COSMOS

Cultivate resilient smart Objects for Sustainable city application

Grant Agreement Nº 609043

D7.4.2 Smart heat and electricity management: Evaluation and recommendations (Updated)

WP7 Use cases Adaptation, Integration and Experimentation

Version: 3.0
Due Date: 30 October 2015
Delivery Date: 30 October 2015
Nature: R
Dissemination Level: PU

Lead partner: Hildebrand Technology Limited
Authors: Joshua Cooper
Abie Cohen
Internal reviewers: Sergio Fernández Balaguer
Andres Recio Martin
Saima Iqbal

www.iot-cosmos.eu

The research leading to these results has received funding from the European Community's Seventh Framework Programme under grant agreement n° 609043

Version Control:

Version	Date	Author	Author's Organization	Changes
0.1	22 April 2014	Abie Cohen	HILD	Creation
0.2	11 March 2015	Joshua Cooper	ICCS/NTUA	Initial Review
0.3	22 March 2015	Abie Cohen	HILD	Review Changes & Updated
0.4	23 March 2015	Joshua Cooper	HILD	Penultimate review and updates
1.0	26 October 2015	Abie Cohen	HILD	Final review
1.1	27 October 2015	Achilleas Marinakis	NTUA	Internal Review
2.0	27 October 2015	Abie Cohen	HILD	Penultimate Changes
2,1	29 October 2015	Juan Rico	ATOS	Internal Review
3.0	30 October 2015	Abie Cohen	HILD	Final Changes

Annexes:

Nº	File Name	Title
1	Appendix A	Appendix A

Table of Contents

1 Introduction 5
2 Methodology 6
3 Criterion 7
4 Technology 9
5 Use Case Scenarios 10
5.1 London Borough of Camden Heat Network 10
5.1.1. Use Cases 10
5.1.2. Technologies 13
6 Requirements 18
6.1 Unmet Requirements 18
7 Recommendations 20
7.1 Overall Recommendations 20
8 Conclusion 22
9 Appendix 24
9.1 Requirements evaluation percentages 24

Acronyms

Acronym	Meaning
3DES (TDES)	Triple Data Encryption Algorithm
API	Application Programming Interface
CBR	Case Base Reasoning
CEP	Complex Event Processing
CHP	Combined Heat and Power
COSMOS	Cultivate resilient smart Objects for Sustainable city applicatiOnS
D	Deliverable
FC	Functional Component
GPIO	General Purpose Input Output
GVE	Group VE
HMM	Hidden Markov Model
IoT	Internet of Things
ML	Machine Learning
SQL	Structured Query Language
SVM	Support Vector Machine
VE	Virtual Entity
VE2VE	Ve to VE
WP	WorkPackage
XP	Experience
Y	Year

1 Introduction

This deliverable focuses on the re-evaluation of the adoption of the COSMOS technologies in real-world smart city cases. We will use the use case scenarios formulated in Year 2 as the basis for the assessment. It is important to re-evaluate the technologies thoroughly by looking at their assessments from Year 1 and looking at their current consistency, correctness and completeness.

Once again these measures can be expanded into a list of criteria with which we will evaluate each of the technologies in WP3-6 for Year 2. In using these measures, we will implement a benefit vs. cost assessment when applied to specific use cases in each of the two COSMOS scenarios.

Finally, it is important to use what we identify in this deliverable to make recommend what our next steps should be for Year 3 and where we should focus our efforts in terms of research activities and productivity.

In this Work Package deliverable we will

- provide the background material describing the current situation in each of the new use case scenarios as well as the ones from Year 1
- define a clear set of evaluation criteria for any given technology
- re-identify the technologies used in COSMOS, in particular the ones implemented in Year 2 prototypes
- assess each of the technologies in WP3-WP6 against the aforementioned criteria
- evaluate each of the technologies in WP3-WP6 alongside the overall requirements
- make extended recommendations based on the evaluations

The outcome of this deliverable is to evaluate the technologies developed in COSMOS in each of the use case scenarios, against a clear and complete set of criteria.

2 Methodology

We follow the same methodology for evaluating the COSMOS technologies as we used in D7.4.1 [1].

In order to thoroughly evaluate the COSMOS technologies and assess the benefits they provide in different domains, we must firstly clearly define a complete set of evaluation criteria. This set of criteria must look at all aspects of a given technology and be able to apply to specific use cases.

We then collect and list all of the technologies used in COSMOS, which have been described in WP3-WP6. Consequently, we look at the London use case scenarios and evaluate how each of the technologies described will be used in them. This is done in a very structured manner by using the criteria to test different aspects of the technologies.

Next, we consider the requirements in D2.2.2 [2] and assess whether the use of the technologies developed in COSMOS will meet the necessary standards and solve the corresponding issues. These are the *Updated* requirements from Year 2.

Finally, based on the results from the evaluation of the technologies in each of the new Smart Heat and Electricity Management use case scenarios, we recommend next steps to take in COSMOS. Using the benefits and costs we found the technologies provide in different domains, we can help direct the research activities of the project.

3 Criterion

In this deliverable we re-iterate the criteria with which we evaluate the COSMOS technologies. These same criteria for evaluation were used in the D7.4.1 deliverable.

In order to properly assess the efficacy of the technologies used in COSMOS and their applications to the different use case scenarios, we need a set of criteria to test each of them against. This will ensure that each evaluation is fair and comparable. It is important that the criteria we choose to use capture the key measures of evaluating technology; namely consistency, correctness and completeness.

When looking at these measures, we realise that they can be broken down further to fundamentals and grouped into four main criteria blocks. We take consistency to mean how technically feasible, reliable and extendible the technology is. Correctness refers to whether it satisfies the problem at hand and how convincingly it does that. Finally, we take completeness to mean whether it is actually acceptable to implement such a technology and weigh its pros against the cons.

The following are a list of evaluation criteria which we will use as a check list when assessing the efficacy of the COSMOS technologies:

1. **Functionality**
   a. *Satisfaction*.

      This concerns the extent to which the designed product satisfies the requirements.

      Does the technology solve the problem? Is it a direct or indirect solution? Does it completely solve the problem or only partly?

   b. *Ease of use*.

      This concerns the ease of use for the users. The users are e.g. operators and application engineers.

      Is it easy to design, implement and maintain? What programming languages are required, if any, and how well known are they? Are some libraries, if any, required and how accessible are they? Does it require specialised operators or application designers?

   c. *Reusability*.

      The extent to which the product can be used in other situations. Includes scalability and ability to use in (dis)similar contexts.

      How extendible is this technology? What sort of scale can it be rolled out to? Can it be applied to any other components of COSMOS? How generalizable is it or is it extremely specific/custom?

2. **Construction**
   a. *Structuring*.

      This concerns the partitioning of the product in logical or physical components.

What architecture is used? How complex is the system? How do different components in the technology communicate with each other and how efficient is that?

b. *Convincingness*.

This concerns the evidence that the construction will work and has the defined functionality (empirical proof/statistical argument).

How well known is this technology? What sort of research has to be done before design and implementation? Has this been used in another component of COSMOS? What is the likelihood of the problem being solved by using this technology as a solution?

3. Realizability

a. *Technical realizability*

This concerns certainty that it is technically possible to produce the product.

What technical requirements are there? How difficult would it be to implement this technology? Do the technical components that make the system's architecture link well together?

b. *Economical realizability*

This concerns the business case for the product.

Is the application of this technology financially feasible? Can the cost be covered by scalability and if so what sorts of volumes are we looking at? Do the benefits outweigh the costs? Is the technology worth the justification? Or is there a more cost effective solution that satisfies the problem?

4. **Impact**

a. *Risks*

Risks of the product during development stage or use.

Does the technology introduce new problems? Are there any privacy or security issues inherent to this technology? Are there authorization restrictions between components? Are there any risks that could end up affecting the end users through the applications?

4 Technology

A technology is the realisation of a function in the Internet of Thing's Architectural Reference Model [3]. This includes physical devices, platforms, services and analytics; all of which are used to solve certain problems or add particular functionality to an IoT system.

COSMOS aims to build a smart system that uses Things in the space of IoT to solve problems experienced in cities nowadays. The two use case scenarios we focus on are Heating Networks in London and the Bus System in Madrid. In order to solve the issues that arise in the two scenarios we develop certain technologies in WP3-WP6. These technologies, when combined, produce the overall COSMOS system ranging from the hardware to the software and from the servers to the sensors. Each of the technologies mentioned in this section fulfil a specific role and have a purpose in COSMOS.

This section provides us with a clear list of technologies used in COSMOS (Year 2), described fully in WP3-WP6 and whose applicability is described in the next section:

(WP3) D3.2.1: *End-to-end Security and Privacy [4]*

- Privelets
- Node-RED Security

(WP4) D4.2.1: *Information and Data Lifecycle Management [5]*

- Integration with Analytics Framework

(WP5) D5.1.1: *Decentralized & Autonomous Things Management [6]*

- The Planner
- Social Analysis
- Network Runtime Adaptability Module

(WP6) D6.1.1: *Reliable & Smart Network of Things [7]*

- Inference/Prediction Functional Component
- Pre-processing Functional Component
- Event (Pattern) Detection Functional Component

5 Use Case Scenarios

5.1 London Borough of Camden Heat Network

5.1.1. Use Cases

5.1.1.1. Heating Control

Use case : Heating Control
ID: 5
Brief Description: The Energyhive system [8] is measuring the temperature of the properties where it is installed and has the ability to control the delivery of heat through a valve. A new tablet has been deployed within the property that allows for a set point and schedule to be entered. Feedback from users has been that they would like the system to automatically help them set a programme and manage efficiencies on an ongoing basis, for instance detection of whether or not at home; using weather forecast to help with program and supply side management when solar thermal is available for use. The tablet is a COSMOS compatible device and it can act locally to run case based reasoning in an efficient manner.
Primary Actors: Resident
Secondary actors: Mechanical & Electrical Engineer, Sustainability Officer
Preconditions: EnergyHive system must be installed within a resident's premises
Main Flow: 1) Resident will select an autopilot function on their tablet 2) Autopilot will determine a recommended set point for the temperature in the house 3) Set point can be over ridden by the resident 4) The system will learn the patterns of occupation and adjust the run programme to turn off the system based on un-occupied property; the resident can over ride 5) Savings should be quantified over using a normal time based programmer
Postconditions: An improvement in the efficiency in the heating system should be reported

5.1.1.2. Building Performance Management

Use case : Building Performance Management
ID: 6
Brief Description: The boiler systems within buildings have master programmers and temperature settings that are controlled by a Trend boiler control system. There are also verification instruments installed within buildings to measure the effects of the boiler control, they can provide feedback to inform the run time commands to the boiler control as well. A more granular view of the energy demand, including tradeoffs with electricity usage is desired so that individual residential premises are getting higher comfort while balancing the energy input.
Primary Actors: Mechanical & Electrical Engineer, Sustainability Officer
Secondary actors: Capital Planning Officer, Resident

Preconditions: EnergyHive system must be installed throughout each building in the estate, boiler controls and verification systems must be installed
Main Flow: 1) Temperature readings are collected at a room level within Camden hostel premises 2) Electricity metering will be logged as an input into the energy demand 3) The energy balance model will be run against the Trend readings and the temperature/electricity readings showing performance indicators (degree hour per kWh) against a network model for the delivery 4) Normalisation for seasons and weather conditions should be applied (subtract degree hours inside versus degree hours from weather)
Postconditions: Ranked performance of buildings and properties is reported

5.1.1.3. Capital Planning

Use case : Capital Planning
ID: 1
Brief Description: The EnergyHive system in each building enables Capital Planning officers to perform a more rigorous cost/benefit analysis of suggested programs or technology installations. The system provides accurate information as to the carbon/monetary saving of an implementation.
Primary Actors: Capital Planning Officer
Secondary actors: Mechanical & Electrical Engineer, Sustainability Officer
Preconditions: EnergyHive system must be installed throughout each building in the estate
Main Flow: 6) Sustainability Officer identify an opportunity for environmental improvement of system 7) Engineer select appropriate technology for instalment 8) EnergyHive system provides detailed information as to the effect of the change in the system 9) Capital Planning officer uses EnergyHive information to assist in cost/benefit analysis
Postconditions: The Capital Planning officer decides whether to rollout the proposal

5.1.1.4. Identification of Opportunities

Using machine learning, identify where energy savings opportunities exist. This will help sustainability officers suggest projects that can then be put through the Capital Planning use case.

Use case : Identification of Opportunities
ID: 7
Brief Description: The EnergyHive system running in planning mode can use machine learning to suggest opportunities for efficiency. This is largely an unsupervised learning exercise where cause and effect models can be run with comparisons to other like buildings or similar conditions that have been observed

Primary Actors: Sustainability Officer
Secondary actors: NA
Preconditions: EnergyHive system must be installed throughout each building in the estate
Main Flow: 1) Sustainability Officer creates model constraints for parameters to optimise (i.e. cost or carbon savings desired with physical systems 2) Model runs within system bringing up bands of savings that can be made from changes in input parameters 3) System provides control ranges that would have to be implemented in order to make potential savings
Postconditions: A quantified opportunity for efficiency within the energy system is presented for evaluation

5.1.1.5. Minimising Carbon

Use case : Minimising Carbon
ID: 2
Brief Description: An effective way to minimise carbon is to give more weighting to processes with lower carbon production levels, whilst maintaining the demand. The interconnected IoT-based system using an energy platform will make possible effective management of the energy supply in order to minimise carbon production. With minimal input by the resident or site staff, the system will predict the estate's heat and electricity consumption in half hourly intervals and manage the CHP and boiler accordingly.
Primary Actors: Resident
Preconditions: Specialised Instalments 1) Gas Flow meter to CHP from boiler to regulate the Gas supply 2) Control system with temperature sensor on boiler 3) Flow meter/temperature sensor on Solar Thermal 4) Heat meter in each dwelling 5) Communication infrastructure between sensors and hub
Main Flow: 1) System predicts the estate's heat and electricity demand for a half hour period 2) System calculates required gas supply and distributes to CHP and boiler accordingly 3) Carbon produced is measured 4) Individual resident heat consumption is monitored
Postconditions: 1) The resident is charged for their personal heat consumption 2) Prediction errors are logged to improve system on later iterations

5.1.1.6. Minimising Demand

Use case : Minimising Demand

ID: 3
Brief Description: Another method of reducing carbon production is to minimise the demand for Heat energy production. This is possible through the current IoT platform, namely EnergyHive (designed by Hildebrand). The EnergyHive system will use smart meters to report real-time energy consumption information automatically and remotely. The system assists the user in setting a heating schedule with accordance to their budget.
Primary Actors: Resident
Preconditions: 1) EnergyHive system implemented in each dwelling 2) Valve up/ down control system to the radiator
Main Flow: 1) Resident accesses their customer account to view balance 2) Resident can set a heating schedule 3) Resident is given tariff and projected balance for a given schedule
Postconditions: 1) User can optimise their schedule to minimise their consumption

5.1.2. Technologies

5.1.2.1. End-to-end Security and Privacy

Privelets component is responsible for the authentication process on top of any kind of VE2VE communication, in order to avoid possible VE impersonation. VE2VE communication includes:

- Accessing a VE's IoT service from another VE
- Decentralized discovery
- Recommendation service between VEs
- Information (e.g. Experience) Sharing

By preventing VEs impersonation, Privelets component tries to give added value to the Trust & Reputation model introduced in the context of WP5, whereas by ignoring repetitive requests, it aims to enable the VEs to protect themselves from wasting valuable computational resources caused by malicious attacks.

Privelets component's source code is developed in Java programming language and depends on Jetty Server, Apache-Jena, Pellet-Jena, json-simple and other Java libraries. The prototype also relies on FreeLan, which is an open source software, in order to establish the COSMOS VPN and connect the VEs to it.

Privelets satisfy the requirement of adding an authentication layer to the hardware in a simple yet robust way. The programming language and associated technologies are easy to use and widely well known. It is easy to design, implement and most importantly maintain. Furthermore, the benefits of using the object store technology is that it ties in well with other elements of COSMOS, as they are all built with the other components in mind.

In terms of construction, they are very straightforward and intuitive to set up in the system. Their efficacy and reliability; however, remains to be seen as this will be proven during larger scale prototypes later on in the project. This is important to note as simple testing may not show all issues and/or limitations.

Technically speaking, it's relatively simple to use this technology in COSMOS however one could argue that it is potentially overkill to satisfy the requirements. This technology is fairly new and untested and perhaps a more established technology could have been used to assure efficacy. However, as we are trying to build an innovative smart city solution with advanced capabilities and functionalities, the choice of privelets is justified.

There are no real risks of using this technology as they deal with the authentication layer of the components. Of course, any reliability issues that occur will create huge risks and will require an entire rethink of how to tackle this issue.

The visual tool for wiring the Internet of Things is a platform-independent software framework that has been developed having in mind its usage not only in big servers or the cloud, but also in small, embedded computers, as the Raspberry PI, Arduino and similar ones. Traditional IoT development can be very technical: access to the GPIO and other hardware modules require skills in C or assembler, output of data to web services or sending tweets and emails requires the use of complex APIs. Node-RED [9] takes care of the technicalities to let concentrate on the logic of the workflow. While most programming in Node-RED is done visually using pre-defined functions (a.k.a. 'nodes'), any additional functionality can be added in JavaScript.

This new technology is a brilliant example of how to build a smart city using state of the art technologies that satisfy current system requirements and needs. Its simple to use interface and wide variety of plugins and components, makes Node-RED a scalable and sharable technology.

The interface makes structuring clear and simple and the concept of data flows inspires confidence that the components using this technology will run smoothly and as expected. In terms of technical realizability, one could argue that this technology is perhaps too much work to implement for the job that it satisfies. Similarly, it could be possible to find a more lightweight technology to satisfy the same requirements without the expensive frills of Node-RED.

5.1.2.2. Information and Data Lifecycle Management

In COSMOS we store historical data for VEs in object storage (based on OpenStack Swift), in a format that is amenable to access by Spark SQL for the purpose of analytics on the data. Our driver allows optimized access to this data. All COSMOS services which make use of historical data can benefit – including applications which use machine learning on historical data and applications which analyse or report on past behaviour or activity of VEs.

The driver is used to help access data from the COSMOS storage system and is tailored to suit the particular structure and implementation style of the object store. In terms of functionality, this driver is specifically designed to work with OpenStack Swift and Spark SQL so is limited in one sense, but could potentially be used when developing the aforementioned technologies in Y3.

Seeing as it is simply a driver for data access its structure and convincingness remains strong and intact. As it simply 'works' we have no reason to questions its setup or ability to provide easy and cheap access to all of the data being stored in the COSMOS servers.

In terms of realizability, both technically and economically speaking, this technology is definitely feasible as it provides us with exactly what we need and is lightweight and cheap.

The only comment to be made here is that the development costs of this technology would be higher than alternatives so that it works with the current setup COSMOS is using.

Once again, we assess the risks of this technology and conclude that there are no major worries or concerns in using the driver to access the data from the object store.

5.1.2.3. Decentralized & Autonomous Things Management

As it is stated in D5.1.1, the functional component that enables the VEs to use CBR is the Planner. The Planner will become part of the VEs during their registration time and will run locally. The main functionality of the Planner is providing the VEs with the ability to react to problems using a reasoning technique for finding the most appropriate solution to be applied based on similarities to previously encountered or experienced situations. This is a step towards the autonomicity of the VEs as depicted by the goals of Task 5.2 (Autonomous and Predictive Reasoning of things).

The interaction metrics (Shares, Assists and Applauses) monitored by the Social Monitoring component of a VE are stored locally in the corresponding Followees Lists. These metrics are calculated in a distributed manner by the VEs on a per-VE basis and are the main input for the services provided by the Social Analysis and Friends Management components. From these metrics, the social indexes of the several kinds of Friends are extracted. These are the Trust Index, the Reputation Index, the Reliability Index and the Dependability Index. It is evident that since the social indexes will constantly change, it is quite important to take under consideration their evolution. Although a VE may have a low Reputation Index when we study a wide time-window, it may have a much greater Reputation Index when we study a smaller and recent time-window. This means that the specific VE is improving and this improvement should be evaluated fairly by the system and the community. For this reason, for each Followee in each Followees List the timestamps (unix time) and the evaluation of the last e.g. 3-10 interactions may be kept, so that, when applying simple rules, the evolution of the indexes can be studied.

The Network Runtime Adaptability module is able to dynamically assign resources performing different activities within COSMOS architecture. In this sense, the key objective of this module is to control the resources usage of every single component. The monitoring of resources usage enables the optimization and prioritization of processes inside a VE. Besides of this, the same functionality can be used in a multi-CPU environment for distributing the computational load thus minimizing the risk of blocking processes.

These technologies allow the VEs to use the Case Based Reasoning technology autonomously so that decisions are actually made locally with the knowledge of a decentralised system. Although it does achieve an extremely powerful logic base, it is a complicated system to manoeuvre and needs to be designed with care and caution. Having said this, the concept behind this technology is extremely scalable and can be extended to the Trust and Reputation models as well as helping integrate autonomicity in other COSMOS components.

Unfortunately, as the concepts introduced by these technologies are quite convoluted and highly theoretical, its convincingness remains to be seen. The structuring of the Planner and the Social Monitoring components in the prototypes will be the pivotal point of assessment for this technology.

The gains from localising these technologies and attaining a decentralised autonomous system are clear; however, reality dictates that this technology could very well be technically insufficient, in terms of reliability and consistency, for its added cost and effort.

As these technologies are most beneficial when integrated into the system locally, the added security risks are one of our concerns. Extra time and effort has to be done in WP3 to ensure that the hardware is encrypted and reliable. Furthermore, making decisions locally adds in the concern that it's more difficult to see how the system is operating, as it creates an extra point of failure. In this sense the Trust and reputation model becomes critical when moving from model to real life because damage caused by malicious users could produce very relevant losses. Effectively, it is a tradeoff between scalability and autonomy failures.

5.1.2.4. Reliable & Smart Network of Things

The Inference/Prediction component is responsible for analysing raw data in order to provide high-level knowledge which can be used for automated, proactive and intelligent applications. In this regard, we have implemented several pattern recognition algorithms on the use-case scenarios such as different variants of Support Vector Machines (SVM) and Hidden Markov Models (HMM) for inferring high-level knowledge.

We have also explored several regression mechanisms for time series prediction of data for providing proactive solutions for smart city applications. In this chapter, we briefly explain the architecture, interfaces of the component and finally the application of the component with the help of use-case scenario.

Pre-Processing is a generic component and different components such as inference/prediction FC and Event (Pattern) detection can use it according to their requirements. It involves several functions ranging from simple data cleaning mechanisms to more sophisticated mechanisms such as data aggregation or feature scaling.

The Event Detection component is intended to provide the functionality for near real-time processing of data for event detection by providing a hybrid solution based on Complex Event Processing (CEP) and Machine Learning (ML) methods. CEP provides the distributed and scalable solution for analysing data stream in near real-time manner, but it does not exploit historical data and the manual setting of rules is a major drawback. Rules for CEP are static and hence solutions provided by CEP are static as well. Though ML methods exploit historical data and provide more automatic solutions, they are unable to provide near real time solutions and scalability is a major issue associated with them. In our proposed solution, we exploit both approaches and combined them in order to provide a near real time solution which is more context aware, adaptive and exploits historical data as well.

All of the components developed in WP6 in Year 2 aim to bridge the gap between data and decisions by the means of pre-processing, data aggregation, feature manipulation and primarily machine learning techniques. The ease of use of these components decreases as we work on more state of the art statistical models and look to solve more complex problems. The big benefit of developing an array of tools and components to handle the data from extraction to the point of decision is that the technologies are highly reusable and extendable. A lot of the fundamentals from the Event Detection component and the Inference/Prediction component can easily be applied and integrated into other COSMOS components such as the CBR Planner developed in WP5.

Once again, the structuring of these components ranges, getting more challenging as we approach the more theoretical and experimental machine learning models. Similarly, in terms of convincingness, as there is no well-established best method of modelling the data sets that we have available to us. Once the prototypes for the Use Case Scenarios are fully up and running and heavily relying on these components, will we see whether they are able to reliably and accurately provide useful predictions.

The benefits for the CEP are clearly good motivations for the development of these components; however, once again we look at the economical realizability of some of the more theoretical and experimental modelling components. The time and effort spent on developing these techniques could prove to be wasteful if the results aren't impressive or that of a state-of-the-art technology. Fortunately, the version of CEP developed in COSMOS is able to run on simple devices and hence goes beyond the capabilities of current technologies, both in terms of ease of use and target platforms. The assessment of improvement provided by the more theoretical contributions will help in the identification of further potential of them as well as the number of domains that will be targeted.

Finally, we assess the risks of these predictive tools and evaluate whether they expose the COSMOS system to too much exposure especially when dealing with vulnerable citizens in a smart city. These risks seem to be minimal due to the testing and positive results shown in WP6.

6 Requirements

In this section, we evaluate the list of Requirements that have been put together over the course of COSMOS. We aim to evaluate our progress for each of the requirements on the three main criteria: consistency, correctness and completeness. We aim to achieve all three criteria for each of the requirements as this shows that we have fully satisfied the needs of COSMOS.

Approximately two thirds of the requirements have been met fully due to the design and implementation of the technologies in the use case scenarios. As the use cases are so diverse and test the system so thoroughly, we find that satisfying the needs of the requirements are consistent not only across both the London and Madrid systems, but also within these systems. The consistency of these technologies for all aspects of all use cases in each of the scenarios has been noted in Section 5 and this is verified in our evaluation of the requirements.

We now look at the requirements that fall into the category of consistent and correct but not yet complete. About 20% of the requirements have been marked with the 'Mostly Met' label as the aforementioned technologies do solve the issues that the requirements propose and do it in a smart, efficient and scalable way. Furthermore, these technological solutions can and have been adapted to fit different aspects of COSMOS and work well with all components in the system. The final criterion of completeness; however, has not been met because there are still parts of the requirement that have yet to be fulfilled.

The final evaluation bucket we look at is the 'Partially Met' one, which only has satisfied correctness, without having achieved consistency or completeness. Just fewer than 10% of the requirements fall under this category, which is pleasing at this stage. This label is given to requirements that have the potential of being met due to the use of the aforementioned technologies; however, it is just the theory behind these technologies that lead us to believe that these problems can be solved. But, in terms of ensuring that the entire requirement can be met across the board without any loopholes or errors, the requirements that fall into this bucket fall short.

6.1 Unmet Requirements

The requirements listed below have been marked as *Unmet* as there is not enough clear documentation on how and where these have been satisfied in the COSMOS project. In this section, we will go through each of the requirements and suggest ways of moving them forward with the aim of meeting them fully.

Three of the unmet requirements are in WP5. The concept of experience sharing has been discussed in depth regarding its usefulness and the benefits it could provide COSMOS. However, no real implementation of an orchestration engine or similar framework has been proposed and therefore cannot be market as a met requirement. Similarly, there has been no mention of how we plan on precisely implementing time synchronization across every COSMOS component and how we plan on maintaining this. Finally, we have discussed in great length the IoT reference architecture and how VEs are structured and fit into the domain model; however, no work has gone into explaining the process of using these GVEs to develop applications.

The other three unmet requirements are in WP6, which deals in the Network of Things. There is no clear explanation of the 'skills' metadata and how we expect to filter across it or use it in a search query. The VE level of the trust and reputation model has been made clear; however, the objective level has not been explained.

Finally, there is no documentation on a Call for Tender feature whereby objects in the COSMOS space can broadcast their needs and XP. This would benefit the communication side of the system and improve the experience sharing features. Our recommendation is that we start looking at ways of advertising these needs and characteristics and attempt to implement them in a use case scenario to test its efficacy.

UNI ID	**Description**
UNI.253	The orchestration engines could support setting preferences for selecting services involved in composition.
UNI.089	COSMOS should support reliable time synchronization.
UNI.245	COSMOS must support creation of new applications through the creation of new GVEs or other mechanisms.
6.14	It must be possible to describe an object skills (and purpose/objective) and to search based on those descriptions
6.16	It could be possible for an object to issue a Call for Tender, in order to advertise its specific needs and get experience-sharing proposals from other objects.
6.37	[Several] Two levels regarding trust & reputation evaluation [could] should be recognised

7 Recommendations

7.1 Overall Recommendations

In this section we will highlight the courses of action we wish to take in the upcoming year, based on our findings in Chapter 5. We aim to objectively suggest areas of COSMOS to focus on further, whilst recommending particular topics to research, concepts to develop further and techniques to continue improving upon. Finally, we look into the requirements that have not been met yet and discuss ways of making them correct, consistent and eventually complete.

Privelets and Node-RED

To achieve end-to-end security over all COSMOS components we utilise the Privelets technology and implement Node-RED as a linking component. We must ensure that all standard protocols are followed in ensure the encryption is secure and that these technologies are not too heavy that it will cause latency in the system.

The Planner and Experience Sharing

Another important recommendation is to find the best way of allowing the VEs to communicate their experiences, not just their raw data or state space. We must find the balance between the speed of having logic done locally and the efficiency of having logic done in the highest level of COSMOS. This is particularly crucial for the implementation of Case Base Reasoning and Experience Sharing. It is also recommended that we are constantly looking to extend the case base so that it can deal with a multitude of different scenarios. The usefulness of this technology heavily depends on the size and diversity of the case base and therefore we must aim to constantly extend and refine it.

There should be an effort made to understand the archetypical cases that may apply for a wide range of applications. For instance VEs that have mobility, VEs that describe environmental conditions and how they may link to generalised actuation plans i.e. change heating, lighting or humidification.

Machine Learning

In terms of analytics, we should compare different Machine Learning techniques for classification and regression for archetypical use case scenarios such that general reuse is possible. Researching many possible ways of modelling our system so that end users can interact with these complex technologies is of paramount importance in COSMOS, as we need these models to make sense to human observers and application developers. There is also great benefit in getting the system to adapt dynamically and improve over time in an unsupervised way.

We should aim to run quality control and bug testing thoroughly on the CEP-based technologies, as this technology may have limitations in large deployments, especially if rule sets are authored by multiple parties.

Furthermore, we should follow a trial and improvement approach when developing the Experience Sharing API, to understand if the best experiences are winning and there are not conflicts in experience ratings that cause poor results.

Practical System Issues

Finally, it is recommended that we research how to make the communication in the Heating Network as reliable and efficient as possible. Issues such as a volatile Internet connection can cause issues such as missing data values and infrequent data transfers. This issue seems to have been either accepted or overlooked, but it is extremely important that we find ways of ensuring the data is regular and complete, as the entire COSMOS platform relies on it or ways of working around data quality become directly addressed by COSMOS.

8 Conclusion

Within Year 2 it is possible to assess technology that is new and improved. Most of the evaluation has been done on either prototypical systems or mostly implemented technologies.

Our evaluation is optimistic for Year 3 in key areas where innovation is occurring namely: The CBR Planner, Privelets and UrbisAPI. Specifically we see

- Using the decentralised approach allows COSMOS to benefit in terms of efficiency. CBR making a big impact in the way low resource devices can become intelligent, that CBR case bases can be exchanged for experience sharing and generalisation for CBR to be widely applied.
- Privelets allow us to authenticate communication between devices, whilst the use of Node-RED allows us to streamline these processes and link many components together in an intuitive way.
- Our next steps for the UrbisAPI platform is to add additional key functionalities of user interface components so that it is ready to be sold to cities for IoT. Further to this, the hardware will be industrialised so that it's durable and compatible with any market and industry. This will facilitate the sales of UrbisAPI to as many cities as possible.

There is clearly more work to be done in Year 3 to integrate COSMOS services and align them to the IoT-A reference architecture. City services are clearly wanting to adopt IoT and having working systems that can realise business processes will have a large impact.

References

[1] FP7 COSMOS Project Deliverable D7.4.1 - Smart heatand electricity management: Evaluation and recommendations (Year 1 Functionality)
[2] FP7 COSMOS Project Deliverable D2.2.2 - State of the Art Analysis and Requirements Definition (Updated)
[3] Introduction to the Architectural Reference Model for the Internet of Things - FP7 IoT-A Project - http://www.iot-a.eu
[4] FP7 COSMOS Porject Deliverable D3.2.1 - End-to-end Security and Privacy: Software prototype
[5] FP7 COSMOS Project Deliverable D4.2.1: Information and Data Lifecycle Management
[6] FP7 COSMOS Project Deliverable D5.1.1: Decentralized & Autonomous Things Management
[7] FP7 COSMOS Project Deliverable D6.1.1: Reliable & Smart Network of Things
[8] Energyhive platform website - http://www.energyhive.com/
[9] Node-RED website - http://nodered.org/

9 Appendix

9.1 Requirements evaluation percentages

The list of the 106 requirements has been already provided in WP2 deliverables. According to the work done the level of compliance with the ones identified are presented in the following table.

Meet fully - Consistent, correct & complete	66	63%
Mostly met - not consistent	4	4%
Mostly met - not correct	0	0%
Mostly met - not complete	19	18%
Partially met - Only consistent	0	0%
Partially met - Only correct	9	9%
Partially met - Only complete	0	0%
Not met	6	6%