO which was supposed to provide a particular service is not available, we have to identify an approximate service. The approximation is done based on the availability of alternatives and possibilities. Depending on the situation, there might be a threshold to decide and select an approximate service (i.e. alternative service).
CVO-F-6	The CVO registry shall be able to address and name stored VOs and CVOs.	Stored VOs and CVOs must have an address and name in order to be retrieved, e.g. as web services.

CVO-F-7	The CVO registry shall be able to store VO interaction history based on the user situation description framework or Category.	A CVO must store information about the CVO components and correlate them with the situation.
CVO-F-8	The CVO registry shall be able to maintain a set of past relationships with other CVO's or VO's in order to create an environment/CVO space for Service composition.	The CVO registry must keep a history of activated CVOs and their components.

#### 3.3.1.4. Pre-conditions

The OnBoard Unit is on.

iCore Services “Parking Around Me” and “Parking at Destination” are both enabled in the “iCore Services” sub-menu of the iCore client application.

A connection between the OnBoard Unit and the VOs (WLAN Connection) is available.

#### 3.3.1.5. Execution steps

- Access the Uconnect™ Apps Menu
- Select (launch) the iCore client application
- Wait for the application to run and check if the “Parking Around Me” and “Parking at Destination” buttons are shown on the display

#### 3.3.1.6. Success criteria/ Expected output

Only if the iCore Services “Parking Around Me” and “Parking at Destination” are actually available on the On Board Unit, related buttons are visualized on the Uconnect™ HMI; thus, if such buttons appear in the application main page, iCore Services have been made available and the test can be considered executed with success.

If any of such buttons does not appear on the HMI, this might be due to one the following reasons:

- the associated iCore Service has been manually disabled by the User at client application level;
- the associated iCore Service is not available.

Initially, only the “Parking Around Me” can be executed due to the fact that a recurrent destination is not yet identified by the Real World Knowledge (User recurrent destinations).

### 3.3.1.7. Test Case Result

The following figures, taken from a live execution of the test case, show the test case result: the iCore Services activation buttons are shown on the OnBoard Unit. The Smart City Transportation Demonstrator Management Unit component initially sets the “Parking at Destination” state to disabled since the identified destination is not yet available from the RWK module.



Figure 85: iCore Services available on the Onboard Unit

### 3.3.2. Test case #2 –The Parking availability information is correctly and timely sent to the OnBoard Unit – case “Parking Around Me”

#### 3.3.2.1. Description

The test case aims at validating the correctness of the execution of the iCore Smart City Transportation Demonstrator, ensuring that the correct information is shown to the driver on the OnBoard Unit. After the activation of the “Parking Around Me” iCore Service, the correct information is pushed to the OnBoard Unit through the Car Display VO.

#### 3.3.2.2. Entities involved

OnBoard Unit, Car Display VO, Car GPS VO, VO Container, Smart City Transportation CVO, CVO Container, Turin Parking System VO, VO Registry

#### 3.3.2.3. Related requirements

Table 12: Requirements related to Smart City: transportation test case #2

Identifier	Name	Description
CVO-F-1	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.



CVO-F-2	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters).
CVO-F-3	The CVO functionality shall be able to autonomically select and integrate the correct VOs that describe and match the current / potential service requirements.	The CVO functionality shall be able to autonomically select and integrate the correct VOs that describe and match the current / potential service requirements.
CVO-F-4	The CVO registry shall actively query information on available VOs and CVOs.	The CVO registry must be able to dynamically retrieve information about available VOs and CVOs.
CVO-F-5	The CVO registry shall be able to obtain Service Logic requirements either by a query (when they are stored somewhere) or by a trigger event (when they are constructed on the fly).	The CVO registry must be able to interface with the upper service level and be able to receive service logic requirements (request parameters).
CVO-F-6	The CVO registry shall be able to address and name stored VOs and CVOs.	Stored VOs and CVOs must have an address and name in order to be retrieved, e.g. as web services.

VO-F-6	A VO should describe the available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe-threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
--------	---	---

#### 3.3.2.4. Pre-conditions

The OnBoard Unit is on.

A connection is available between the OnBoard Unit and the VOs (WLAN Connection)

The iCore client application is running and iCore Service “Parking Around Me” is enabled in the “iCore Services” sub-menu.

The Smart City Transportation (Smart Parking) “Parking around Me” CVO is executed (find a parking close to the last car position).

#### 3.3.2.5. Execution steps

- Activate the “Parking Around Me” iCore Smart Parking Service from the OnBoard Unit
- Wait for the parking information to be retrieved and shown to the Driver in a popup window
- Check if the information shown is correct

#### 3.3.2.6. Success criteria/ Expected output

- The Parking availability information is shown to the Driver by the OnBoard Unit in a reasonable time (less the 10 seconds)
- The Parking area suggested meets the following requirements:
  - it is the nearest to the vehicle’s current position among those made available by the Open Data Service in use
  - it has at least one free parking slot
- The information shown corresponds to what available in the Open Data Service in use

#### 3.3.2.7. Test Case Result

Figure 86, taken from a live execution of the test case, shows the test case result: the Parking availability information coming from the iCore Smart City Transportation Demonstrator is shown to the driver through the OnBoard Unit.



**Figure 86: Parking Availability Information shown on the Onboard Unit (Parking around Me)**

Several tests have been performed in different locations within and around the city of Turin. In all cases, the Parking area suggested was identified correctly among those made available by the Turin Open Data Service and the information shown on the display was consistent.

### 3.3.3. Test case #3 – The User Behaviour learning module is able to correctly recognize and store recurrent trips/destinations

#### 3.3.3.1. Description

Once the On Board Unit is activated (either automatically at the key on event or manually by the User), the RWK (Simodule detecting recurrent User destination (User Behaviour Adaptation) starts the elaboration of GPS localization signals coming from the Car GPS VO. When the User Behaviour learning module recognizes that a particular trip is performed recurrently, information about origin, destination and route is stored in the **Real World Knowledge** database and the trip is added to the list of recurrent trips.

#### 3.3.3.2. Entities involved

On Board Unit, Management Unit, CVO Execution, Recurrent Trip Learning

#### 3.3.3.3. Related requirements

**Table 13: Requirements related to Smart City: transportation test case #3**

Identifier	Name	Description
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe semantically not only iCore objects but also situations, and service request trends and actor intentions.

SL-F-12	The iCore system shall be able to aggregate and infer relevant situational dimensions (e.g. time, place, preference of users) to describe the current situation and anticipate changes to it.	The iCore system is envisioned to describe semantically not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-13	The iCore system shall be able to infer (reason) on user intentions/behaviour in order to support current and future services and service requests.	Importance of storing relationships between objects, people, applications. The iCore system must be adaptable to User intentions/behaviour for dynamic service composition.
SL-F-15	The iCore system shall maintain and be able to store, classify, rank, and annotate a categorisation of different situations based on user intentions/behaviour per situation for future use.	It is an essential element for the iCore system to 'grow' intelligence and foster reuse of iCore objects in multiple services and across multiple situations.

#### 3.3.3.4. Pre-conditions

The On Board Unit is on.

A connection is available between the On Board Unit and the VOs (WLAN Connection).

The Recurrent Trip Learning module is executed.

#### 3.3.3.5. Execution steps

- Perform multiple test drive sessions along the same routes
- Check in the Real World Knowledge database implementation if the corresponding trip has been added to recurrent trips list and the associated information (origin, destination, route) has been stored correctly

### 3.3.3.6. Success criteria/ Expected output

The test is considered performed with success if the trip is added to the database after a minimum number of driving sessions (details about specific criteria and algorithms cannot be disclosed because they contain information belonging to CRF background knowledge) and all the associated information stored in the database correspond to the actual trip.

### 3.3.3.7. Test Case Result

In order to make the test execution easier, a specific SW tool to access the RWK database and extract information was developed and a front-end (in form of a web portal) was released to create graphical maps upon the extracted information and display it on a web page. This allowed comparisons between the information stored in the RWK database and the corresponding expected results.

Tests were performed using 12 selected, predefined trips: 4 urban routes, 4 extra-urban routes and 4 trips with both urban and extra-urban routes.

In all cases, trips have been recognized correctly and related information stored in the RWK database.

Figure 87 shows a comparison between the route provided to CRF professional drivers for the execution of the test drive (left) and the graphical representation of the trip recognized by the User Behaviour learning module (right).

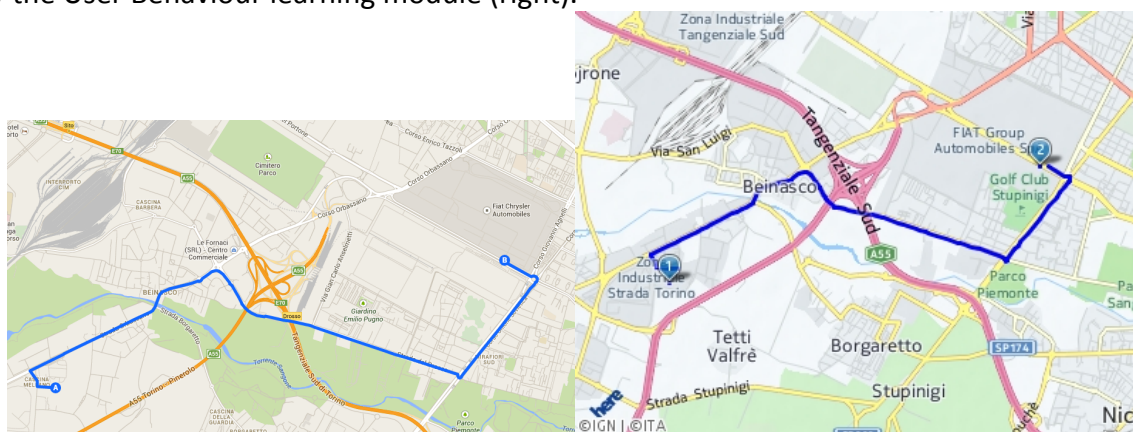


Figure 87: Comparison between actual and detected trips

### 3.3.4. Test case #4 – The User Behaviour prediction module is able to correctly infer recurrent trips

#### 3.3.4.1. Description

Once the On Board Unit is activated (either automatically at the key on event or manually by the User), the RWK (Simodule detecting recurrent User destination (User Behaviour Adaptation) starts the elaboration of GPS localization signals coming from the Car GPS VO. When the User Behaviour prediction module detects that the position information received from the Car GPS VO match with one of the recurrent trips available in the **Real World Knowledge** database, infers the final destination and suggest it to the driver (thus enabling the “Parking At Destination” iCore Service).

### 3.3.4.2. Entities involved

On Board Unit, Management Unit, CVO Execution, Recurrent Trip Learning

### 3.3.4.3. Related requirements

**Table 14: Requirements related to Smart City: transportation test case #4**

Identifier	Name	Description
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe semantically not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-12	The iCore system shall be able to aggregate and infer relevant situational dimensions (e.g. time, place, preference of users) to describe the current situation and anticipate changes to it.	The iCore system is envisioned to describe semantically not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-13	The iCore system shall be able to infer (reason) on user intentions/behaviour in order to support current and future services and service requests.	Importance of storing relationships between objects, people, applications. The iCore system must be adaptable to User intentions/behaviour for dynamic service composition.
SL-F-15	The iCore system shall maintain and be able to store, classify, rank, and annotate a categorisation of different situations based on user intentions/behaviour per	It is an essential element for the iCore system to 'grow' intelligence and foster reuse of iCore objects in multiple services and across multiple situations.

situation for future use.

#### 3.3.4.4. Pre-conditions

The On Board Unit is on.

A connection is available between the On Board Unit and the VOs (WLAN Connection).

The Recurrent Trip Learning module is executed.

#### 3.3.4.5. Execution steps

- Access the Uconnect™ Apps Menu
- Select (launch) the iCore client application
- Wait for the application to run
- While driving toward a predefined recurrent destination, wait for the User Behaviour prediction module to show the inferred destination
- Check if the information shown is correct

#### 3.3.4.6. Success criteria/ Expected output

The test is considered performed with success if the inferred destination corresponds to the actual destination (shown in the upper part of the Uconnect™ display under the label “Current destination”).

As an additional check, the graphical map – representing the entire inferred route – shown in the left part of the Uconnect™ display can be compared with the predefined trip selected for the test.

#### 3.3.4.7. Test Case Result

Tests have been performed with success for each of the recurrent trips stored in the RWK database.

The following picture shows an example of visualization when a recurrent trip is inferred.



Figure 88: The inferred destination and route are shown to the Driver



### 3.3.5. Test case #5 – The CVO are available for execution based on identified recurrent User destination

#### 3.3.5.1. Description

##### Test case #5a

When the User Behaviour prediction module detects a recurrent destination, the “Parking at Destination” service becomes available.

##### Test case #5b

The Parking availability information is correctly and timely sent to the On Board Unit – case “Parking At Destination”

#### 3.3.5.2. Entities involved

OnBoard Unit, Management Unit, CVO Execution, Recurrent Trip Learning

#### 3.3.5.3. Related requirements

Table 15: Requirements related to Smart City: transportation test case #5

Identifier	Name	Description
SL-F-13	The iCore system shall be able to infer (reason) on user intentions/behaviour in order to support current and future services and service requests.	Importance of storing relationships between objects, people, applications. The iCore system must be adaptable to User intentions/behaviour for dynamic service composition.
CVO-F-5	The CVO registry shall be able to obtain Service Logic requirements either by a query (when they are stored somewhere) or by a trigger event (when they are constructed on the fly).	The CVO registry must be able to interface with the upper service level and be able to receive service logic requirements (request parameters).
CVO-F-11	A CVO shall be created based on a template that is produced at design time and intends to	Templates are necessary for a semi-automatic generation of CVOs from data acquisition and events processing.

	define the CVO and determine his capabilities.	
--	--	--

#### 3.3.5.4. Pre-conditions

The On Board Unit is on.

A connection is available between the On Board Unit and the VOs (WLAN Connection)

The iCore Service “Parking At Destination” is enabled in the “iCore Services” sub-menu.

The Smart City Transportation (Smart Parking) “Parking At Destination” CVO is executed (find a parking close to the last car position).

#### 3.3.5.5. Execution steps

##### Test case #5a

- Access the Uconnect™ Apps Menu
- Select (launch) the iCore client application
- Wait for the application to run
- While driving toward the selected, predefined destination, wait for the User Behaviour prediction module to detect a recurrent destination and check if the “Parking At Destination” button becomes available

##### Test case #5b

- Activate the “Parking At Destination” iCore Smart Parking Service from the On Board Unit
- Wait for the parking information to be retrieved and shown to the Driver in a popup window
- Check if the information shown is correct

#### 3.3.5.6. Success criteria/ Expected output

##### Test case #5a

- The “Parking Near Destination” button is enabled on the application HMI as soon as a recurrent destination is identified by the User Behaviour prediction module.

##### Test case #5b

- Once the “Parking At Destination” is activated, the Parking availability information is shown to the Driver by the On Board Unit in a reasonable time (less the 10 seconds)
- The Parking area suggested meets the following requirements:
  - it is the nearest to the destination among those made available by the Open Data Service in use
  - it has at least one free parking slot
- The information shown corresponds to what available in the Open Data Service in use

### 3.3.5.7. Test Case Result

Figure 89, taken from a live execution of the test case, shows the test case result: the “Parking Near Destination” Service activation button is available to the Driver.



**Figure 89: A new Service is available for the Driver**

Several tests have been performed in real driving conditions to check the correctness of retrieved information in correspondence to different recurrent trips/destinations detected. In all cases, the Parking area suggested was identified correctly among those made available by the Turin Open Data Service and the information shown on the display was consistent.

## 3.4 Smart City: Urban security

In section 2.4, we have introduced the smart city urban security case as well as its main Proof of Concept (PoC) components, both hardware and software, including iCore platform. We remind here shortly a high level view and description of the PoC behaviour at functional level focussing on the main iCore platform goals for the use case:

1. To provide to the police operators the best situation awareness to support real-time decision support; it means in our case selecting the minimal and relevant set of video cameras streaming showing accurately the situation on the field also minimizing the cognitive overload of operators.
2. To support and enhance as much as possible the availability and the performance of the surveillance system; it means in our case maximizing the use wireless sensors network taking into account video cameras streaming criticality from an operational point of view.

So, first is about video cameras streaming management, and second is about corresponding network management.

Our technical validation process is strongly based on synthetic environment application that simulates real world urban city with buildings, various city objects and number of bio-inspired citizens as well as sensors deployed within the city by security forces and the exhibition area operator. The city simulation application is then physically connected to the

real surveillance system, a wireless network infrastructure connecting itself iCore platform with control and Command (C2) applications. So the PoC exposes very close characteristics to an operational surveillance system: simulated citizens behave like humans, simulated sensors such as video cameras provide H264 based video streaming, chemical sensors report their detections according to a simulated toxic gas cloud with realistic dispersion and impact over the exhibition area, etc.

The overall validation is made through an operational scenario where a VIP first visits a big exhibition area with a pre-planned trajectory; during the visit an unattended luggage (containing dirty Improvised Explosive Device) explodes a few hundred meters behind the VIP (missing him) triggering panic within the area and generating a highly toxic cloud dispersal; The evacuation of VIP is decided by the C2 police operators; the evacuation path is based on observed situation with presence of crowds and chemical detections possibly “blocking” some of the possible evacuation exists.

So the iCore platform technical validation is driven by “operational way of doing” validation with police forces simulated scenario and urban environment.

The test cases described hereafter correspond to the Service Requests introduced in section 2.4 and all triggered by C2 police operators according to the situation and needs:

- SR1: Areas of interest overview through video cameras,
- SR5: CBRNE monitoring through detections with deployed chemical sensors,
- SR2: VIP tracking and control,
- SR3: Crowds monitoring,
- SR4: Toxic cloud monitoring by people effect through video cameras.

Because of the VIP visit scenario, Service Request #1 and #5 are triggered first before the VIP is on site to set-up situation awareness; then Service Request #2 is triggered to track effectively the VIP when he arrives. Service Request #3 and #4 are triggered immediately after the explosion happens and is confirmed to C2; the Service Template for Service Request #2 also contains the service logic to control the VIP evacuation up to the C2 decided exit.

The Service Requests (except #5) here are also competing for relevant video cameras, so a relative priority is assigned to each one by the planning security forces. They are all issued from the C2 video oriented situation awareness GUI, while #5 is issued via the C2 CBRNE oriented situation awareness GUI.

### 3.4.1. Test case #1 – Areas of interest video overview (video C2 GUI)

#### 3.4.1.1. Description

The objective is to validate areas of interest video overview Service Request. (This is the simplest case, and therefore is tested first.)

A limited number of areas of interest are defined by the security forces based on potential threats, exhibition area geo-configuration, planned VIP visit, etc.

Service Request operational goal here is to provide to C2 operators a view on each area with at least one camera without overloading the network, no camera view redundancy (i.e. multiple cameras showing the same area)..

If the number of areas of interest exceeds network capacity, a video cameras round-robin is implemented over the areas of interest with defined frequency.

Video cameras round-robin minimum viewing frequency can be a C2 operators preference provided as Service Request parameter like priority of the Service Request and video streaming placeholder within the GUI, but can also be considered a known ‘ideal’ number for the application domain, i.e. as RWK, as is assumed here

### 3.4.1.2. Entities involved

All VO components are registered and queried/used from the dynamically running service instances.

The specific SR is triggered from the C2 GUI invoking the instantiation of a service instance via the SRA function, based on a Service Template corresponding with the specific SR, and deployed as a graph of CVO instances by the CVOMU. This results in video streams being activated/deactivated at the RWK-given frequency and according to the RWK-given redundancy across the areas of interest, and according to the available network capacity (SK). At the CVO Level additional CVOMU parts are introduced: a stream manager and a flow priority (and network route) controllers have a one-to-one correspondence to the wireless ad-hoc network used (and so are not CVOs deployed as corresponding to service instances).

### 3.4.1.3. Related requirements

Table 16: Requirements related to Smart City: Urban security test case #1

Identifier	Name	Description
SL-F-3	The iCore system shall enable the search of relevant iCore objects.	iCore users must be able to see / search the space of “available objects”. Services in this use case can use all available VOs, and can query them according to their geographical coverage area.
SL-F-4	The iCore system shall offer a way to handle dynamic iCore Objects/RWOs connections and disconnections gracefully, impacting service execution as minimally as possible.	Applications and services must be able to “carry-on” with minimal features rather than “hang” if objects are disconnected. The iCore platform in this use case uses video streams and sensor data via a stream manager and OGC standard layer hiding any normal as well as unexpected VO volatility.
SL-F-5	iCore external actors shall be able to instantiate new logic and reuse existing logic for dynamic iCore service	This is the systematic ‘programmability’ of an environment of iCore objects. In this use case iCore Service Requests and Service Templates are used for this purpose.

	composition by means of newly generated and existing iCore objects.	
SL-F-9.	The iCore system shall enable the proper and efficient composition of iCore objects that fulfil the query from the service level.	The iCore system shall be able to configure/monitor/select iCore objects on the basis of what the services requested by external actors need. This is fully supported in this use case.
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions. This is ensured in this use case by the semantic annotation of VOs and by the semantic meaning of RWK and RW domain service logic expressed in Service Templates and Service Request, as such allowing for dynamic, predictive situation awareness of service instances.
SL-F-12	The iCore system shall be able to <b>aggregate</b> and infer relevant situational dimensions (e.g. time, place, preference of users) to describe the current situation and anticipate changes to it.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions. This is ensured in this use case by the semantic annotation of VOs and by the semantic meaning of RWK and RW domain service logic expressed in Service Templates and Service Request, as such allowing for dynamic, predictive situation awareness of service instances. Moreover, the iCore platform in this use case is able to aggregate this situation awareness over all service instances for prioritizing network resource use dynamically.
SL-F-13	The iCore system shall be able to <b>infer</b> (reason) on user intentions/behaviour in order to support current and future services and service	Importance of storing relationships between objects, people, applications. The iCore system must be adaptable to user intentions/behaviour for dynamic service composition. In this use case, behaviour of several RW phenomena is tracked and reasoned upon for predicting relevance and priority of video streams

	requests.	for current and upcoming service requests.
SL-F-14	The iCore system may have an interface where users can explicitly describe or query his/her intentions/behaviour if necessary regarding his/her current situation (iCore is supposed to be able to sense those changes).	Simple API to communicate user intentions, semantic interrogation (SPARQL) for current situation and personal history. Feature is addressed to the cases where “observation” capability at Service Level can’t perform or direct information from the user when demanded. The service requests in this use case are such live queries that abstract a situation/phenomenon and can track it upon simple request, and execute the service instance(s) according to the situation.
SL-F-15	The iCore system shall maintain and be able to store, classify, rank, and annotate a categorisation of different situations based on user intentions/behaviour per situation for future use	It is an essential element for the iCore system to ‘grow’ intelligence and foster reuse of iCore objects in multiple services and across multiple situations. In this use case situation awareness is obtained via RWK.
CVO-F-1.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO. CVOs in this use case comply to this.
CVO-F-2.	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters). Situation Observers are considered CVO component graph instances that can be reused by multiple simultaneously running service instances. VOs may be used also directly by multiple service instances; a VO-controller CVO is used to arbitrate/prioritise control commands from multiple



		service instances to a single VO instance.
CVO-F-5	The CVO registry shall be able to obtain Service Logic requirements either by a query (when they are stored somewhere) or by a trigger event (when they are constructed on the fly).	The CVO registry must be able to interface with the upper service level and be able to receive service logic requirements (request parameters). The iCore platform used in this use case has a CVO Level API that supports issuing of SERs, which can hold the criteria for dynamic CVO and VO selection.
CVO-F-6	The CVO registry shall be able to address and name stored VOs and CVOs.	Stored VOs and CVOs must have an address and name in order to be retrieved, e.g. as web services. The CVO Level of the iCore platform used in this use case maintains all needed bookkeeping to reuse or reconnect existing VO and CVO instances.
CVO-F-11	A CVO shall be created based on a template that is produced at design time and intends to define the CVO and determine his capabilities.	Templates are necessary for a semi-automatic generation of CVOs from data acquisition and events processing. CVO Templates are used in this use case for automated instantiation and deployment of CVOs upon SER fulfilment.
CVO-F-17	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters). Situation Observers are considered CVO component graph instances that can be reused by multiple simultaneously running service instances. VOs may be used also directly by multiple service instances; a VO-controller CVO is used to arbitrate/prioritise control commands from multiple service instances to a single VO instance. The iCore platform in this use case is able to aggregate this situation awareness over all service instances for prioritizing network resource use dynamically.

VO-F-1	VO-RWO coupled-decoupled notification	iCore platform as well as C2 operators must be warned when connection with deployed sensors/actuators is lost
VO-F-2	VO-RWO coupled notification, VO registered	iCore platform as well as C2 operators must be warned when deployed sensors/actuators are fully registered (discovery); devices are usable
VO-F-3	Proximity of RWOs is formalized with VOs	geographical proximity and wireless communication proximity are provided to iCore platform as well as C2 operators
VO-F-4	VOs lookup based on RWOs geographical location	iCore platform retrieves relevant sensors based on their geographical location
VO-F-5	New RWOs deployed at run-time creates new VOs	iCore platform supports devices deployment at run-time
VO-F-6	VO interface	VO has interface
VO-F-7	VO geo-location	VO saves the geo-location of its related RWO
VO-F-12	VO describes RWO	VO has interface that describes semantically RWO

#### 3.4.1.4. Pre-conditions

Deployed video cameras are registered within the VO registry.

As Real World Knowledge (RWK): Areas of interest (*polygons*) are defined by C2 operators through C2 GUI (geo-area selection), indicating to the system which areas are distinguished by the human users and considered important for the scene.

Video cameras main characteristics are retrieved via their VOs (installation GPS location, coverage area as triangle, etc.), which determines which camera is able to cover which area of interest.

As System Knowledge (SK), network maximum throughput capacity indicates the maximum number of simultaneous video streams (at given bandwidth requirements for each type) that the network can handle at best. This gives the iCore platform an upper bound to obey irrespective of the number of active service requests. (For additional optimisations flows are defined and routing is determined, given the network topology that can be retrieved via network manager.)

#### 3.4.1.5. Execution steps

Selection of tab “areas of interest overview” through video C2 GUI.

This implies the deployment of the service instance according to iCore principles, with use of relevant RWK and SK for controlled network loading.

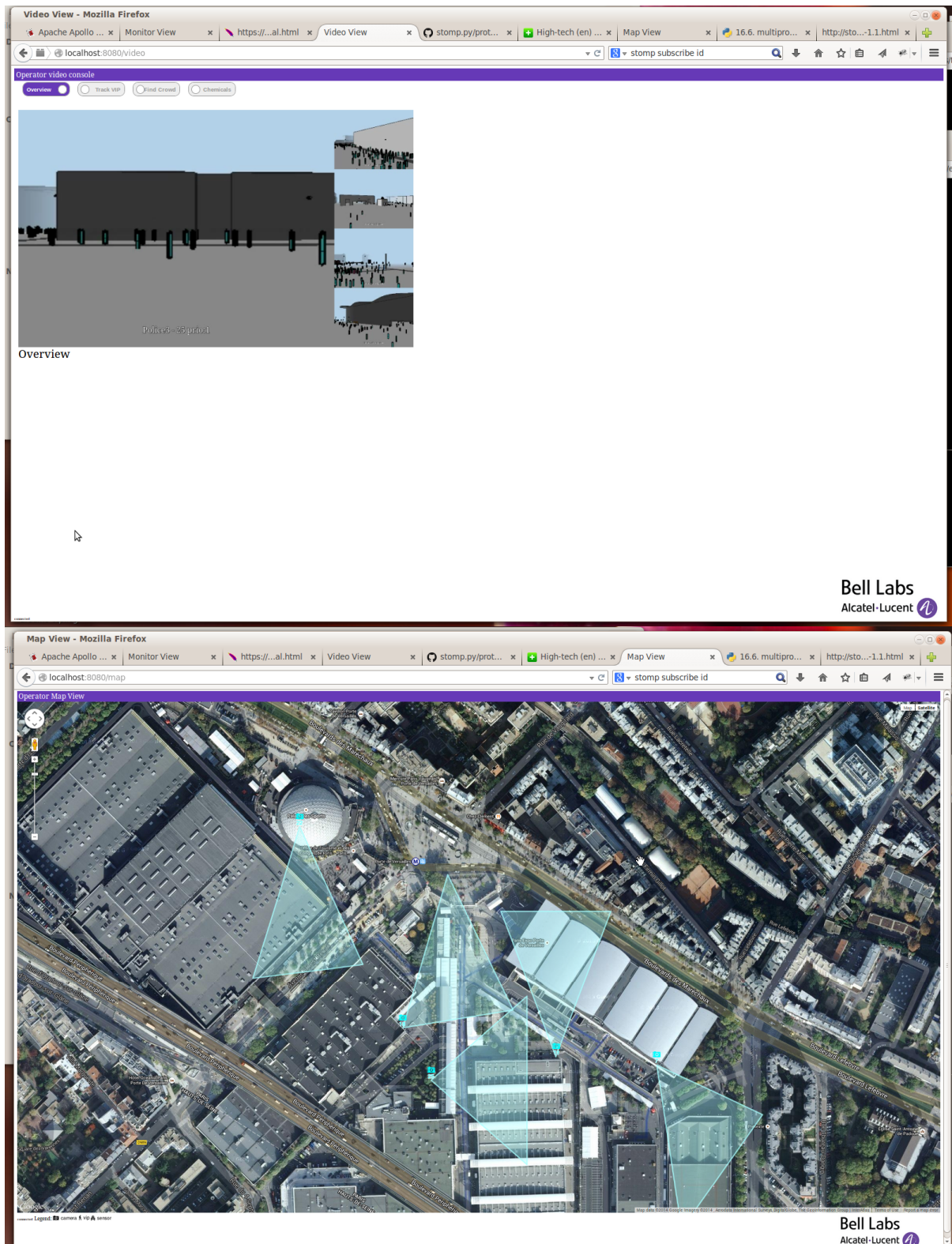
#### **3.4.1.6. Success criteria/ Expected output**

Video streaming is controlled by the iCore platform in such a way that all areas of interest are monitored with at least one camera according to the round-robin viewing minimum frequency.

Number of video-streams is never above the maximum defined keeping the network load under control (basic control for this basic first case).

#### **3.4.1.7. Test Case Result**

The demonstrator successfully achieved the expected output and network loading success criteria, as illustrated by Figure 90.



**Figure 90: Smart city urban security use case - video based Control and Command and areas overview**

### 3.4.2. Test case #2 – CBRNE monitoring through detections with deployed chemical sensors (CBRNE C2 GUI)

#### 3.4.2.1. Description

Objective is to validate chemical detections based CBRNE monitoring Service Request. Service Request operational goal here is to provide to C2 operators monitoring of toxic chemical cloud dispersal. Information is gathered here about type of chemical particles detected, level of toxicity, etc.

#### 3.4.2.2. Entities involved

All VO components are used. No CVO or Service components are associated with the C2 CBRNE GUI.

#### 3.4.2.3. Related requirements

Table 17: Requirements related to Smart City: Urban security test case #2

Identifier	Name	Description
See test case #1 VO requirements	See test case #1 VO requirements	See test case #1 VO requirements

#### 3.4.2.4. Pre-conditions

Deployed chemical sensors are registered within the VO registry. Chemical sensors main characteristics are retrieved via their VOs (installation GPS location, type of sensors, etc.). Sensors are shown within the CBRNE C2 GUI.

#### 3.4.2.5. Execution steps

For all sensors, detections frequency is defined through “register” button with CBRNE C2 GUI thus triggering reception of sensors readings by C2 GUI.

#### 3.4.2.6. Success criteria/ Expected output

Before the VIP visit with no toxic cloud, all CBRNE sensors report detections with sulphur gas concentration equal to 0.

After explosion, during toxic cloud dispersal, CBRNE sensors report increasing sulphur gas concentration consistently around the explosion point (centre of next screenshot).



### 3.4.2.7. Test Case Result

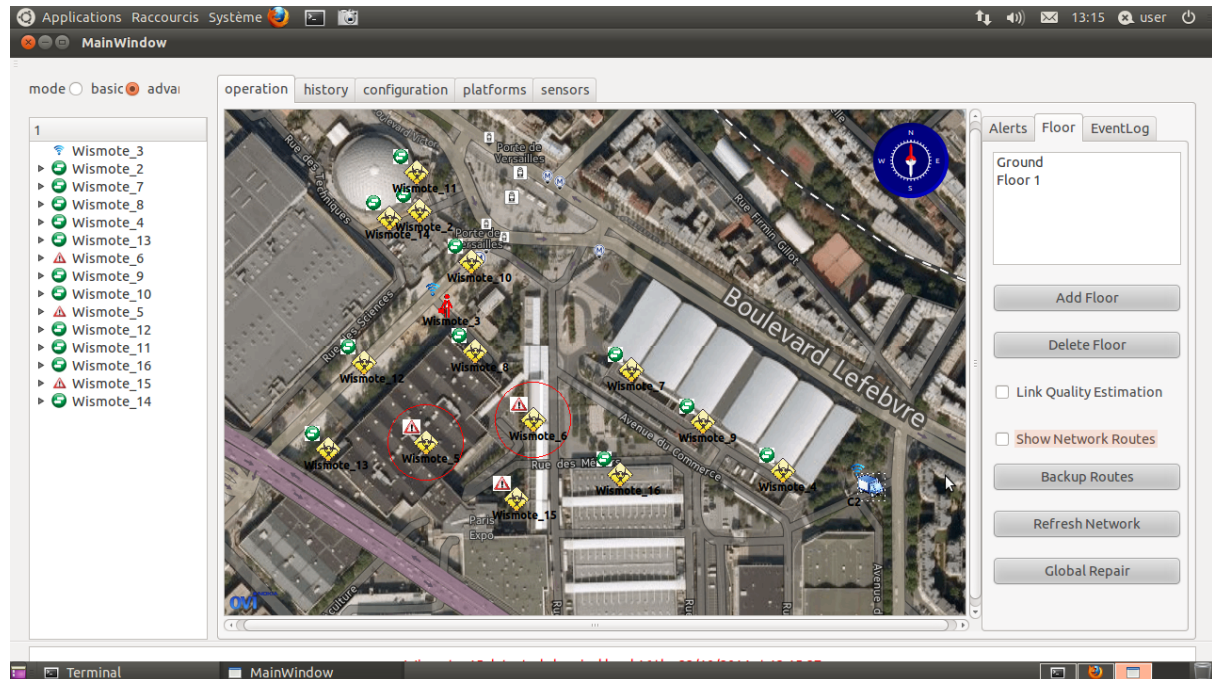


Figure 91: Smart city urban security use case - CBRNE Control and Command and detected toxic cloud

### 3.4.3. Test case #3 – VIP tracking and control (video C2 GUI)

#### 3.4.3.1. Description

Objective is to validate VIP tracking and control Service Request. (This is the first SR with live phenomenon tracking. Tested in combination with SR1.)

Service Request operational goal here is:

- To provide to C2 operators at all-time a view on the VIP and his protection team with one camera.
- To provide to C2 operators a mean to control the VIP evacuation up to the decided exit.

(This is the 'main' SR, which is now combined with the SR1 and SR5 from Test cases 1 and 2. The added complexity is the use of RWK for person tracking, and the potential concurrent activation of SR1 and SR5. So, this is the first test case where 2 SRs, SR1 and SR2, compete for VO control and network resources.)

During his visit, the VIP pre-planned trajectory as a set of waypoints is based on some of the areas of interest already defined for Service Request #1. This video tracking uses the VIP GPS sensor in combination with the course waypoints to be able to predict the near future VIP location.

Service Request has 4 parameters; VIP identifier, the priority of the Service Request (highest priority for this service request), a video streaming placeholder within the GUI and the VIP visit path way points, the last parameter in fact being RWK considered for the whole mission scenario as planned by the security forces.

### 3.4.3.2. Entities involved

All VO components are used.

The specific SR is triggered from the C2 GUI invoking the instantiation of a service instance via the SRA function, based on a Service Template corresponding with the specific SR, and deployed as a graph of CVO instances by the CVOMU. It results in video streams being activated/deactivated based on the RWK-based VIP trajectory prediction, and according to the available network capacity (SK). At the CVO Level the resource-competing service instances (SR1 and SR2 invoked) are combined, by means of combining their Situation Observers and overall SR-based priority into a dynamic overall video stream prioritization. This is served by a single CVO instance that provides input to the flow priority (and network route) controller of the system and receives live inputs from a varying number of Situation Observers-based predictors (on the fly extended per further SRs issued). As such it is ensured that the most critically needed video streams get priority for network resource spending.

### 3.4.3.3. Related requirements

**Table 18: Requirements related to Smart City: Urban security test case #3**

Identifier	Name	Description
See test case #1 SL, CVO and VO requirements	See test case #1 SL, CVO and VO requirements	See test case #1 SL, CVO and VO requirements

### 3.4.3.4. Pre-conditions

The Service Request uses the same pre-conditions as in Service Request #1 (areas of interest overview) and requires a few other ones.

The VIP GPS sensor is registered within the VO registry. The VIP arriving at the exhibition area is geo-localized by means of his GPS (as simulated in Se-Star).

As additional RWK, a set of course waypoints (typically corresponding to the granularity of the areas of interest from Test case 1) represents the planned VIP trajectory. (In combination with the GPS current position and speed, the VIP short-term trajectory can be predicted for predictive anticipation of video stream relevance.)

As further additional RWK, in order to support the security forces in evacuation events, the planned trajectory waypoints are each extended with several alternative evacuation paths. This is the evacuation plan of which an auto-selected subset will be presented in the C2 GUI in case of VIP evacuation.

### 3.4.3.5. Execution steps

Selection of tab “VIP tracking” through the video C2 GUI is again implying the deployment of the service instance according to iCore principles, with use of relevant RWK and SK for controlled network loading.



When an explosion happens, the evacuation paths relevant to the VIP's current position are displayed for C2 operators' decision, which is implied an update of the further VIP trajectory according to the plan.

Next to video stream selection according to VIP position and predicted position, this test case also adds PTZ control of video camera VOs by a service instance, i.e. for tracking the VIP and 'handing over' camera views for continuity. All this is done while also a service instance for SR1 is executing (making a totally unrelated video selection), but overall stream selection prioritization is managed at the CVO Level according to RWK and SR-based configuration of priority.

#### **3.4.3.6. Success criteria/ Expected output**

The VIP GPS based location is sent to the iCore platform (and C2 operators via GUI) according to VIP path.

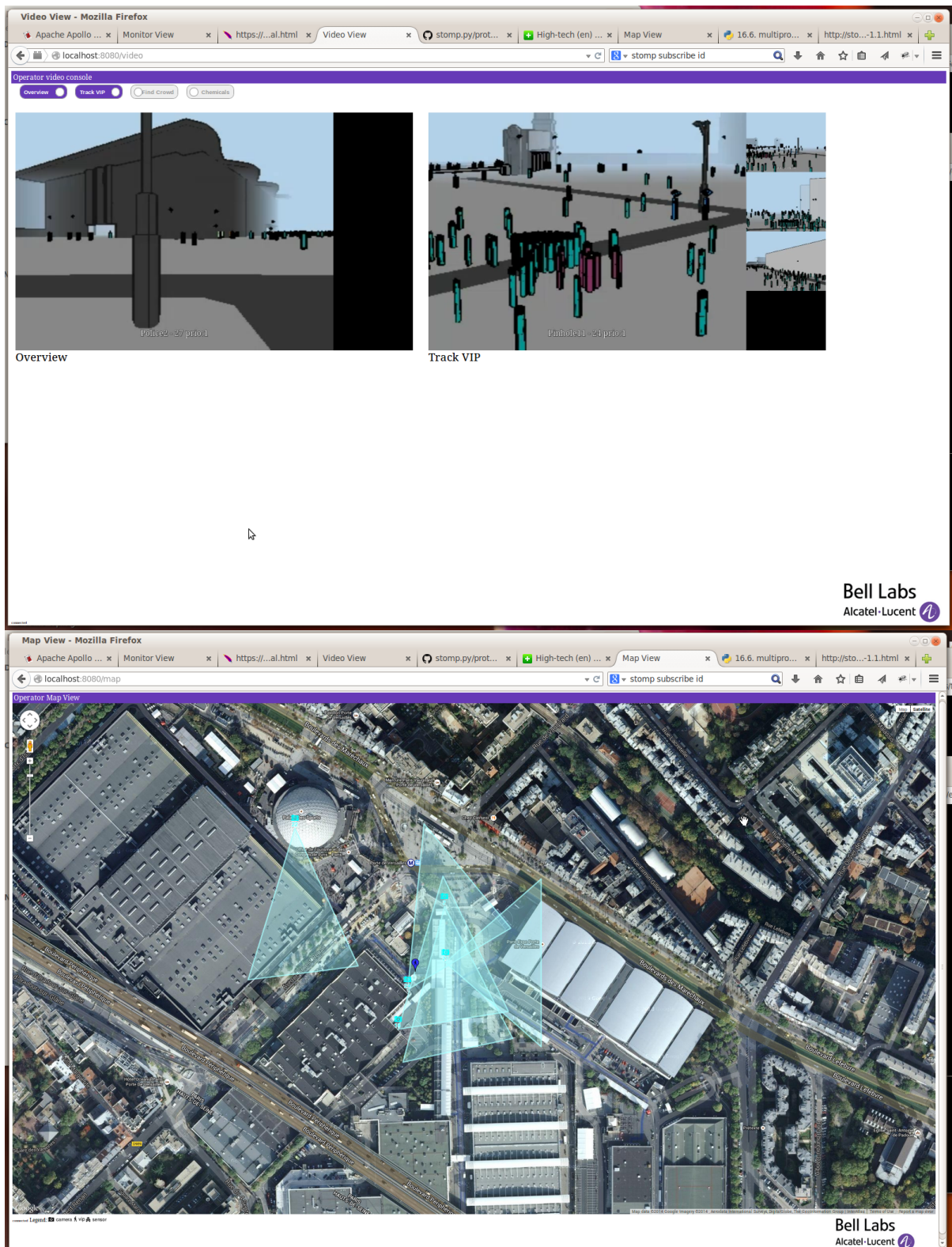
Video streams are selected by the iCore platform, so as to show the most critical video streams for visual VIP tracking by the C2 operators.

During VIP evacuation, the iCore platform dictates the VIP evacuation waypoints to the Se-Star simulator upon choice of the evacuation path by the C2 operator (as if the C2 operator gives evacuation commands to the VIP body guards after the decision).

The number of video-streams is never above the maximum defined keeping the network load under control, despite the fact that two totally different service instances are simultaneously running and the right priority of streaming is kept dynamically under the multi-service instance condition.

#### **3.4.3.7. Test Case Result**

The demonstrator successfully achieved the expected output and network loading success criteria, as illustrated by Figure 92.



**Figure 92: Smart city urban security use case - video based Control and Command and tracked VIP**

### 3.4.4. Test case #4 – Crowds monitoring (video C2 GUI)

#### 3.4.4.1. Description

Objective is to validate crowds tracking Service Request. (This is a second phenomenon being tracked, adding to a further more complex composition of 3 service instances being executed simultaneously.)

Service Request operational goal here is to provide to C2 operators at all-time a view on biggest crowds (highest number of people within a given video camera field view) within the exhibition area; during VIP evacuation, crowds representing potential passive threats that can block the VIP evacuation path.

Crowds' detection is primarily supported by video cameras than are able to send number of people within camera field view and related speed vector. Crowds are detected based on a pre-configured number of people threshold (RWK definition of 'crowd') and according to an RWK-defined crowd detection mechanism.

#### 3.4.4.2. Entities involved

All VO components are used.

As always, the specific SR is triggered from the C2 GUI invoking the instantiation of a service instance via the SRA function, based on a Service Template corresponding with the specific SR, and deployed as a graph of CVO instances by the CVOMU. It results in video streams being activated/deactivated according to the overall priority of streams, given the multiple phenomena tracked as related to the active services' Situation Observers.

#### 3.4.4.3. Related requirements

Table 19: Requirements related to Smart City: Urban security test case #4

Identifier	Name	Description
See test case #1 SL, CVO and VO requirements	See test case #1 SL, CVO and VO requirements	See test case #1 SL, CVO and VO requirements

#### 3.4.4.4. Pre-conditions

Same pre-conditions as in previous test cases, except for the addition of the mentioned crowd detection RWK and Service Template.

#### 3.4.4.5. Execution steps

Selection of tab "Crowds tracking" through video C2 GUI, followed by regular iCore platform operation as before, now running more service instances simultaneously, with more complex prioritisation of streams as a result.

#### **3.4.4.6. Success criteria/ Expected output**

Crowds compliant to the crowd defined threshold should be effectively tracked and corresponding most relevant video streams should be displayed, again obeying the network (and human cognitive overload) limitations dynamically.

The Se-Star application viewer allows verifying whether this happens correctly.

#### **3.4.4.7. Test Case Result**

The demonstrator successfully achieved the expected output and network loading success criteria, as illustrated by Figure 93.



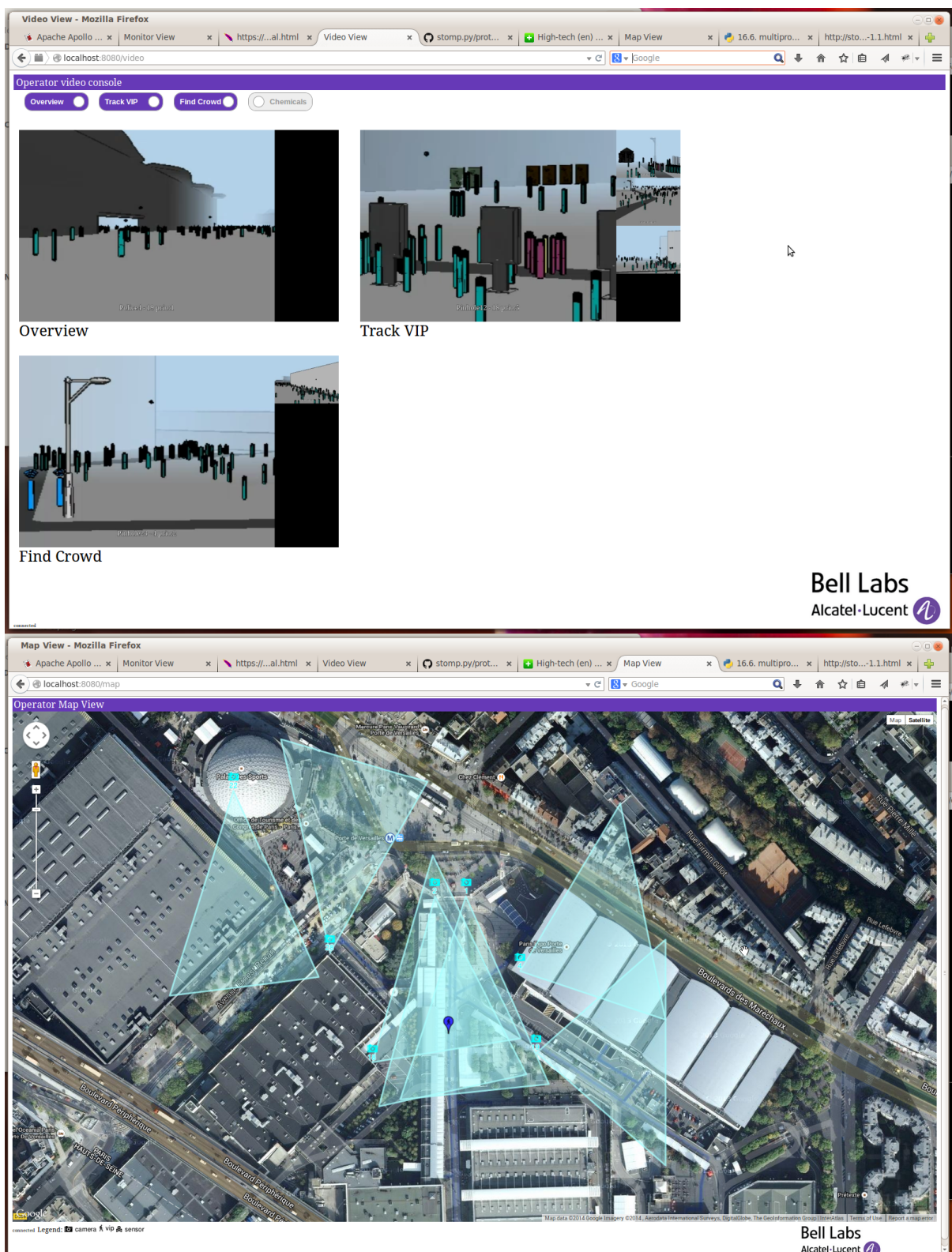


Figure 93: Smart city urban security use case - video based Control and Command and crowds tracking

### 3.4.5. Test case #5 – Toxic cloud monitoring by people effect through video cameras (video C2 GUI)

#### 3.4.5.1. Description

Objective is to validate toxic cloud monitoring by effect on people Service Request. (This is a third phenomenon tracked, this time based on chemical sensor readings & corresponding RWK-based prediction.)

Service Request operational goal here is to provide to C2 operators a view on people impacted by toxic cloud, as well as to take this into account as VIP evacuation decision support.

Impacted area is initially detected through chemical threshold detections on all deployed chemical sensors.

#### 3.4.5.2. Entities involved

All VO components are used.

As before, the specific SR is triggered from the C2 GUI invoking the instantiation of a service instance via the SRA function, based on a Service Template corresponding with the specific SR, and deployed as a graph of CVO instances by the CVOMU. This results in video streams being activated/deactivated according to the overall priority of streams, given the still multiple phenomena tracked as related to the active services' Situation Observers.

#### 3.4.5.3. Related requirements

Table 20: Requirements related to Smart City: Urban security test case #5

Identifier	Name	Description
See test case #1 SL, CVO and VO requirements	See test case #1 SL, CVO and VO requirements	See test case #1 SL, CVO and VO requirements

#### 3.4.5.4. Pre-conditions

Same pre-conditions as in previous test cases, except for the addition of the mentioned chemical detection prediction RWK and Service Template. Chemical detections are sent to iCore platform.

The RWK used is a simplified chemical cloud dispersal prediction model (assuming a growing circle) and pre-configured levels/thresholds of gas concentration used to show the danger level to C2 operators.

#### 3.4.5.5. Execution steps

Selection of tab “Toxic cloud by effect on people tracking” through video C2 GUI, followed by regular iCore platform operation as before, now running still more service instances simultaneously, with still more complex prioritisation of streams as a result.

#### **3.4.5.6. Success criteria/ Expected output**

People falling down are effectively shown to C2 operators. View may be interrupted sometimes depending of higher priority Service Requests.

Impacted area is shown and updated with concentric circles (red, orange, etc.) provides a first indication about gas concentration and danger level for people life.

As always, the number of simultaneous video streams should never exceed the maximum allowed and the network load is kept under control dynamically for the complexity of the full set of active SRs.

Again the Se-Star application viewer can be used to verify that this is running correctly.

#### **3.4.5.7. Test Case Result**

The demonstrator successfully achieved the expected output and network loading success criteria, as illustrated by Figure 94.





Figure 94: Smart city urban security use case - video based Control and Command and toxic cloud tracking

### 3.5 Smart Business: Supply Chain Management and Logistics

#### 3.5.1. Test case #1 Transportation reports

##### 3.5.1.1. Description

This test case validates the following features of the prototype:

- Monitor continuously the transportation conditions of the parcels;
- Identify the moments of time when the transportation conditions are not fulfilled;
- When the transportation process is finished, generate the report for the transportation company clients.

##### 3.5.1.2. Entities involved

Temperature and humidity VOs, VO registry, CVO management unit, perishable/non-perishable products, monitoring CVO templates, CVO management unit, RWK.

##### 3.5.1.3. Related requirements

Table 21: Requirements related to Smart Business test case #1

Identifier	Name	Description
SL-F-1	The iCore platform requires a service model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-2	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-9	The iCore system shall enable the proper and efficient composition of iCore objects that fulfil the query from the service level.	The iCore system should be able to configure/monitor/select iCore objects on the basis of what the services requested by external actors need.
CVO-F-1	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.

CVO-F-2	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters).
CVO-F-11	A CVO shall be created based on a template that is produced at design time and intends to define the CVO and determine his capabilities.	Templates are necessary for a semi-automatic generation of CVOs from data acquisition and events processing.
VO-F-1	The VO Registry shall be able to maintain and record the links with each RWO (i.e. send notification (signalling) when coupled and when decoupled).	Handle to the real world is maintained in a best possible way. Awareness regarding the RWO.
VO-F-2	The VO shall be able to name, type (Metadata) and address each RWOs that becomes available (i.e. at the point of discovery).	RWOs are appropriately named and addressed within their respective proximity domains.
VO-F-6	A VO should describe the available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe-threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
VO-F-14	VOs shall be enabled to adapt to the mobility of the iCore system (or part of it) or user.	This requirement deals with mobility issues and all changes that are manifested because of the mobility of actors and devices of the iCore system.

#### 3.5.1.4. Pre-conditions

- A set of sensors is deployed in a container;
- The sensors are semantically described in the VO registry;
- The acceptance criteria are included in the RWK for the product which will be transported.

#### 3.5.1.5. Execution steps

The usual steps that have to be accomplished in order to register a product transportation request are listed below:

1. [Transport company client] Using the GUI, he/she fills in the transportation details: product type, sender and receiver address, delivery time;
2. [Transport company client] Hand over the package that has to be transported to the transport company employee.
3. [Transport company employee] Selects the appropriate sensor, attaches it to the package, registers the binding using the employee GUI.
4. [Transport company employee] Triggers the transportation start.

At the end, when the products are transported, the transport company client can check the transportation conditions using the GUI.

#### 3.5.1.6. Success criteria/ Expected output

A product transportation report which contains:

- All the raw data collected during transportation (less than 2% missing data);
- The considered transportation criteria of the products;
- Highlight the moments of time when the products are not transported according to the transportation criteria.

#### 3.5.1.7. Test Case Result

In the following snapshot is presented the transportation report generated after a product had been transported.



**Table 22: The transportation report generated after a product had been transported.**

Several tests have been successfully performed to check the correctness of retrieved information in correspondence to different transportation scenarios.

### 3.5.2. Test case #2: Early warnings

#### 3.5.2.1. Description

Getting the predicted temperature values for parcel's sensor for a 10 minutes, 1 hour and two hours forecasting window.

#### 3.5.2.2. Entities involved

VOs for sensor indicating inside and outside temperature, humidity, current clamp, AC unit state, and parcel temperature, VO registry, CVO execution engine, CVO management unit, Service request analysis, RWK (which contains the prediction models), prediction CVO;

#### 3.5.2.3. Related requirements

**Table 23: Requirements related to Smart Business test case #2**

Identifier	Name	Description
SL-F-2	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-9	The iCore system shall enable the proper and efficient	The iCore system should be able to configure/monitor/select iCore objects on the basis of what the services requested by external actors need.

	composition of iCore objects that fulfil the query from the service level.	
CVO-F-1	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-2	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters).
CVO-F-11	A CVO shall be created based on a template that is produced at design time and intends to define the CVO and determine his capabilities.	Templates are necessary for a semi-automatic generation of CVOs from data acquisition and events processing.
VO-F-1	The VO Registry shall be able to maintain and record the links with each RWO (i.e. send notification (signalling) when coupled and when decoupled).	Handle to the real world is maintained in a best possible way. Awareness regarding the RWO.
VO-F-2	The VO shall be able to name, type (Metadata) and address each RWOs that becomes available (i.e. at the point of	RWOs are appropriately named and addressed within their respective proximity domains.



	discovery).	
VO-F-6	A VO should describe the available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe-threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
VO-F-14	VOs shall be enabled to adapt to the mobility of the iCore system (or part of it) or user.	This requirement deals with mobility issues and all changes that are manifested because of the mobility of actors and devices of the iCore system.

#### 3.5.2.4. Pre-conditions

The VOs linked to the inside and outside temperature, humidity, current clamp, AC unit state, and parcel temperature are properly selected from the VO registry, and they transmit the recorded value to the prediction CVO; the prediction CVO is passed from the Service Level, deserialized at the CVO level and is linked to the corresponding CEP outputs.

#### 3.5.2.5. Execution steps

The same steps as in 3.5.1.5 are used for starting the process. When the transportation is triggered, the prediction CVO in JVM serialized form is handled to the CVO level. At the CVO level it is deserialized and embedded into a CVO. The CVO is fed with complex events generated by the CEP engine.

#### 3.5.2.6. Success criteria/ Expected output

The prediction CVO must receive the most recent inputs registered events, as filtered by the CEP statements. The predicted temperature values are stored as RWI, together with the actual values. They are mainly used for test case #3. Later on, the ML expert can perform model assessment based on the predicted-output values.

#### 3.5.2.7. Test Case Result

The evolution of the predicted temperature is depicted in Figure 95 for the 10 minutes forecasting horizon. Figure 96 shows the actual and predicted values, for 10 minutes forecasting horizon.



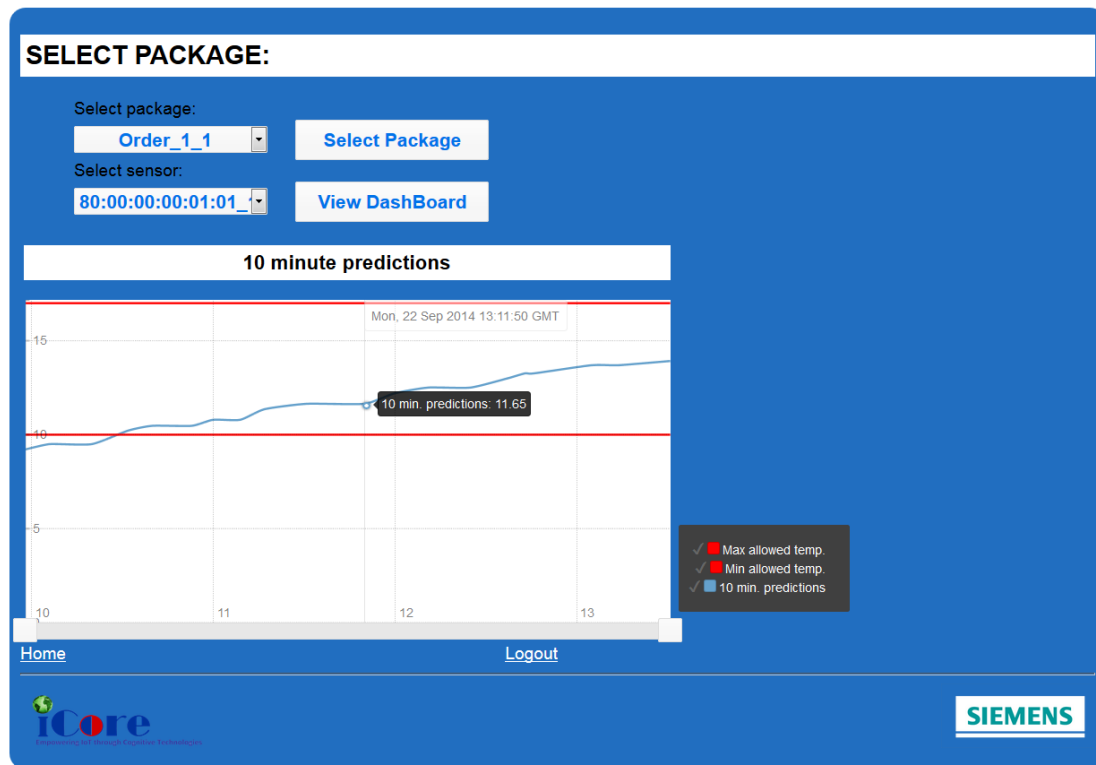


Figure 95: The blue line shows the predicted temperature value of the parcel, based on the current input provided by VOs. The second dropdown contains the ID of the temperature sensor attached to the parcel.

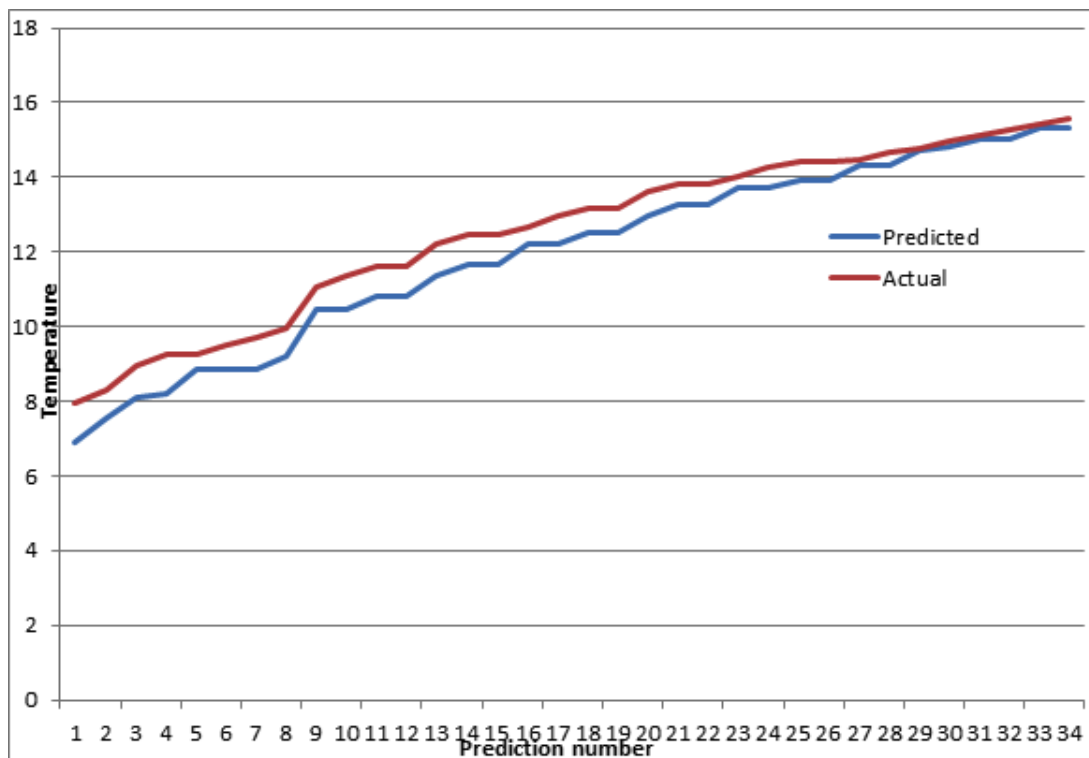


Figure 96: Predicted vs. actual values for the temperature sensor attached to a parcel, for a 10 minute forecasting horizon

### 3.5.3. Test case #3 AC unit control based on container internal conditions

#### 3.5.3.1. Description

The test case is used to validate the AC unit control CVOs, which are responsible to monitor the transportation conditions inside a container (using the temperature sensors) and to generate the appropriate commands for the AC unit of the container.

#### 3.5.3.2. Entities involved

VOs for sensor indicating inside temperature and parcel temperature; AC unit control CVO template, CVO management unit.

#### 3.5.3.3. Related requirements

Table 24: Requirements related to Smart Business trial test case #1

Identifier	Name	Description
SL-F-2	The iCore platform requires a situational model enabling the representation of situational domains.	iCore must offer an environment that enables the relevant iCore objects, applications and service profiles to be distinguished, modelled and stored for future reference, adaptation and reasoning upon.
SL-F-9	The iCore system shall enable the proper and efficient composition of iCore objects that fulfil the query from the service level.	The iCore system should be able to configure/monitor/select iCore objects on the basis of what the services requested by external actors need.
CVO-F-1	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-11	A CVO shall be created based on a template that is produced at design time and intends to define the CVO and	Templates are necessary for a semi-automatic generation of CVOs from data acquisition and events processing.

	determine his capabilities.	
VO-F-1	The VO Registry shall be able to maintain and record the links with each RWO (i.e. send notification (signalling) when coupled and when decoupled).	Handle to the real world is maintained in a best possible way. Awareness regarding the RWO.
VO-F-2	The VO shall be able to name, type (Metadata) and address each RWOs that becomes available (i.e. at the point of discovery).	RWOs are appropriately named and addressed within their respective proximity domains.
VO-F-6	A VO should describe the available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe-threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
VO-F-14	VOs shall be enabled to adapt to the mobility of the iCore system (or part of it) or user.	This requirement deals with mobility issues and all changes that are manifested because of the mobility of actors and devices of the iCore system.

#### 3.5.3.4. Pre-conditions

The VOs linked to the inside temperature, AC unit state, and parcel temperature are properly selected from the VO registry, and they transmit the recorded value to the AC unit control CVO.

#### 3.5.3.5. Execution steps

The same steps as in 3.5.1.5 are used for starting the process. The CVO management unit deploys the AC unit control CVO in the execution engine. The CVO receives the temperature

measurements and using the products transportation requirements computes the appropriate commands for the AC unit.

### 3.5.3.6. Success criteria/ Expected output

The temperature inside the container must remain constant and within the acceptance range regards less the outside temperature variation.

### 3.5.3.7. Test Case Result

In order to test the AC unit control feature of the prototype, the transport emulator (see Figure 97) had been used to generate the temperature data (see Figure 97) and to react at the control messages (the AC unit deployed in the emulated container). Several test had been performed in different conditions and the CVO generated the appropriate commands for the AC unit (the temperature inside the container converged to the stable value within the acceptance criteria).

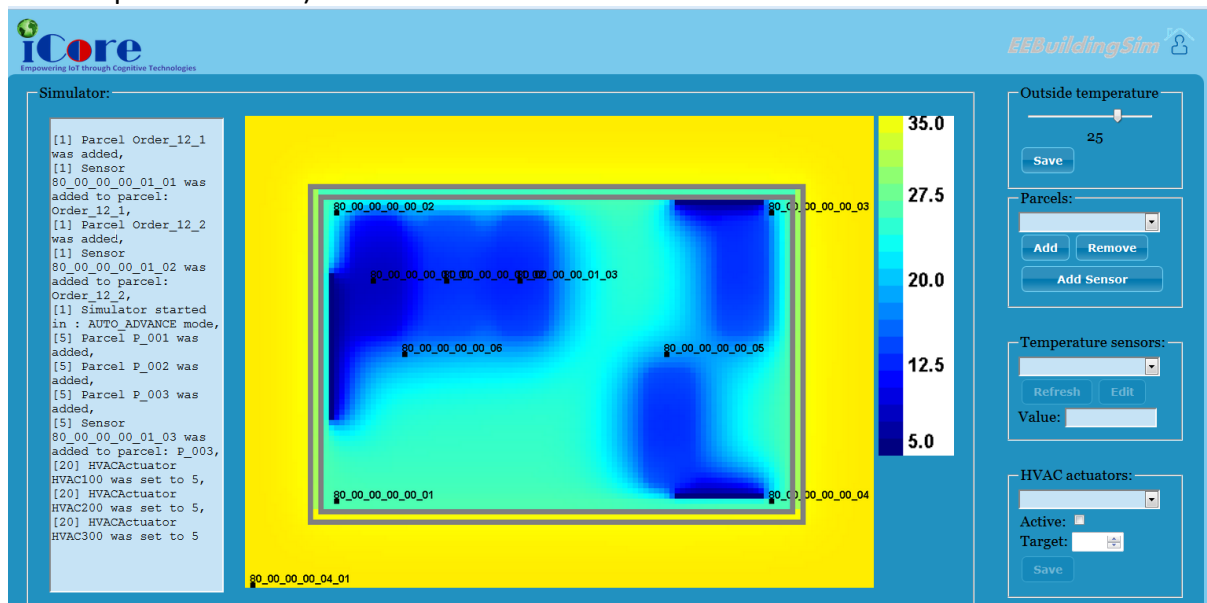


Figure 97: The transportation emulator interface.

## 3.6 Smart Tour in the city trial

### 3.6.1. Test case #1 – Retrieval of best route according to live traffic in Athens

#### 3.6.1.1. Description

This test case's objective is to provide an insight into the performance of the Traffic Analyzer CVO when retrieving and processing live traffic data for Athens. The test was performed on a laptop PC with:

- i7 4700MQ Intel CPU (3.22 GHz)
- 8 GB RAM
- 100Mbit internet connection

### 3.6.1.2. Entities involved

Smart Tour Android application, Traffic Analyzer CVO, Athens Traffic VO

### 3.6.1.3. Related requirements

Table 25 represents the functional requirements (according to deliverable D2.1) related to this test case.

**Table 25: Requirements related to Smart Tour in the City trial test case #1**

Identifier	Name	Description
CVO-F-1	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-3	The CVO functionality shall be able to autonomically select and integrate the correct VOs that describe and match the current / potential service requirements.	The CVO functionality shall be able to autonomically select and integrate the correct VOs that describe and match the current / potential service requirements.
CVO-F-15	CVOs shall be enabled to adapt to the Mobility of the iCore system (or part of it) or user.	This requirement deals with the changes that are manifested because of mobility of actors and devices of the iCore system.

### 3.6.1.4. Pre-conditions

1. Traffic Analyzer CVO is running
2. Smart Tour android application is running on at least one device
3. Athens traffic VO is available
4. The origin point of the routes is the UPRC's lab location (Odyssea Androutsou 144-150)

### 3.6.1.5. Execution steps

Sign in (or register) through Smart Tour.

In the Navigation tab:

- Insert "Chalandri" in the destination field and select enter
- From the dropdown list select "Chalandri, Chalandri, Greece"
- Select Driving mode
- Select "Avoid Traffic"
- Click Calculate routes

### 3.6.1.6. Success criteria/ Expected output

A successful test means a traffic value has been determined for every route received by Traffic Analyzer and the best route is returned to the android application.

The expected output is a relatively slow response for a “cold start” (completely unknown routes) and a very fast response for a “hot start” (known or partially known routes).

### 3.6.1.7. Test Case Result

The live traffic is loaded successfully every hour from the Greek Police by the Athens traffic VO. The operation takes 0.8 – 2 seconds to complete.

The first iteration of the test took 65.31 seconds to complete (no geocoding cache was used). Of these, 65.07 seconds were spent geocoding points (latitude / longitude pairs). The best route was successfully sent to the user.

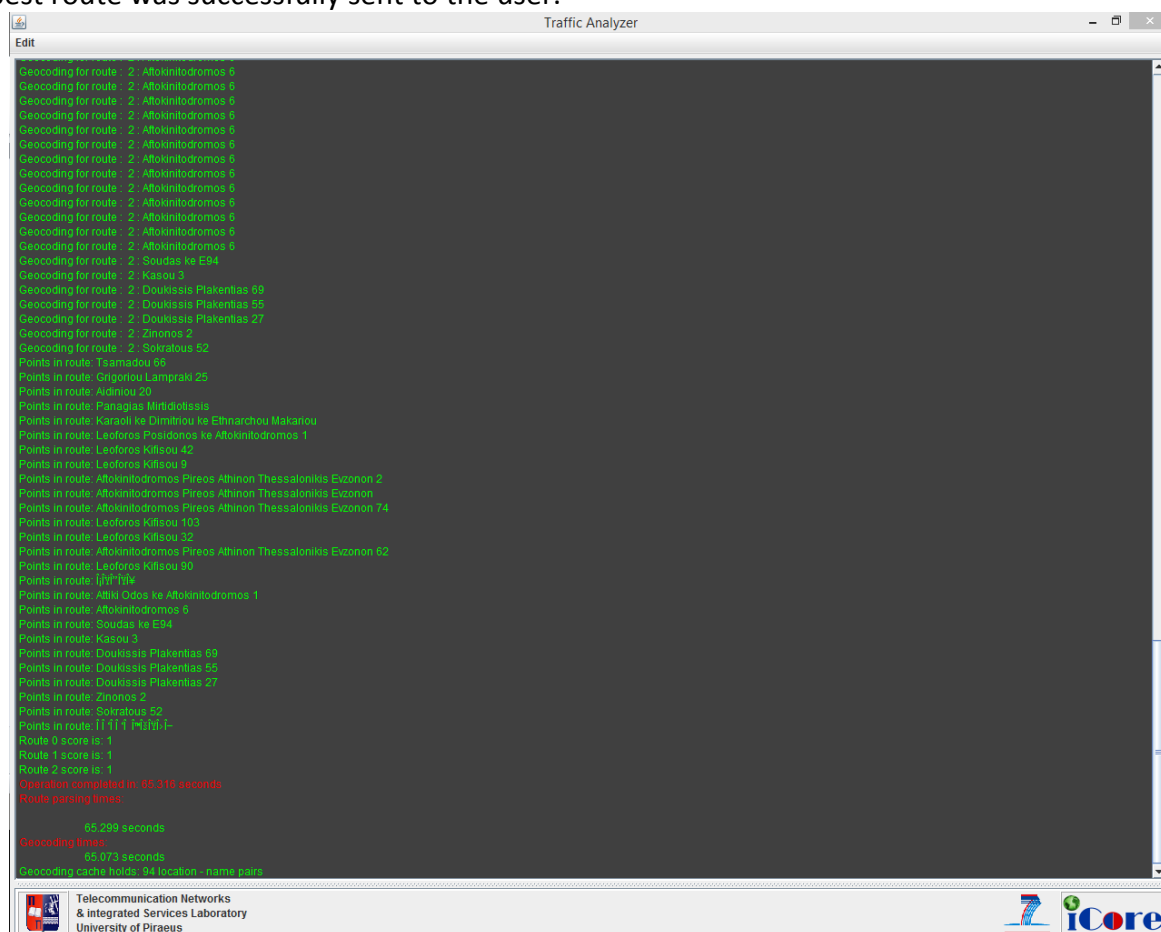


Figure 98: Output of Traffic Analyzer CVO – first iteration

The second iteration of the test took 0.124seconds to complete (only geocoding cache was used). No geocoding was done. The best route was successfully sent to the user. The reason for the speed increase is that no geocoding needed to be done as all the routes were known (cached).

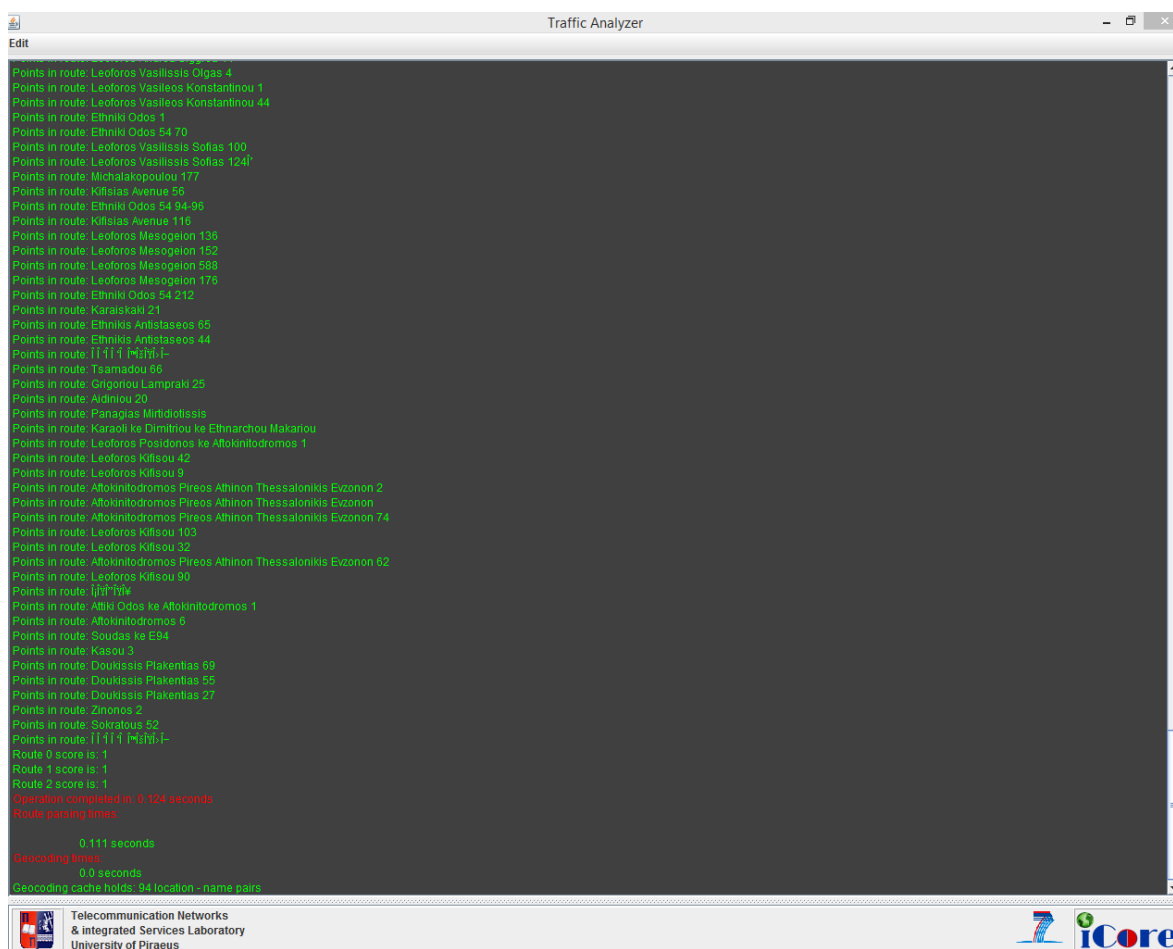


Figure 99: Output of Traffic Analyzer CVO – second iteration

The performance of the Traffic Analyzer CVO for different routes is presented in Figure 100. Routes of 14, 21, 25 and 36 km were tested in three cases. The first case is the performance with no geocoding cache (represented by the blue line). The second case is the performance in the case only geocoding cache is used (represented by the red line) while in the third case the CVO is operating normally (geocoding as well as using cache where possible, represented by the green line).

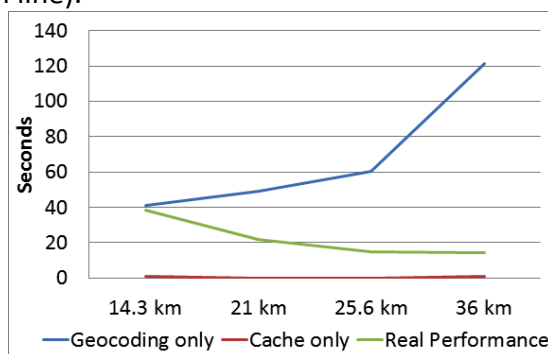


Figure 100: Traffic Analyzer CVO performance chart



### 3.6.2. Test case #2 – Derivation of real world information (Hadoop part in iCore platform)

#### 3.6.2.1. Description

In the iCore platform, Hadoop is used for the efficient processing of rough data that are collected from the Smart Santander platform in the Smart Santander CVO and the derivation of meaningful statistics that will be further exploited from the Traffic Analyzer CVO.

#### 3.6.2.2. Entities involved

- Hadoop mechanism (at the Service level)
- Smart Santander CVO
- Traffic Analyzer CVO

#### 3.6.2.3. Related requirements

Table 26 presents the functional requirements (iCore Deliverable 2.1) that are related to this test case. The developed mechanism should comply with them and furthermore, can be exploited for the fulfilment of some of them.

**Table 26: Requirements related to Smart Tour in the City trial test case #2**

Identifier	Name	Description
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe semantically not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-12	The iCore system shall be able to aggregate and infer relevant situational dimensions (e.g. time, place, preference of users) to describe the current situation and anticipate changes to it.	The iCore system is envisioned to describe semantically not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-13	The iCore system shall be able to infer (reason) on user intentions/behaviour in order to support current and future services and service requests.	Importance of storing relationships between objects, people, applications. The iCore system must be adaptable to User intentions/behaviour for dynamic service composition.
SL-F-15	The iCore system shall maintain and be able to store, classify, rank, and annotate a categorisation of different situations based on user intentions/ behaviour per situation for future use.	It is an essential element for the iCore system to 'grow' intelligence and foster reuse of iCore objects in multiple services and across multiple situations.

### 3.6.2.4. Pre-conditions

The existence of rough data from the Smart Santander CVO can be identified as a pre-condition for this test case.

### 3.6.2.5. Execution steps

- 1: Received rough data
- 2: Large-scale processing
- 3: Derivation/Update of statistics
- 4: Send to the traffic analyzer CVO (when requested)

### 3.6.2.6. Success criteria/ Expected output

The production and maintenance of updated statistics is the main objective of this process.

### 3.6.2.7. Test Case Result

Figure 101 presents the evaluation results of the Hadoop mechanism. Different volumes of rough data were examined in order to evaluate the necessary processing time for the derivation of the statistics.

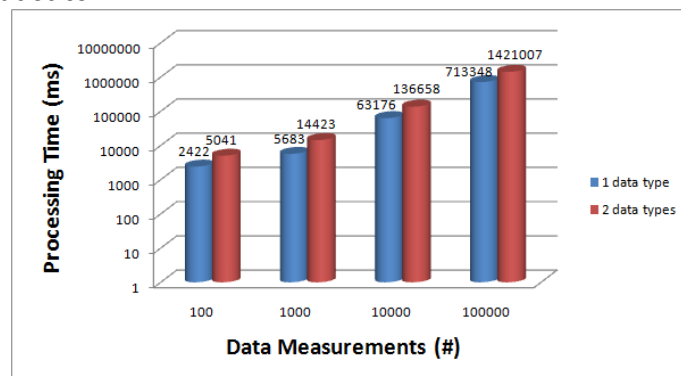


Figure 101: Hadoop processing time for different volume of data measurements

In addition, the experiments included the evaluation of two cases. In the first one, only one data type was specified to be processed (e.g. traffic intensity) from the overall available measurements, while in the second case, two data types were examined (e.g. traffic intensity and luminosity). As depicted, the needed time increases when either the available measurements or the number of the examined data types increase. It should be noted that the presented results are indicative and can be optimized by adding more active nodes in the Hadoop clustering (for more efficient parallel processing).

## 3.6.3. Test case #3 – Dynamic Composition of Virtual Objects

### 3.6.3.1. Description

In the **Smart tour in the city** trial, numerous VOs are involved and exploited as sources of information. The CVO Composition Engine (CCE) is responsible for deciding the optimal composition of these VOs according to the defined requirements. For this purpose, the mechanism evaluates the available VOs, taking into account their features (e.g. power consumption, performance, quality levels). Moreover, it allows the approximation of

requested functionalities from similar ones that can provide what is needed in an acceptable way.

### 3.6.3.2. Entities involved

- Approximation and Reuse Opportunity Detection
- CVO Composition Engine
- VOs

### 3.6.3.3. Related requirements

Table 27 presents the functional requirements (iCore Deliverable 2.1) that are related to this test case. The developed mechanism should comply with them and furthermore, can be exploited for the fulfilment of some of them.

**Table 27: Requirements related to Smart Tour in the City trial test case #3**

Identifier	Name	Description
SL-F-17.	The iCore system shall be able to identify an approximate CVO quickly to match requested services as best as possible.	It is not always possible to find exact services for a user all the time. Thus, if an exact CVO which was supposed to provide a particular service is not available, we have to identify an approximate service. The approximation is done based on the availability of alternatives and possibilities. Depending on the situation, there might be a threshold to decide and select an approximate service (i.e. alternative service).
CVO-F-3.	The CVO functionality shall be able to autonomically select and integrate the correct VOs that describe and match the current / potential service requirements.	A CVO must be able to evaluate the service function offered by VOs and reconfigure itself if necessary by selecting / maintaining the best VOs.
CVO-F-16.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
VO-F-3.	A VO shall enable the representation of proximity. It must also represent the relationship of proximity between	A VO must be able to publish its natural relation with another VO and represent these as two related RWOs.

multiple RWOs.

### 3.6.3.4. Pre-conditions

The Approximation and Reuse Opportunity Detection mechanism that lies in the CVO Management Unit, searches for already existing CVOs that can fulfil the requested service. If this is not possible, then it triggers the CCE and requests for an optimal selection of VOs, in order to compose the most suitable CVO from scratch.

### 3.6.3.5. Execution steps

- 1: Triggering event from the Approximation and Reuse Opportunity Detection mechanism
- 2: Evaluation of available VOs and requirements
- 3: Selection of the VOs that can compose the optimal CVO

### 3.6.3.6. Success criteria/ Expected output

The output of this process is a set of VOs that may compose the most suitable CVO according to the defined requirements.

### 3.6.3.7. Test Case Result

Figure 102 and Figure 103 present the evaluation results for the CVO Composition Engine. The performed experiments include scenarios with different numbers of available VOs and requested functions. Figure 102 depicts the CCE processing time for cases where the available VOs increase from 10 up to 1000. Even in the case of 1000 VOs the corresponding time can be considered acceptable for online operations, as it remains in the range of a few dozens of milliseconds. Figure 103 presents the CCE processing times for scenarios where the number of requested functions increases. Similarly with previous case, the findings validate that CCE is a time-efficient mechanism.

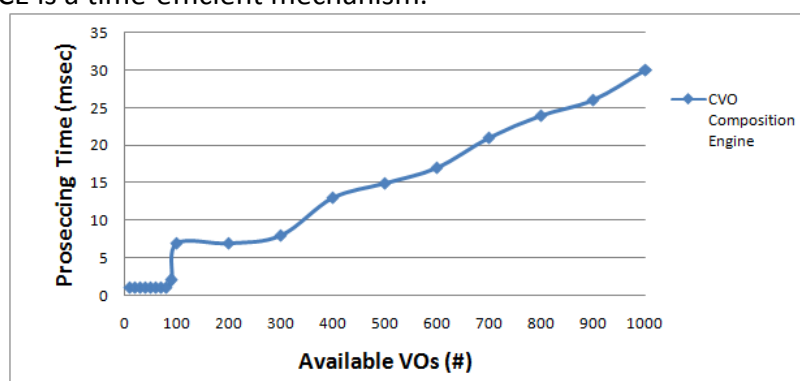


Figure 102: CCE processing time for different numbers of available VOs in the case of 10 requested functions

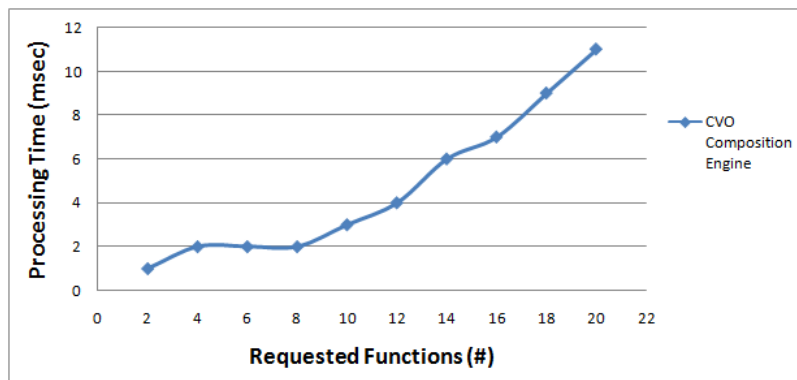


Figure 103: CCE processing time for different numbers of requested functions in the case of 100 available VOs

### 3.6.4. Test case #4 – Smart Santander acquisition 3 – Dynamic Composition of data through FI-WARE GEs – (VOs and CVOs) Virtual Objects

#### 3.6.4.1. Description

The iCore platform has been successfully integrated with the Smart Santander FI-WARE GEs in order to enable the acquisition of data from RWOs that are deployed in the city of Santander in Spain. For this reason we have deployed a set of different VOs that in the VO Back End have been optimized so as to use the FIWARE GEs and specifically the NGSI10 Device Backend so as to retrieve the uploaded real-time measurements in the global Context Broker GE instance, as well as the historical data measurements through the global instance of the BigData Analysis GE. Each VO has been registered in the public instance of the iCore VO Registry (Figure 104) and it is reachable from external entities, which exploit / use the VO Registry API. The VOs are deployed in the public instance of the iCore VO Container (Figure 105) and they are included in the CVO Smart Santander Composition (Figure 106), which together with the rest of the trial CVOs, is hosted in the public instance of the CVO Container (Figure 107). In this test case we elaborate the performance of the VOs in terms of the average time that is required for the retrieval of the real-time and the historical data measurements. We have chosen an indicative range of twenty (20) different VOs that are associated with RWOs that are deployed in Smart Santander, whereas we have performed nine iterations ( $t_0 - t_8$ ) so as to have a reliable sample of time data.

VO Registry Administrator Panel - <http://83.212.238.249:8092/vo\_registry>

File Tools Semantic Repository SPARQL Endpoint VO Management Security Control

Fetch VO registry [Performance Time = 15 msec] [VOs in the list = 82]

No	VIRTUAL OBJECT	VO Status	ICT Object	Non-ICT Object
74	http://83.212.238.249:8019/OUTSMART.NODE_3500	AVAILABLE	http://83.212.238.249:8019/OUTSMART.NODE_3500/ict	http://83.212.238.249:8019/OUTSMART.NODE_3500/ictnon_ictsSmart_Santan...
72	http://83.212.238.249:8019/OUTSMART.NODE_3501	AVAILABLE	http://83.212.238.249:8019/OUTSMART.NODE_3501/ict	http://83.212.238.249:8019/OUTSMART.NODE_3501/ictnon_ictsSmart_Santan...
47	http://83.212.238.249:8019/OUTSMART.NODE_3504	AVAILABLE	http://83.212.238.249:8019/OUTSMART.NODE_3504/ict	http://83.212.238.249:8019/OUTSMART.NODE_3504/ictnon_ictsSmart_Santan...
7	http://83.212.238.249:8019/OUTSMART.NODE_3505	AVAILABLE	http://83.212.238.249:8019/OUTSMART.NODE_3505/ict	http://83.212.238.249:8019/OUTSMART.NODE_3505/ictnon_ictsSmart_Santan...
42	http://83.212.238.249:8019/OUTSMART.NODE_3506	AVAILABLE	http://83.212.238.249:8019/OUTSMART.NODE_3506/ict	http://83.212.238.249:8019/OUTSMART.NODE_3506/ictnon_ictsSmart_Santan...
68	http://83.212.238.249:8019/OUTSMART.NODE_3507	AVAILABLE	http://83.212.238.249:8019/OUTSMART.NODE_3507/ict	http://83.212.238.249:8019/OUTSMART.NODE_3507/ictnon_ictsSmart_Santan...
32	http://83.212.238.249:8019/OUTSMART.NODE_3508	AVAILABLE	http://83.212.238.249:8019/OUTSMART.NODE_3508/ict	http://83.212.238.249:8019/OUTSMART.NODE_3508/ictnon_ictsSmart_Santan...
41	http://83.212.238.249:8019/OUTSMART.NODE_3509	AVAILABLE	http://83.212.238.249:8019/OUTSMART.NODE_3509/ict	http://83.212.238.249:8019/OUTSMART.NODE_3509/ictnon_ictsSmart_Santan...
79	http://83.212.238.249:8019/OUTSMART.NODE_3510	AVAILABLE	http://83.212.238.249:8019/OUTSMART.NODE_3510/ict	http://83.212.238.249:8019/OUTSMART.NODE_3510/ictnon_ictsSmart_Santan...
24	http://83.212.238.249:8019/um_smartsantander_testbed_3300	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3300/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3300/ictnon_ictsSm...
58	http://83.212.238.249:8019/um_smartsantander_testbed_3301	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3301/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3301/ictnon_ictsSm...
69	http://83.212.238.249:8019/um_smartsantander_testbed_3302	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3302/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3302/ictnon_ictsSm...
37	http://83.212.238.249:8019/um_smartsantander_testbed_3303	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3303/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3303/ictnon_ictsSm...
44	http://83.212.238.249:8019/um_smartsantander_testbed_3304	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3304/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3304/ictnon_ictsSm...
33	http://83.212.238.249:8019/um_smartsantander_testbed_3305	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3305/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3305/ictnon_ictsSm...
12	http://83.212.238.249:8019/um_smartsantander_testbed_3306	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3306/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3306/ictnon_ictsSm...
46	http://83.212.238.249:8019/um_smartsantander_testbed_3307	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3307/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3307/ictnon_ictsSm...
55	http://83.212.238.249:8019/um_smartsantander_testbed_3308	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3308/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3308/ictnon_ictsSm...
10	http://83.212.238.249:8019/um_smartsantander_testbed_3309	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3309/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3309/ictnon_ictsSm...
49	http://83.212.238.249:8019/um_smartsantander_testbed_3310	AVAILABLE	http://83.212.238.249:8019/um_smartsantander_testbed_3310/ict	http://83.212.238.249:8019/um_smartsantander_testbed_3310/ictnon_ictsSm...

Telecommunication Networks & Services Laboratory University of Piraeus




Figure 104: iCore VO Registry public instance

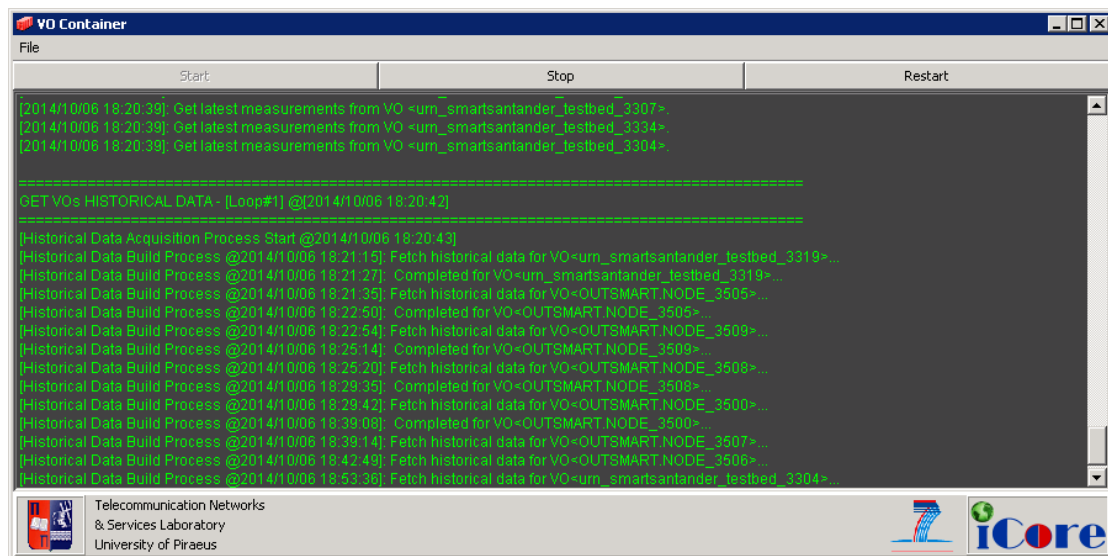


Figure 105: Pubic instance of iCore VO Container for Smart Tour in the city Trial deployment

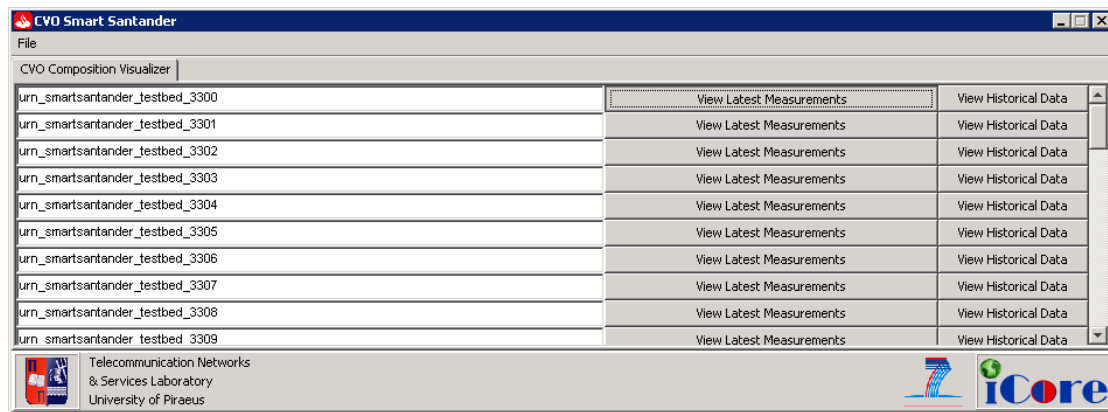


Figure 106: Smart Santander CVO UI Instance

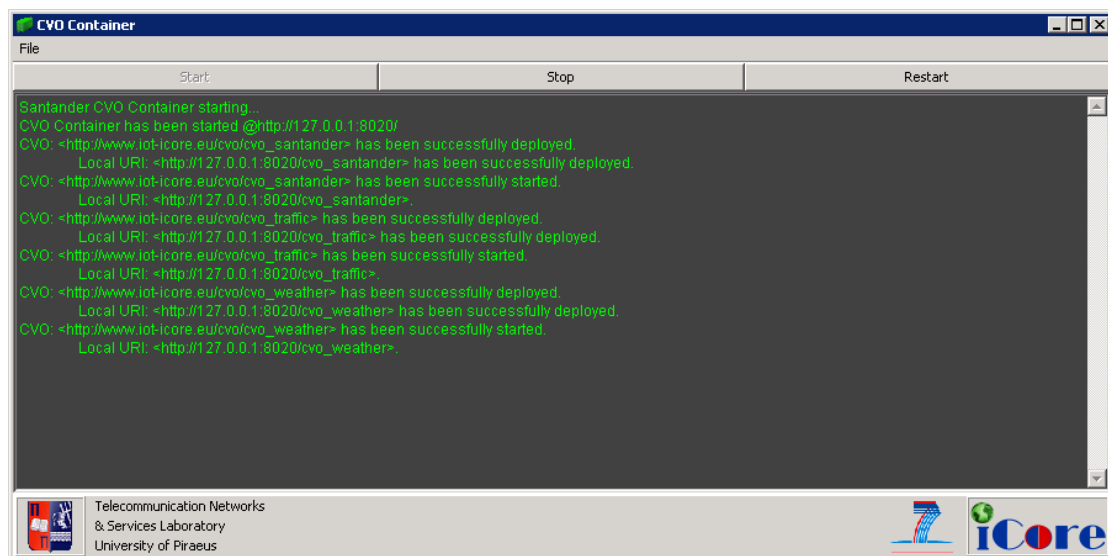


Figure 107: Pubic instance of iCore CVO Container for Smart Tour in the city Trial deployment

### 3.6.4.2. Entities involved

- VOs (VO Front End: RESTful WS compatible and VO Back End: FIWARE GEs compatible);
- VO Container: hosts the deployed VOs;
- CVO Container: hosts the deployed CVOs, in this case the Smart Santander CVO;
- Smart Santander CVO: collects, share / distribute the real-time and historical data measurements to third party entities, such as the Hadoop mechanism.

### 3.6.4.3. Related requirements

Table 28 presents the functional requirements (iCore Deliverable 2.1) that are related to this test case. The deployed CVO and VO level components should comply with them and furthermore, can be exploited for the fulfilment of some of them.



**Table 28: Requirements related to Smart Tour in the City trial test case #4**

Identifier	Name	Description
CVO-F-1	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-2	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters).
CVO-F-14	A CVO shall be enabled with functionality to support service recovery after an internal or external failure or due to mobility.	Composition rules are needed in order to insure a "repair" capability for CVOs (e.g. the replacing of VO/CVO segments with an equivalent VO/CVO when part of the execution chain is compromised).
CVO-F-15	CVOs shall be enabled to adapt to the Mobility of the iCore system (or part of it) or user.	This requirement deals with the changes that are manifested because of mobility of actors and devices of the iCore system.
VO-F-1	The VO Registry shall be able to maintain and record the links with each RWO (i.e. send notification (signalling) when coupled and when decoupled).	Handle to the real world is maintained in a best possible way. Awareness regarding the RWO.
VO-F-2	The VO shall be able to name, type (Metadata) and address each RWOs that becomes available (i.e. at the point of discovery).	RWOs are appropriately named and addressed within their respective proximity domains.

VO-F-6	A VO should describe the available methods/services for interaction with other VOs.	A sensor VO could provide the services/methods of single values (i.e. request-response), or a stream of values (i.e. subscribe-stream), or send a message when a threshold is reached (i.e. subscribe-threshold). Analogously an actuator VO could provide methods/services for example, an on/off or on/off-duration actuation response.
VO-F-7	A VO should be able to know the current location (i.e. Be location aware) of the RWO it is representing.	The location of the RWO should be abstracted at the VO level.
VO-F-9	A VO should notify CVO registries of its current status, availability and reliability for re-use.	The VO discovery interface between the VO and CVO levels. A VO should have capability to update/modify itself and then notify the CVO for composition.
VO-F-10	A VO should be able to represent a single RWO or many RWOs.	A VO may be formed out of multiple RWOs.
VO-F-12	A VO shall provide a semantic description of a RWO including its control and configuration interfaces.	Description of features provided by the RWO, how to access these features (e.g. call a web service, send an event, etc.) and how to configure parameters. The relationship of the RWO and its resources should be described at the VO level.

#### 3.6.4.4. Pre-conditions

The VOs, that have been deployed so as to retrieve real-time and historical data from the RWOs that belongs to Smart Santander infrastructure, are hosted in the public instance of the iCore VO Container. The VOs through their customized FIWARE GEs compatible back-end are programmed to retrieve the measurement data, whereas through their VO Front-End, pass the data to the Smart Santander CVO that is hosted in the CVO Container.

### 3.6.4.5. Execution steps

- 1 Start the public instance of the iCore VO Container.
- 2 Start the public instance of the CVO Container.
- 3 The VOs that are deployed in the VO Container start automatically their time scheduled execution for the retrieval of real-time and historical data.
- 4 The Smart Santander CVO retrieves the data from the deployed VOs and interacts with Hadoop platform by passing large-scale historical data for their processing.

### 3.6.4.6. Success criteria/ Expected output

The output of the above processes is a large-scale set of measurement data that lies on the real-time and historical measurements of Traffic, Temperature, Luminosity and crow presence in the smart city of Santander, in Spain.

### 3.6.4.7. Test Case Result

Figure 108 presents the average values for the time that is required for the retrieval of real-time and historical data from the deployed sensors in the city of Smart Santander. In general, the average time for the retrieval of the real-time data ranges around 1,20 seconds, whereas the average time for the retrieval of the historical data ranges around 80,5 seconds.

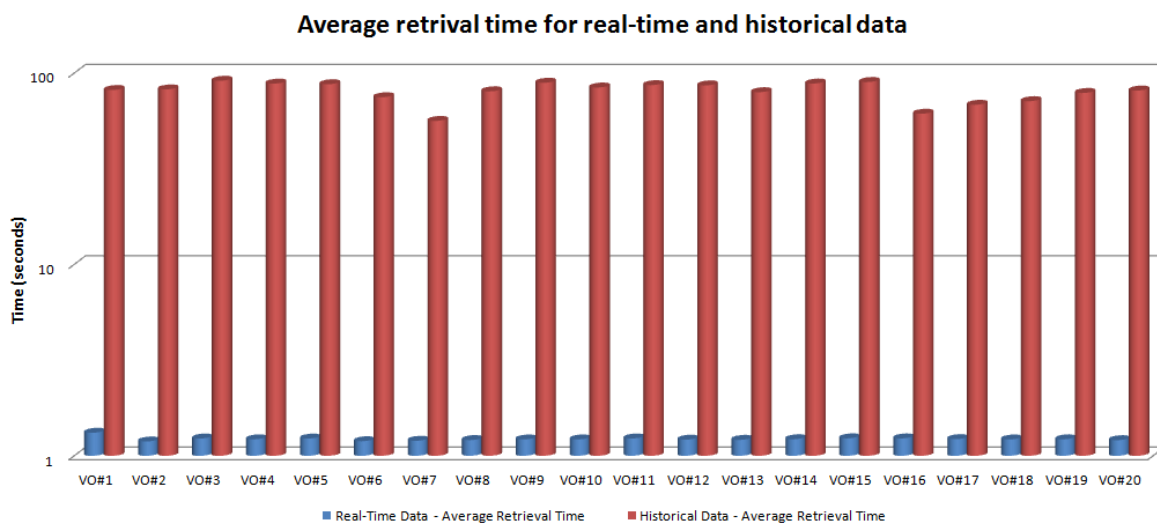


Figure 108: Average retrieval time for real-time and historical data

## 3.7 Smart Hospital trial

### 3.7.1. Test case #1 – Localization of assets

#### 3.7.1.1. Description

A trial user starts the mobile application and specifies what he wants to do in English. The iCore system translates the request into a machine readable language and starts its processing. Among others the localization request is the main one. An asset can be

discovered by connecting it directly or its location can be predicted. Upon finalizing the localization procedure the user is informed about the asset location in the hospital.

### 3.7.1.2. Entities involved

Service request, RWK, CVO registry, CVO container, VO registry.

### 3.7.1.3. Related requirements

Table 29 presents the relevant functional requirements for Smart hospital trial test case #1 taking into account the requirements listed in D2.1

Table 29: Relevant requirements for Smart Hospital test case #1

Identifier	Name	Description
SL-F-9.	The iCore system shall enable the proper and efficient composition of iCore objects that fulfil the query from the service level.	The iCore system shall be able to configure/monitor/select iCore objects on the basis of what the services requested by external actors need.
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-12	The iCore system shall be able to aggregate and infer relevant situational dimensions (e.g. time, place, preference of users) to describe the current situation and anticipate changes to it.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-15	The iCore system shall maintain and be able to store, classify, rank, and annotate a categorisation of	It is an essential element for the iCore system to 'grow' intelligence and foster reuse of iCore objects in multiple services and across multiple situations.

	different situations based on user intentions/behaviour per situation for future use	
SL-F-17	The iCore system shall be able to identify an approximate CVO quickly to match requested services as best as possible.	It is not always possible to find exact services for a user all the time. Thus, if an exact CVO which was supposed to provide a particular service is not available, we have to identify an approximate service. The approximation is done based on the availability of alternatives and possibilities. Depending on the situation, there might be a threshold to decide and select an approximate service (i.e. alternative service).
CVO-F-1.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
CVO-F-2.	A CVO shall have ability to hold (store) many VO's based on a common situation and service description.	A CVO aggregates services from many VOs and must store information about these VOs in correlation with the service logic requirement and the specific situation (situation and request parameters).
CVO-F-3	The CVO functionality shall be able to autonomically select and integrate the correct VOs that describe and match the current / potential service requirements.	A CVO must be able to evaluate the service function offered by VOs and reconfigure itself if necessary by selecting / maintaining the best VOs.
CVO-F-5	The CVO registry shall be able to obtain Service Logic requirements either by a query (when they are stored somewhere) or by a	The CVO registry must be able to interface with the upper service level and be able to receive service logic requirements (request parameters).

	trigger event (when they are constructed on the fly).	
CVO-F-6	The CVO registry shall be able to address and name stored VOs and CVOs.	Stored VOs and CVOs must have an address and name in order to be retrieved, e.g. as web services.
CVO-F-9	The CVO shall be enabled with an auto construction of a “scheduler”.	A CVO must somehow evolve a schedule to use each VO (and its functionalities) in a sequence (some in parallel) in order to achieve a required service.
CVO-F-11	A CVO shall be created based on a template that is produced at design time and intends to define the CVO and determine his capabilities.	Templates are necessary for a semi-automatic generation of CVOs from data acquisition and events processing.
CVO-F-12	A CVO shall be validated before the execution, based on the received, complete and accepted non-conflict situation with a clear lifespan for execution.	Validation is necessary in order to add in valid CVOs against SLA rules (we need to insure that the decomposition of a SLA will result in a potential CVO discovery) and to insure that CVO execution will not generate bad results (e.g. circular loops or system crashes).
VO-F-1	The VO Registry shall be able to maintain and record the links with each RWO (i.e. send notification (signalling) when coupled and when decoupled).	Handle to the real world is maintained in a best possible way. Awareness regarding the RWO.
VO-F-2	The VO shall be able to name, type (Metadata) and address each RWOs that becomes	RWOs are appropriately named and addressed within their respective proximity domains.

	available (i.e. at the point of discovery).	
VO-F-5	The VO functionality shall be able to dynamically describe RWOs and match (tag) the correct RWOs to current CVO / Service / Application requirements.	New RWOs can be added to the iCore system during run-time. Once VO functionalities tag with new RWOs then it also becomes a new VO and the old VO may survive for a certain time.
VO-F-7	A VO shall be able to know the current location (i.e. Be location aware) of the RWO it is representing.	The location of the RWO shall be abstracted at the VO level.
VO-F-10	A VO shall be able to represent a single RWO or many RWOs.	A VO may be formed out of multiple RWOs.
VO-F-12	A VO shall provide a semantic description of a RWO including its control and configuration interfaces.	Description of features provided by the RWO, how to access these features (e.g. call a web service, send an event, etc.) and how to configure parameters. The relationship of the RWO and its resources shall be described at the VO level.

#### 3.7.1.4. Pre-conditions

To ensure the correct operation of the entire system, the mobile phone has to be on, the iCore system should be ready for new request. The assets located in the hospital can be either on or in sleep mode.

#### 3.7.1.5. Execution steps

- Start mobile application and choose what to do with an asset: locate, trace, schedule maintenance, enable power saving policy
- Select asset of your interest
- Follow the instructions of mobile application
- Get the results (it can be location of an asset, confirmed date of its maintenance, confirmation on the enabled power saving policy)



### 3.7.1.6. Success criteria/ Expected output

The mobile application and its graphical user interface must be as simple as possible and must hide all technical complexity from the hospital personnel.

The wireless tags connected to the medical assets and wireless technology involved in the localization algorithm must meet safety requirements in terms of electromagnetic compatibility.

Another success criterion is about the location prediction of the assets in sleep mode – it has to be as precise as possible.

### 3.7.1.7. Test case results

#### Android Mobile Application

The medical stuff in the Hospital of Santa Chiara in Trento is assumed to exploit the trial after its deployment. Due to the lack of deep technical background of the personnel, the trial developers have designed an Android mobile application to make the iCore solution as ‘friendly’ as possible for the medical personnel.

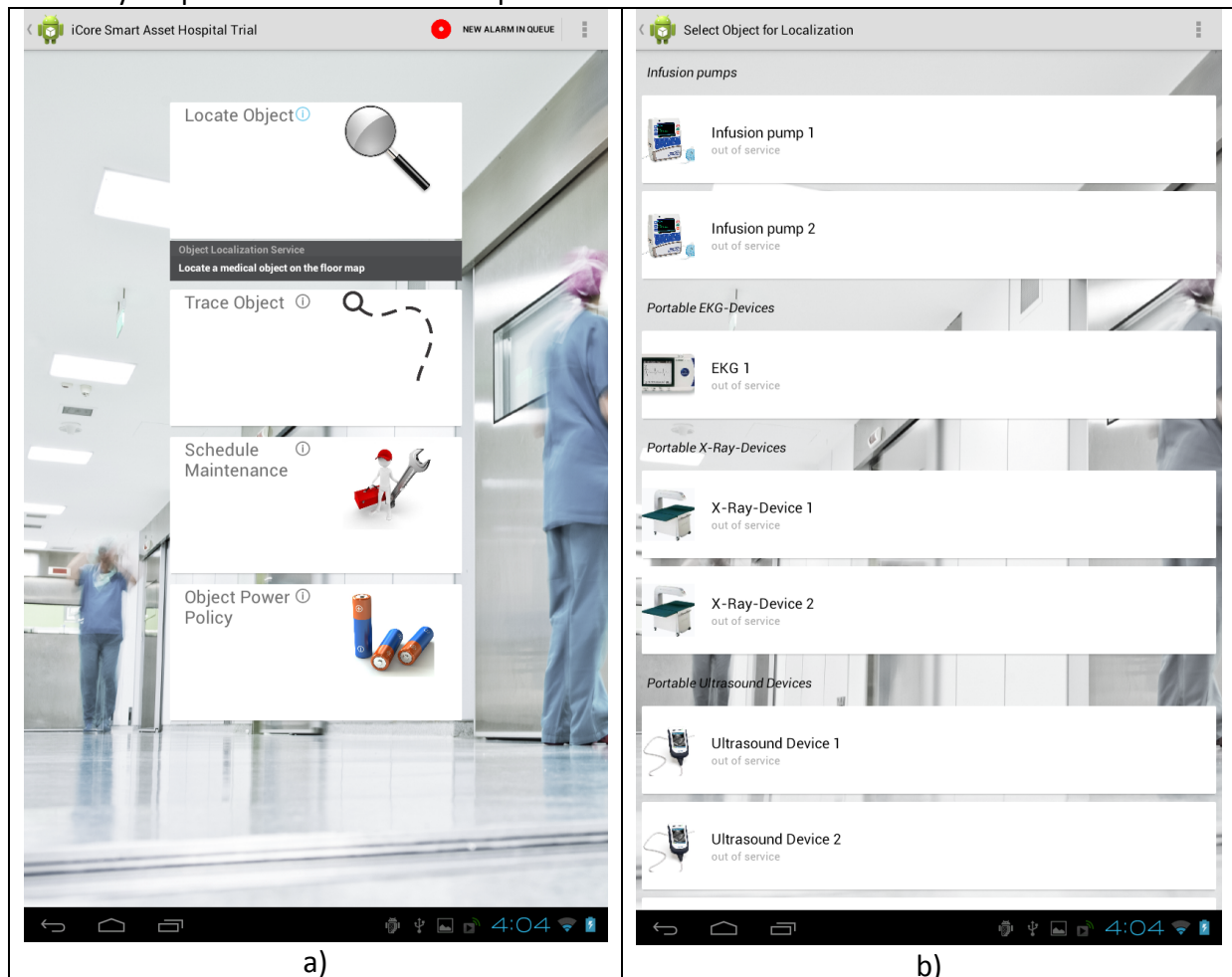


Figure 109: An example of Android mobile application (screenshots).

Figure 109 demonstrates two examples of developed mobile application for Android-based smart phones. In particular, Figure 109(a) allows a user to choose an activity among the available options: trace or locate an object, schedule its maintenance or set up its power

policy. Figure 109(b) features a screenshot showing the maintenance status of the objects. Using the menu in the right upper corner of the mobile application the user can schedule or setup maintenance for the objects.

### Evaluation of prediction model

In this section we describe the evaluation of AR model presented in Section 2.7.2.2

To test the AR prediction model, the AR model is trained with 1 entire day object location data and the parameters of AR are computed. The object location is then predicted based on the model.

**Table 30: Accuracy of AR model for different time granularity**

Time	Correct predicted locations	Total Locations	Accuracy
4 days complete data	4782	5760	83%
0hr to 5hr	1058	1440	73%
6hr to 11hr	1220	1440	85%
12hr to 17hr	1252	1440	87%
18hr to 23hr	82	1435	5%

Table 30, shows the accuracy of the prediction model. The first row indicates when one entire day data is used to train and obtain model parameters the accuracy of prediction is 83%. The rows 2, 3, 4, 5 shows when the data is split every 6 hours, the prediction accuracy is 73%, 85%, 87%, 5% respectively. Splitting data for AR model can increase prediction accuracy only when the object is actively moving in the data split duration. For example, the accuracy for data 12hr to 17hr is 87% whereas 18hr to 23hr is only 5% this is because the object was actively moving during the 12hr to 17hr and more stagnant during the 18hr to 23hr. (As the data was collected in an office environment for our experiments, the objects are not moved after 18hr).

As it is noticed in the document on the interview of the hospital personnel, their major concern was associated with the certification of hardware used in the Smart Hospital trial. The following parts specify how the sensor nodes and operation frequencies are in line with the standardization. The experiments and, in particular, the data collection for the evaluation of AR model are conducted with respect to these formal regulations.

### ETSI certification

Standardization is a key factor for single market and is needed for coexistence of various technologies sharing the same physical medium. Use of wireless technologies in different environment's needs to accord with actual regulation and legislation. ETSI, the European Telecommunications Standards Institute, produces standards for Information and Communications Technologies (ICT) that are used by national governments to enforce regulations. The standard "EN 301 489-1 (v1.8.1) (Council Directive 2004/108/EC on

Electromagnetic Compatibility) Electromagnetic compatibility and Radio spectrum Matters (ERM); Electromagnetic Compatibility (EMC) standard for radio equipment and services; Part 1: Common technical requirements” needs to be fulfilled to operate with radio equipment. Manufacturer, that produces or import products inside the European Union have to certify their products and sign it with the CE logo of the “Communautés Européennes”. Therefore, it is needed to fulfil all terms of references. The manufacturer himself asserts conformance based on its compliance assessment.

### **Electromagnetic compatibility requirements**

Indoor localization in ICore will make use of radiofrequencies (RF). In particular, the used technologies will be:

- ZIGPOS-RTLS: based on the IEEE 802.15.4 standard.

The usage of radio spectrum is regulated at the national, European and international level. All the technologies to be used in ICore for indoor localization purposes operate in the so-called ISM (Industrial, Scientific and Medical) bands.

The ISM bands are portions of the radio spectrum reserved internationally for the use of RF energy for industrial, scientific and medical purposes. Large portions of the ISM bands (and, in particular, those used within ICore) are unlicensed, i.e., no license is required to operate a device transmitting in such a band. This implies, conversely, that devices may be operating in a harsh environment characterised by high levels of interference.

The ISM bands are defined by the ITU-R in 5.138, 5.150, and 5.280 of the Radio Regulations [60]. Individual countries' use of the bands designated in these sections may differ due to variations in national radio regulations. The bands in which the ICore enabling devices will transmit are available for unlicensed use in all countries where ICore pilots will be deployed. In the EU, low power wireless devices are generally referred to as short-range devices (SRD). The allocation of frequency bands and their use in the EU are based on recommendations by the Electronic Communications Committee (ECC), which is part of the European Conference of Postal and Telecommunication Administration (CEPT). The ECC document covering SRD is ERC/REC 70-03. The 45 member countries of the CEPT must then adopt these recommendations into law for them to be binding, so there are occasionally differences between the member countries. No significant difference has been identified for the purposes of the ICore piloting activities.

ECC recommendation defines 13 different types of SRD applications. The SRD applications relevant for ICore are n.1, “Non-specific Short Range Devices” and n. 3, “Local Area Networks, RLANs and HIPERLANs”. The ECC recommendation 70-03 defines both the maximum transmit power and limits to the duty cycle and the bandwidth of the transmitter for each allocated frequency band. For example a limit of -10dB on the ERP (Effective Radiated Power) is required when operating in the band 2400MHz-2483.5 MHz, which is one of the bands in which ICore devices will operate.

Electromagnetic Compatibility (EMC) refers to the ability of a device to operate properly in its intended environment without producing excessive interference to other devices. All electronic devices must meet certain regulations regarding EMC. These regulations cover both intentional (for example, transmission signals) and non-intentional (electrical noise) radiation.

Other potential regulatory issues are induced radiation and RF exposure. Induced radiation refers to how well a device withstands unintentional radiation from an external source (e.g. high voltage line or microwave oven). RF exposure regulations, on the other hand, determine if the device emits radiation that is harmful to human beings. This is normally only a concern for high-power transmission devices, but there have been some concerns (yet to be proven) that long-term exposure to even low-levels of electromagnetic radiation could potentially result in cancer and other health problems. For devices that may be positioned within 20 cm of a human body, SAR (Specific Absorption Rate) testing is required to ensure radiation levels are below a certain limit. In Europe, compliance in terms of RF exposure for the kind of devices used within iCore is standardized as CENELEC EN 62479:2010, which has been made part of 2006/95/EC directive.

The procedure required to bring wireless equipment to the EU market is outlined in the Directive 199/5/EC of the European Parliament and of the Council (R&TTE - Radio and Telecommunications Terminal Equipment directive). The R&TTE directive is based on self-declaration, that is, the manufacturer who supplies wireless equipment to the market declares that the product satisfies the legal requirements. Basically, the entity that places the equipment on the market is responsible for its compliance.

The CE marking is a mandatory conformity marking for some categories of products sold within the European Economic Area (EEA). It consists of the CE-Logo and, if applicable, the four digit identification number of the notified body involved in the conformity assessment procedure. The CE marking states that the product is assessed before being placed on the market and meets EU safety, health and environmental protection requirements. Devices to be used in iCore should be required to be CE marked.

### Medical devices (or operation in medical domain)

According to the **Directive 2007/47/EC** amending Council Directive 93/42/EEC “medical device means any instrument, apparatus, appliance, software, material or other article, whether used alone or in combination, including the software intended by its manufacturer to be used specifically for diagnostic and/or therapeutic purposes and necessary for its proper application, intended by the manufacturer to be used for human beings for the purpose of:

- Diagnosis, prevention, monitoring, treatment or alleviation of disease,
- Diagnosis, monitoring, treatment, alleviation of or compensation for an injury or handicap,
- Investigation, replacement or modification of the anatomy or of a physiological process,
- Control of conception,

and which does not achieve its principal intended action in or on the human body by pharmacological, immunological or metabolic means, but which may be assisted in its function by such means”.

According to the same law, “devices shall be divided into Classes I, IIa, IIb and III. Classification shall be carried out in accordance with Annex IX”.

### 3.8 Smart Theme Park trial

#### 3.8.1. Test case #1 The highlights data corresponding to a round of ride could be precisely retrieved.

##### 3.8.1.1. Description

It is important to retrieve the video and audio data captured during a round of ride from, neither more nor less. As the data are transmitted wirelessly to the backend storage server and saved in the format of discrete video clips (with audio track), the starting point and ending point of the data corresponding to a ride may be across video files. We need to cut out the desirable length of video data from single files. The precision of the cut relies on two factors, the precision of time sensing and the operation of video / audio editing (depending on video frame rate, audio sample frequency, video and audio compression algorithms).

##### 3.8.1.2. Entities involved

Video recording VO, audio recording VO, Time sensing VO, identity mapping VO, highlights retrieval CVO

##### 3.8.1.3. Related requirements

Table 31: Requirements related to Smart Theme Park trial test case #1

Identifier	Name	Description
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-12	The iCore system shall be able to <b>aggregate</b> and infer relevant situational dimensions (e.g. time, place, preference of users) to describe the current situation and	See above

	anticipate changes to it.	
CVO-F-1.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
VO-F-3	A VO shall enable the representation of proximity. It must also represent the relationship of proximity between multiple RWOs.	A VO must be able to publish its natural relation with another VO and represent these as two related RWOs.
VO-F-10	A VO shall be able to represent a single RWO or many RWOs.	A VO may be formed out of multiple RWOs.

#### 3.8.1.4. Pre-conditions

- The mounted camera, develop board is on.
- The storage server deployed on station is working.
- The video capturing VO, audio capturing VO and time sensing VO is up and running.
- The highlights retrieval CVO is up and running, waiting for input.
- A customer is issued a wrist band and seated in the ride.

#### 3.8.1.5. Execution steps

- Scan customer wrist band and camera tag to establish the identity mapping.
- Start the ride and record the starting time.
- When the ride returns back, record the ending time and then search for the personal highlights (video clip) with wrist band EPC.

#### 3.8.1.6. Success criteria/ Expected output

The time difference between the starting and ending points of retrieved data and recorded time should be no more than 1 second, which is about 25 frames in this trial.

#### 3.8.1.7. Test Case Result

We run the test case 10 time and list the results in the following table. The result is specified in the number of frames less(-) or more (+) than the recorded time.

	1	2	3	4	5	6	7	8	9	10
StartDiff	+13	+19	+2	-20	+19	+8	-16	-11	-9	-7

EndDiff	-10	+8	-9	-14	-4	-9	+8	+7	+7	+9
---------	-----	----	----	-----	----	----	----	----	----	----

### 3.8.2. Test case #2 The highlights data of a customer should be correctly retrieved.

#### 3.8.2.1. Description

As a ride will take multiple riders in a round, we need to correctly retrieve the highlight data of each rider to generate the output souvenir.

#### 3.8.2.2. Related requirements

Table 32: Requirements related to Smart Theme Park trial test case #2

Identifier	Name	Description
SL-F-11	Situations and user properties shall be appropriately modelled for representation in an iCore system.	The iCore system is envisioned to describe <b>semantically</b> not only iCore objects but also situations, and service request trends and actor intentions.
SL-F-12	The iCore system shall be able to <b>aggregate</b> and infer relevant situational dimensions (e.g. time, place, preference of users) to describe the current situation and anticipate changes to it.	See above
CVO-F-1.	A CVO shall be able to integrate more than one VO.	A CVO is a cognitive mash-up of semantically interoperable VOs and uses the services offered by VOs in order to satisfy Service Logic requirements. Therefore, a CVO must contain at least one VO.
VO-F-3	A VO shall enable the representation of proximity. It must also represent the relationship of	A VO must be able to publish its natural relation with another VO and represent these as two related RWOs.



	proximity between multiple RWOs.	
VO-F-10	A VO shall be able to represent a single RWO or many RWOs.	A VO may be formed out of multiple RWOs.

### 3.8.2.3. Pre-conditions

- The mounted camera, develop board is on.
- The storage server deployed on station is working.
- The video capturing VO, audio capturing VO is up and running.
- The highlights retrieval CVO is up and running, waiting for input.
- Five customers are issued wrist bands and seated in the ride.

### 3.8.2.4. Execution steps

- Scan customer wrist bands and camera tags to establish the identity mapping.
- Start the ride.
- When the ride returns back, search for the personal highlights (video clip) with wrist band EPC.

### 3.8.2.5. Success criteria/ Expected output

Highlight data of each of the five customers should be correctly retrieved. In other words, the retrieved highlight data should contain images and sound of the desired rider.

### 3.8.2.6. Test Case Result

The result of this test case is shown in the following table.

Rider EPC	0001	0002	0003	0004	0005
Result(Yes/No)	Y	Y	Y	Y	Y

## 4 Non-technical validation

This section contains the results of the iCore use cases/trials non-technical evaluation. In that sense several interviews had been carried out with different stakeholders. The scope of this evaluation was to identify insights about how iCore is creating value for service developers. As a result of that the main points followed during the interviews were: rapid application development using iCore concepts/architecture/framework, usability of iCore concepts/architecture/framework, technical skills required for developing new applications using the iCore framework, prototype running environment non-functional features (number of users, response time, etc.), open source components (CVOs and SLs), access to data, security and privacy.

This assessment activity (T6.5 Non-technical evaluation) is complementary to T1.5 Socio Economic Evaluation. In the former case, the stakeholders involved in the interviews are persons with technical background, and the above mentioned point had been evaluated. In the latter case, the stakeholders are mainly persons with business/marketing profile which provided insights about the business potential of iCore in different domains.

### 4.1 Smart home: Living Assistant

The non-technical validation of iCore in case of the Smart Home use case is done by means of an interview with a doctor leading a praxis within a consortium building with other doctors and a pharmacy. The building consists of 10 different praxis each specialized in another area of expertise and their job is service is split in consulting hours and home visits. During their job, all information the doctors can get is important and needs to be stored with high priority to privacy. Every time a patient is visiting, they try to differ between well-known patterns and new symptoms. Every change in intake of medicine or happening in patients' lives results to new deviations, which makes it important for them to obtain as much information as possible. Next to the data the doctors check at patients visit, it is important for them to get information from colleagues or hospitals as well (in case the patients had been hospitalized). Privacy and security makes it difficult for them to collect such data as every patient documents are collected manually and through the communication with the patient a lot of information gets lost due to expert knowledge that is needed for total understanding.

Daily business and different medications administered in parallel makes it difficult to identify various medications running at the same time. It happens that more than one medication is running in parallel with different doctor's caring for the patient (e.g. an orthopaedic specialist can prescribe pain reliever while a general practitioner gives antihypertensive drugs to a patient). This could lead to interdependencies that stay unknown due to the missing communication between the doctors.

Mainly they are collecting these data manually during the consultation. Blood pressure and pulse are taken standard to have at least periodically values to compare with (the older patients of course visit the praxis more often or are visited by doctors at home). For them it would be very useful if vital functions can be obtained more periodically e.g. if the patients are able to measure the values their self at home based on their clinical picture. In addition, general conditions of a patient are helpful, such as forgotten intake of pills, wrong intake or

simply comparison of daily routines (e.g. recognition of high change in daily pattern like immediately change of usual travel paths) what is not traceable and mostly recognized upon mutual trust.

The prototype features appreciated by the interviewee are:

- the interoperability and scalability of the system (enhance functionalities with any kind of sensor, actuator or service available);
- get informed in case of any error internally or on high change in transmitted data, so that they don't have to keep track about all data as long as they get informed or met the patient personally.

Apart from these, such a solution should tackle very seriously the security and privacy aspects (E.g. collecting data from patients need to be stored somewhere or transmitted and there should be clear policies that specify who has the right to access de private data).

An iCore based solution has the potential to help the doctors to schedule consultations based on predictions whenever an asset is needed or when there is more free-time due to the automated collection of needed data. But they need help to installing the system and it will be a good idea to have a marketplace wherein one can see the available services

## 4.2 Smart Office: Easy meeting

The non-technical validation of iCore for the Smart Office: Easy Meeting use case was done through interviews with a software architect in ATOS and a Senior Research Fellow in UniS, both of which are heavily involved in software development and as such can provide valuable opinion on the benefits an iCore compliant implementation can bring, seen from a software developer point of view.

It is worth noting that both interviewees saw the merit not strictly bound to the specific use case, but as a more generic implementation platform.

The ability to fast and easily develop new applications, compatible with the iCore platform was acknowledged. This is assisted by the use of standard APIs. As such the interviewees found the ability to easily integrate new customized features very useful. Also the ability to reuse services for composition of more complex services (even in a different context) is a feature deemed particularly beneficial.

Technical skills will be always required, however, the iCore platform allows for easy integration of new applications since the use of templates allows for "standardizing" the way a developer has to proceed, knowing also the ways the apps will need to interact with the other components, expose and retrieve information from other components and make use of their offered capabilities.

For both interviewees, the use of data plays a pivotal role in their business. One point of major concern for them is the heterogeneity of data sources that need to be tackled. As such the notion of virtualization which allows hiding the heterogeneity of data sources was acknowledged as assisting greatly in handling the underlying heterogeneity and its implications.

While the iCore platform allows for controlled access and use of resources, based on contextual information (e.g. through the properties of entries in the VO registry) some more dynamic privacy handling mechanisms that can adapt across services would be beneficial.

Both interviewees have seen parts of the Smart Meeting demo. They would like to see though a scenario where all architectural components are developed and involved in a prototype before adopting the iCore solution.

The building and exploitation of knowledge and self-x capabilities, which allow for improved performance and reduction in needs for manual reconfigurations is something extremely useful and desirable, but would need to be further demonstrated.

In addition, one of the interviewees mentioned the platform offered by the EU COMPOSE project as a competing solution, though the self-x capabilities of iCore appear to be not as much developed in the COMPOSE platform.

### 4.3 Smart City: Transportation

The non-technical evaluation of the smart city transportation use case is done by an interview with two persons working for a large car manufacturer. Among the main services provided to customers can be mentioned: the provision of information and suggestions that are useful to improve the driving experience, fuel efficiency and others.

According to the interviewees opinion, iCore is very attractive and potentially of high value for the car manufacturing domain. The most attractive feature is related to creation and the easy deployment of components (well separated and with standard interfaces) that could make the creation, enhancement, and/or replacement of services easier than today. There are concerns about the hidden complexity of such an infrastructure and related costs. The interviewee thinks the decoupling of components enabling easy replacement/enhancement and the standardization of interfaces, which potentially decreases the efforts needed to introduce new services attractive features of the iCore solution. In addition to these, the cognitive capability allows to build enhanced services with a short time to market.

Less attractive for a manufacturing company seems to be In the context of the Smart City Transportation Use Case the non-technical validation has been accomplished by means of two different stakeholder interviews. The stakeholder were internal Car Maker R&D Division managers: the Telematics Manager and the Telematics Features and Services Manager. Both the stakeholders are heavily involved in Car Telematics innovative actions and so perfectly suited to evaluate the proposed architecture and the use case which has been implemented in their domain.

Both the stakeholder have been positively impressed by the iCore architecture and proposed methodology. In both cases, the most important iCore features which have been considered relevant and interesting were:

- the potential standardization of communications links from/to the car, thanks to the Object Virtualization (VO)
- the impact on in-vehicle implementations and well-defined, modularized interfaces between the changes needed to make different levels of the current architectures compliant with architecture, allowing a decoupling of components and easy replacement or enhancement
- the availability of a “programmable” platform to speed up the deployment of innovative services towards the end users
- the availability of features to leverage the great amount of data coming from the vehicles in order to grow real world knowledge and in order to improve the creation of services and/or reduce their setup cost

Regarding the potential barriers or , it seems that the perceived value of the iCore approach is not clear. The interviewee expects that, in the domain of Smart Transportation, heavily depends on the availability of “external content providers”, i.e. road infrastructure real world object providers.

In general, both the stakeholder validated positively the iCore solution can provide approach for the Smart City – Transportation domain, from the point of view of the Car Maker. With respect to already known solutions, the most relevant iCore perceived added value through a significant acceleration in service development and deployment phases and, thus, a decreased time-to-markets the potential to “standardize” the way to realize services for the Car Drivers.

#### 4.4 Smart City: Urban security

In order to evaluate the smart city urban security trial, from non-technical point of view one interview was carried out with a solution provider for city protection systems from THALES. The applications developed by the THALES group for the domain and for which iCore provide a clear value are: airport security solutions; oil and gas infrastructure protection. In both cases the data from sensors, the derived information, and knowledge about the domain are very important.

The most attractive iCore features for the domain are:

- System self-adaptation, data format and data sources unification/abstraction, situation prediction, easy update/introduction of services;
- Machine learning with growing RWK/SK and situation prediction is deemed very attractive, as is the update of services or new services;
- iCore does not offer a knowledge model built-in improvement feature through simulation.

In the opposite direction, data & sources unification is not very attractive), because it does not add value at the business level, but is mandatory at the technical level.

In addition to these the cognitive techniques are valuable only if:

1. The models have been intensively validated according to exceptional and hard situations and demonstrate their theoretical efficiency during validation process with end-users and initial learning also with end-users;
2. The models can be enhanced easily after missions based on feedbacks and experience;
3. The models and overall behaviour of the iCore platform decision process can be summarized to the operators at run-time during missions: human in the loop with possible automated decision process switch off.

#### 4.5 Smart Business: Supply Chain Management and Logistics

The non-technical validation of iCore in case of the Smart Business use case is done by means of an interview with a solution provider for the cold chain logistics market (Antaris Solution, NL). The use case is in depth discussed and some parts of the trail have been demonstrated.

Another interview was carried out with a service developer (SIEMENS Corporate Tehnology, Romania) for different domains including cold chain logistics. The main topics of discussion were smart business use case and potential/added value of the iCore platform.

The complete interviews are available in WP1 D1.5.

### **Antaris Solution Evaluation**

The solution provider sees clear benefits in iCore concerning rapid application development. This is due at two levels: (1) the VO level ensures that somehow the way of interacting with data is standardised and (2) the iCore platform automates combining of data sources on best fit for the application (based on templates) and are explained as follows. The VO level is recognised as de facto communication standard; it requires e.g. hardware vendor to provide interfacing of their products towards iCore and it ensures that the solution provider has less effort in integrating new devices etc. It enables them to focus more on providing additional services to end-users in the cold chain domain (moving from device integrator towards providing analyses services). Learning capabilities and automated enrichment of data are very interesting. Would make it easier for us to move from reactive/proactive mind set to predictive plus analysis.

There is a clear change of technical skill that are required to operate an iCore platform versus the platform in use today: much more focus on domain knowledge and “template engineering” opposed to integration work. This would mean that other/new competences are required in the company.

iCore is perceived as “usable” by the solution provider, although the usability of the lower levels (VO and CVO) is much clearer than the higher levels. This has to do with the fact that iCore in this domain is not used by individuals, but by companies e.g. functions as natural language processing are not (so) relevant in the domain. What is important and might also be harder for the SL than other levels is the proven data integrity of the platform (certifications on this are required in the cold chain domain).

The solution provider finds it hard at the time of the interview to estimate what the runtime requirements are of iCore. How is the platform sold? What server capabilities are required etc.? These questions need to be answered before a reasoned judgement can be made about usability.

The solution provider embraces the concept of the de facto standardised VO level and information about devices which is present in the various registries in iCore. This eases their task of device management. The security concepts in iCore are perceived as ‘nice to have’, but not required (or asked for by customers). Since, sensors observe objects and not people and the end-users are companies, privacy is not a major concern and iCore provides sufficient measures.

The solution provider wants to move from system integrator role towards providing proactive analyses in the cold chain domain. For this role, data must be augmented with context e.g. to find the root cause of disruptions in temperature. iCore can provide this context; however, the solution provider perceives that much of “the magic” is in the creation of CVO’s and service level templates.

The solution provider has seen (parts of) the in T6.4 implemented use case and demonstrator. The demonstrator shows powerful iCore concepts and is well-received. However, the demonstrator (as may be expected) lacks important features that are required for an application to be deployed for real: ERP integration, GAMP certification, audit trails,

etc. For a demonstrator, the results of T6.4 are mature is the opinion of the solution provider.

### Siemens CT evaluation

The iCore platform or a system that implements the iCore architecture has the following clear advantages: the overall products development time is reduced; simplifies the process of accessing the data sources and collecting the data; simplifies the job of the domain expert; breaks the silos type of application typology and enhances the horizontal application development process; improves the way the apps are developed using the CVO and service templates; allows rapid application extension. All these advantages are very attractive for a service developer type of company as it is SIEMENS CT, regardless the domain for which develop applications (cold chain logistics, smart grids etc.).

### Technical skills required:

The advantage exposed by the CVO execution engine (which is wrapper over a CEP engine) is represented by the syntax used for creating the CVO templates. More precisely the domain expert doesn't have to have advanced programming skills in order to create the CVO templates. What is missing is a CVO container for executing jobs, because right now it provides only data analysis/processing capabilities.

From a logistic company perspective the iCore prototype (the instantiation of the iCore framework for the logistic domain) has a great potential: it reduces costs, increases the prestige, increases the shelf time and many others. But, a real logistic needs a lot of other features (e.g. parcel management, route planning etc.) which right now are not implemented and have to be further developed.

## 4.6 Smart Hospital Trial

### 4.6.1. Stakeholder Description

The trial will be done at the Santa Chiara Hospital in Trento, in the Neonatologia(Newborn) department. The people involved in the interview for the department were:

- Dr. Massimo Soffiati (Department Director)
- Dr. Annalisa Cuccu
- Marina Cologna (nurses coordinator)

The indirect "customers" of the Neonatologia department are, first of all, the newborn babies who receive the care and attention of the medical staff. It is therefore important to relieve the medical staff from having to dedicate too much time to non-core activities. Many medical devices in this department, like other hospital departments, are present to guarantee the daily work. In order to achieve the above objective and guarantee good healthcare services, it is also necessary to ensure a well-managed environment, able to track, monitor and maintain all the available technologies that are present inside the department in a correct way.

Identify which medical devices are inside the department and which are temporary outside, which devices are in the correct room / place and which have been misplaced, which ones are used more often than others etc., could generate a lot of inferred information that can improve the quality of the service the department supply to its customers. Currently this type of data is not collected and there are no means to acquire this type of information. Any inferred information is based on the experience of the people who work in the department but there isn't an automatic system that helps manage this type of information.



Furthermore, a recent Italian legislation (law 81/08) requires having a device book that contains all the history of the devices of a given type. For the moment, this type of documentation is created and managed in hardcopy. For every object typology there is a folder containing sheets with the history of the various devices (the information collected inside this folder relates to maintenance mostly). For each device there is ordinary maintenance (periodic) and extra-ordinary maintenance (based on usage / faults) that needs to be carried out. The routine maintenance is managed by a separate clinical engineering unit / company (Ingegneria Clinica) and the records of this type of operations are managed directly by this separate organisation. Currently roughly 240 “technologies / devices” are present inside the department (medical and non-medical but electrical / electronic devices) and have to be monitored. The list of these devices includes monitor, personal computer, medical equipment and all the electronics devices available inside the department.

During the interview, we have identified the objects that we have to monitor. In total, 36 is the number of devices that the system will have to monitor. At the same time we have defined, with the contribution of the stakeholders, the areas where tracking will take place. We have identified two gates / doors to monitor for entrance / exit of objects. These gates are the only two ones that can be used to access the department (the two gates in the image are visible in red). The area inside the department where to monitor and track devices is composed by 8 rooms and with a surface of 190 m2.

The stakeholders' principal needs and wish is to be able to have a collection of added data available. Having this data enables the stakeholders to have more information for taking some decisions concerning an efficient usage and procurement of devices. Observing the object movements and aggregating this data we can also create models that help derive usage trends and patterns that help medical staff assess the overall situation in the department. For example in the department there are two mobile incubators. The two devices are different in terms of functionality and during the interview it has emerged that one of these incubators is preferred with respect to the other. This is caused by the intrinsic features of the two different devices. In a case like this one, a system able to monitor the usage of the devices could be useful to give a concrete measurement of which of the two is more used and to what extent this is so. This information could be useful next time that the department will need to procure more of these devices. At the same time is possible have a real measurement of the usability of a device and know better if indeed it is needed or not. None others similar solution is known by the stakeholders'. The willingness to try our solution is strong and tied to the effective usefulness. The stakeholders' relate the presence of a Wi-Fi network inside the department.

The idea of trying the ICore solution inside the department has been accepted. The idea into the department would be to use this type of solution. This is tied to the real improvement that the application of this solution could generate inside the specific hospital structure.

The stakeholders asked only some question with regards to the usage of the tracking tags close to the medical devices. The doubt expressed by the stakeholder was related to any possible electromagnetic field interference generated by the tags. This type of question has been explained to the stakeholders' and the law that regulate this situation is described in Section 3.7.1.7.

## 4.7 Smart tour in the city trial

Dr Konstantinos Tsagkaris from WINGS ICT Solutions was interviewed in order to evaluate from a non-technical point of view the smart tour in the city trial. WINGS ICT Solutions pursues research and prototyping achievements in diverse areas such as smart wireless access, device management, IoT-cloud-wearables, software networks/ NFV/ management & orchestration, big data & predictive analytics and development of applications and services. Due to the nature of a stakeholder such as WINGS ICT Solutions (mainly Platform Service Provider) the value of iCore is not tightly linked to the Use Case. The value is the iCore platform itself being used for the development and delivery of IoT applications/services in a variety of domains, exploiting virtualization, dynamic/programmable service composition, dynamic resource optimisation, self-x.

WINGS considers the iCore Value proposition interesting especially regarding the features of:

- Self-management features (configuration, healing, optimization, protection).
- Autonomic management and control of devices in the smart home.
- The potential for dynamic on demand creation of IoT Apps, taking into account user preferences and context.
- Semi-dynamic / Semi-automatic software deployment for VOs and CVOs.

The iCore technology has the potential to improve the process of developing applications for customers because of:

- Easier creation of new services through use of the iCore platform and re-use of objects and applications. Allowing the offering of new types of services with increased context-awareness and degree of personalization due to cognitive features.
- Simplification of management of IoT infrastructure reducing the need for human intervention due to self-management.
- Potential facilitation of integration of devices (sensors, actuators, etc.) /data-sources.

#### 4.8 Smart Theme Park trial

The trial will be deployed on a thrilling ride in a theme park in Jiangsu Province in China. The interviewees include two stakeholders. One is a young customer randomly selected in the park who just finish a thrilling ride. The other is in charge of a thrilling ride in the park. He manages a team responsible for customer reception, security check, ride operation, gift selling, etc.

##### Access of data

The stakeholder working in the park does not use any data in his job, while the customer himself brings a smart phone to take pictures. However, when taking the rides, he is not allowed to carry the phone. The trial satisfied customer needs and provide a more attractive product than still and silent images.

##### Acceptance

The customer is very interested in the idea of the trial and after watching the demo souvenir we prepared beforehand, the customer is very willing to pay for it. The stakeholder who works in the park is also attracted by the potential profits the trial will bring to the park. Meanwhile, in terms of the privacy problems related to visual and audio data of customers, they are well resolved by the control access mechanism of iCore architecture.

### **Business ecosystem**

On one hand, the trial cooperates with theme parks and device producers to provide a personalized product for tourists. On the other hand, when large amounts of customers are attracted, the trial could act as a platform of marketing and promotion for all kinds of amusement parks. This trial creates a new business channel to connect customers and amusement parks from a novel anchor point where the social and emotional needs are satisfied.

## 5 Conclusion

---

This document presented the results on the technical and non-technical validation of iCore concepts derived through the implemented iCore proof of concept prototypes and trials. The document started from an overview of the different proof of concept prototype and trial environments in terms of functional components and hardware elements. Then results on the technical validation of the various proof of concept prototypes and trials were presented. For each proof of concept prototype and trial, a set of test cases was presented. Each test case description included information on the involved entities of proof of concept prototype and trial environment, the relevant functional requirements, pre-conditions, execution steps, success criteria and finally results of the execution of the test case in the respective proof of concept prototype or trial environment. In total 29 test cases for 4 use cases and 4 trials have been presented.

In addition, results on the non-technical validation of the different proof of concept prototypes and trials were presented. For the non-technical validation interviews with stakeholders were organised with the aim of evaluating aspects such as rapid application development using iCore concepts/architecture/framework, usability of iCore concepts/architecture/framework, technical skills required for developing new applications using the iCore framework, prototype running environment non-functional features (number of users, response time, etc.), open source components (CVOs and SLs), access to data, security and privacy.

Analysing the results of the non-technical evaluation one may conclude that the iCore architecture and concepts are attractive for the service developer type of stakeholders. The most interesting iCore features mentioned by the stakeholders are: the native ability to handle numerous heterogeneous devices, the interface for the VOs, the creation of CVO's and service level templates and the cognitive techniques. By implementing the iCore technology they estimate that development time of related applications is shorter and; furthermore, one can offer additional/improved services to the end users. Even if the interviews have revealed that there are domains where not all the iCore features provide value, the general feedback is positive.

## 6 References

---

- [1] FP7/ICT project iCore (Internet Connected Objects for Reconfigurable Eco-systems, ICT-287708), Oct. 2011 – Sept. 2014, Website: <http://www.iot-icore.eu>, accessed Sep. 2012
- [2] FP7/ICT project, Deliverable D6.2 Final proof of concept prototype for Smart Home: Living Assistant, May 2014
- [3] V. Stavroulaki, N. Koutsouris, Y. Kritikou, P. Demestichas, “Cognitive Management of Devices in the Wireless World”, in Wireless Personal Communications, Springer, October 2013, pp: 1-34, doi: 10.1007/s11277-013-1468-2.
- [4] FP7/ICT project, Deliverable D6.1 Specification of proof of concept prototypes for four application domains, July 2013

## 7 List of Acronyms and Abbreviations

Acronym	Definition
API	Application Programming Interface
AROD	Approximation and Reuse Opportunity Detection
C2	Control and Command
CBRNE	Chemical Biological Radiological, Nuclear and Explosive
CCE	CVO Composition Engine
CEP	Complex Event Processing
CoAP	Constrained Application Protocol
CVO	Composite Virtual Object
DB	Data Base
FP7	Seventh Framework Programme
GUI	Graphical User Interface
HTTP	Hypertext transfer protocol
HVAC	Heat, Ventilation and Air Conditioning
HW	Hardware
ICT	Information and Communications Technologies
IED	Improvised Explosive Device
IETF	Internet Engineering Task Force
IoT	Internet of Things
LED	Lighting Emitting Diode
ML	Machine Learning
MQTT	Message Queue Telemetry Transport
MVOC	Modular Virtual Object Container
NLP	Natural Language Processing
OGC	Open Geospatial Consortium
OWL	Web Ontology Language
PoC	Proof of Concept
QoS	Quality of Service
QR	Quick Response
RDF	Resource Description Framework
REST	Representational State Transfer
RWK	Real World Knowledge
RWO	Real World Object
SER	Service Execution Request
SK	System Knowledge
SL	Service Level
SMS	Short Message Service
SoC	System on Chip
SR	Service Request
ssCVO	Smart Santander CVO
UI	User Interface
VO	Virtual Object
wCVO	Weather CVO

WS	Web Service
WSN	Wireless Sensor Network
XML	Extensible Markup Language



## 8 Appendix: Questionnaire used for non-technical validation

### iCore Stakeholder interview guide

*(text in italics shows instructions and examples that serve as background information for the interviewer, text in italics + underline gives small tasks that need to be prepared by the interviewer before the interview)*

#### Part 0: Introduction

*Before I start with my interview I want to introduce myself and explain the goal of this project and the reasons for this interview to you.*

- Introduce yourself
- Explain the goal of iCore and this interview:
  - o [Interviewer company] takes part in a European research project called iCore (Internet Connected Objects for Reconfigurable Ecosystems) which aims to provide the technological foundations for the Internet of Things (IoT). There are two main classes of problems that need to be addressed by the Internet of Things (IoT). First, **the vast amounts of heterogeneous objects**. Second, the existence of **different users and stakeholders**.
  - o The iCore initiative addresses two key issues in the context of the Internet of Things (IoT), namely how to **abstract the technological heterogeneity** that derives from the vast amounts of heterogeneous objects, while enhancing reliability and how to consider the views of different users/stakeholders (owners of objects & communication means) for ensuring proper application provision, business integrity and, therefore, maximize exploitation opportunities.
  - o The iCore proposed solution is a **cognitive framework** comprising three levels of functionality, reusable for various and diverse applications. The levels under consideration are virtual objects (VOs), composite virtual objects (CVOs) and functional blocks for representing the user/stakeholder perspectives.
  - o The goal of this interview is twofold: first, we want to analyze your needs and wishes, as an iCore stakeholder, with respect to IoT solutions. Second, we want to validate the iCore value proposition for your business with you.
  - o We have identified you as an iCore stakeholder, since ... [specify]
- Interview process
  - o We will use your responses together with those of other stakeholders to identify to which extent the iCore Value proposition differs or overlaps in different applications and to derive insights about the acceptability of iCore in different application contexts. This interview will be recorded/noted and be

processed to a written report. The report will not be included integrally in a project deliverable but in a consortium only appendix. Upon your request we can anonymize your name and company and provide you the opportunity to approve the report.

- I expect that the interview will last for 2-3 hours. *(Decide if you want to record the interview, ask for permission if you do)*
- Are we allowed to mention your name and organization when reporting on the results or do you want to stay anonymous?
- Definitions:
  - *The Internet of Things (IoT) refers to uniquely identifiable objects and their virtual representations in an Internet-like structure. Typical examples are: smart grids, remote smart meters, assisted living, traffic control, ...*
  - *iCore: Internet Connected Objects for Reconfigurable Ecosystems*
  - *VO: virtual objects*
  - *CVO: composed virtual objects*
  - *RWO: real world objectsBM: Business Model*
  - *VP: Value Proposition*

## Part A: Stakeholder jobs, needs and wishes

*In this first set of questions, I will ask you a few questions about the organization you work for to get a better understanding of your work, your customers and the role of data in your organization.*

1. Could you give a short introduction of yourself and the organization you work for?
  - Is your role with respect to iCore \_\_\_\_? *E.g. service provider, object provider, soft component provider, end user... The interviewer should prepare the answer to this question in beforehand and cross check with the interviewee if this role is indeed correct.*
2. What are the main jobs you/your organization is trying to accomplish?
  - Which (business) processes or activities do you try to control / manage? *(Think of control in terms of measuring, monitoring, predicting and correcting and improving)*
  - Does data play a role in accomplishing these jobs? *(Explain that data will be zoomed in in a few minutes with more questions)*
  - Which real world objects play a role in these processes? Do they produce or process data?
    - *Examples real world objects: Sensors, mobile devices, cameras,... but also transported goods, people, rooms, schedules,...*
3. Who are your customers and their needs?
  - Which services do you offer to these customers?

## Exploring use and opportunities of data, information, knowledge

*The following questions will zoom into the topic of "data". In this interview we do not distinguish data, information or knowledge. \*It might be useful to specifically consider the*

*role of reusable information, applicability of predefined templates, aspects that might be observed or learned (on the fly).*

4. Which information/data do you currently use (collect, analyze) to fulfill your job(s)?
5. How do you gather these data at the moment?
  - Do you gather these data yourself or do you get it from third parties? If so, from whom?
6. Which additional information/data would you like to have to improve your job(s)?  
*(Interview leader should prepare some domain specific examples of potentially interesting other data sources in beforehand)*
7. In which way could additional information/data improve your job?\*
8. Do you see other opportunities for data in your domain?
9. Which barriers do you encounter in gathering, analyzing and using these additional data?
  - Are the barriers of organizational, technical or juridical nature?

## Part B: iCore value proposition

*In the next half hour I want to introduce you to the iCore platform and the value proposition it offers for your domain and your organization in particular. (please make an estimation of the time in beforehand depending on the complexity of the case)*

10. Introduction iCore solution + value proposition *(see separate document with Interview Resources, XXX)*
  - Interviewer explains the general idea behind iCore incl. features *(examples of features are: self-x, virtualization, growing real world knowledge, growing system knowledge, high-level semantic expressions, complex service abstraction, dynamic/programmable service composition, dynamic resource optimization and*
  - the instantiation of iCore for this Use Case domain, i.e. what iCore looks like in this Use Case *(it may be difficult to distinguish iCore from the prerequisites/context of the solution)*
  - How could iCore help to create value in your Use Case? *(this question is meant to check if the interviewee indeed understood the iCore concept; this further detailed in the next question)*
  - Explain the value delivered through iCore making use of the Value Proposition you have developed *(see chapter 3)*
    - *think of better control with respect to goals, improved compliance, decreased risks, improved opportunity spotting and realization, improved continuity, improved utilization*
    - *The Value Proposition / business model canvas has to be prepared in beforehand by the interviewer, (s)he can receive support from TNO for filling out the canvas*
  - Sketch how typical non-technical requirements are being addressed in iCore *(see separate document with Interview Resources, XXX)*
    - Control
    - Quality Guarantees
    - Security
    - Interoperability
    - System Performance
    - Scalability

## Part C: Compliance with stakeholder's needs and wishes (socio-economic requirements)

*After having introduced iCore to you, I would like to discuss with you if the iCore solution indeed can provide value for your organization and how it corresponds with your needs and wishes.*

11. What is your first reaction to the iCore value proposition?
12. Which features do you find most attractive?
  - *Prepare the features applicable for your domain in beforehand (see question 10)*
13. Which features do you not like?
14. Which features do you miss?
15. What would be the added value of the iCore solution for your job?
  - How important is an increased quality/performance of ...product/service... for you/ your organization? *(Also consider perception of quality)*
  - How important is and increased safety of ...product/service... for you/ your organization? *(Also consider perception of safety)*
  - How important are better strategic choices, **e.g. with respect to new products or services or new roles in an ecosystem**, for you/ your organization?
16. Does iCore framework, allows you to develop applications in other domains than the ones where you are leader? If yes, what domains? If no, what is the main obstacle?
17. Considering that the framework has a standard interface for resources and all the soft component providers use it, will you develop drivers for you sensors that are compliant with iCore technology? *(this question is valid only for sensor vendors type of stakeholders)*
18. What type of services (service templates) can you offer for an iCore enabled platform?

## Part D: Competing solutions

19. Which other solutions do you know in your domain that have similar functionality as iCore? *(Interview leaders should prepare this question with a first quick scan of other solutions)*
20. Have you considered using these solutions?
21. What is the added value of iCore over existing solutions?

## Part E: Acceptance

22. Would you make use of the iCore solution? Under what conditions?

## Part F: Conclusion

Thank you very much for your time. Are we allowed to mention your name and organization when reporting on the results or do you want to stay anonymous? *(better to ask this question twice at the beginning and end of the interview)* Do you want to receive the results of this study?