D4.2 Urban Sharing Platform Reference Model

Grant Agreement nr.	210271592
Project acronym	SharCities
Project title	Sharing Cities
Funding scheme	Innovation Action
First draft of USP Focus areas	
Due date of deliverable	M12
Main editor (s)	Maurilio Zuccalà
Contributor (s)	WP4 Partners

Project funded by the European Commission within the H2020 Support Programme		
DISSEMINATION LEVEL		
PU	Public	**X**
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium and the Commission Services	

Disclaimer

Statement of Originality

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Document Change Log			
Version	Date	Editor	Summary of Modifications
V01	29/11/2016	Maurilio Zuccalà	Initial TOC
V02	13/12/2016	Maurilio Zuccalà	TOC revised, contents added
V03	19/12/2016	Concirrus	Section 3.2.2: DSA Description Section 3.1.4: Data Layer components description Section 4.2: Greenwich USP description
V04	21/12/2016	Maurilio Zuccalà	Updated Sections 2, 3, 4, 5
V06	22/12/2016	Maurilio Zuccalà	Updated Milan's content
V06	22/12/2016	Vasco Móra	Updated Lisbon's content Section 4.3
R01	23/12/2016	Maurilio Zuccalà	Release version

Executive Summary

This deliverable describes the reference model of the Urban Sharing Platform (USP), as designed in Task 4.2 “Co-Development of the Urban Sharing Platform Reference Architecture”.

This deliverable contains the consolidated description of the USP according to London, Lisbon and Milan standpoint and expertise, enhanced by feedback from other project WPs.

Where possible the USP design aligns with the work of the European Innovation Partnership for Smart Cities and Communities (EIP SCC) Integrated infrastructure Action Cluster – Urban Platform.

Table of Contents

List of Figures

List of Tables

1 Introduction

The Urban Sharing Platform (USP) is an overarching collection of technical components, capabilities, standards, guidelines and processes, which provides functions and services that enable a Smart City.

The main purpose of the USP is to aggregate data and control from a wide variety of devices and sensors, store and process the data, and support visualization of the information to the city and citizens, which enables better use of the city resources.

In particular, the USP aims to:

- Support real-time data collection from field sensors and device.
- Provide components for data storage and business intelligence.
- Provide API-based access to all data and functionalities managed by the platform.
- Support seamless integration of third-party open data and APIs.
- Support people engagement, by enabling the development of dashboards and applications for end-users (e.g., citizens, city managers) to exploit data collected and elaborated through the USP.
- Support proper governance processes;
- Enable federation between different instances of the USP.

The USP enables the creation of federated ecosystems of open, multi-stakeholder service environments, thus enabling digital interoperability between different players to effectively support the smart city concept.

The USP and the related models will develop over the life of the project as the concept of platform sustaining the smart city develops and actual solutions are built and tested.

The remainder of this document describes the common reference model of the USP, as designed in Task 4.2 “Co-Development of the Urban Sharing Platform Reference Architecture” according to the requirements captured in deliverable D4.1 Urban Sharing Platform Requirements.

1.1 Acronyms

API	Application Programming Interface
CCOC	Cloud City Operation Centre
CEP	Complex Event Processing
EIP	European Innovation Partnership

EMS	Energy Management System
ESP	Event Stream Processing
IaaS	Infrastructure as a Service
ICT	Information and Communication Technology
IoT	Internet of Things
IT	Information Technology
KPI	Key Performance Indicator
MQTT	Message Queuing Telemetry Transport
NGSI	Next Generation Service Interface
PaaS	Platform as a Service
REST	REpresentational State Transfer
SaaS	Software as a Service
SCC	Smart Cities and Communities
SDB	Service Delivery Broker
SLA	Service Level Agreement
USP	Urban Sharing Platform
WP	Work Package
XaaS	"X" (everything/anything) as a Service

1.2 References and Supporting Documentation

The following references and supporting documentation are appropriate for this document.

- Sharing Cities: H2020-SCC-2015 SHAR-LLM Proposal, final version.
- Sharing Cities: D4.1 Urban Sharing Platform Requirements, R01.
- Sharing Cities: D4.3 Urban Sharing Platform Relisation, R01.
- EIP SCC: UP Initiatives and Standards Mapping on ICT Urban Platforms for Smart Cities, v3.0.
- EIP SCC: Reference Architecture and Design Principles, v0.62.

- ESPRESSO: D4.2 Definition of Smart City Reference Architecture, rev. 4.
- Triangulum: D6.1 ICT Reference Architecture, Final Version – July 2016.

2 Related Work

The goal of this section is to briefly present a selection of state-of-the-art initiatives, research efforts and solutions, which are of interest for the design and implementation of the Sharing Cities USP. In particular, WP4 partners are in touch with some of these initiatives and closely follow – and in some cases inform – their developments.

2.1 EIP SCC Open Urban Platform

The European Innovation Partnership on Smart Cities and Communities[1] (EIP SCC) is a stakeholder-driven initiative stimulated and supported by the European Commission. It has defined key priority areas which will be addressed through Action Clusters including "Integrated Infrastructures and Open Data". In fact, open urban platforms are a pre-requisite to support fast take-up of smart solutions in smart cities.

In particular, in the context of the Urban Platforms initiative[2] of the Integrated Infrastructures & Processes (including Open Data) EIP SCC Action Cluster, a reference architecture has been proposed to provide cities and communities with a model for the development of a urban platform (goals, requirements, domain entities, use cases, logical model, capabilities etc.). EIP SCC work already takes into account several other initiatives (e.g., the ESPRESSO project) and related technical standards.

The USP reference model described in this document is in line with the current EIP SCC outcomes. WP4 is in touch with the EIP SCC team working on the urban platform reference architecture in order to ensure full alignment and mutual knowledge exchange during the project.

2.2 ESPRESSO

The systEmic Standardisation apPRoach to Empower Smart citieS and cOmmunities[3] (ESPRESSO) project is a two year H2020 project started in 2016. The project focuses on the development of a conceptual Smart City Information Framework based on open standards. This framework will consist of a Smart City platform (the "Smart City enterprise application") and a number of data provision and processing services to integrate relevant data, workflows, and processes.

ESPRESSO and EIP SCC (see Sect. 2.1) reference models are now aligned, so alignment with EIP SCC also implies alignment with ESPRESSO. WP4 will put particular care in analysing and evaluating standards and technologies considered by ESPRESSO.

[1] http://ec.europa.eu/eip/smartcities/
[2] https://eu-smartcities.eu/content/urban-platforms
[3] http://espresso-project.eu/

2.3 Triangulum

Triangulum[4] is a H2020 lighthouse project started in 2015. One of the missions of the project is to develop and implement a smart city reference model.

The USP reference model described in this document is in line with the current Triangulum reference architecture. WP4 plans to evaluate the relevance for Sharing Cities of the work done in Triangulum on the concept of 'service primitives' linked to each architectural layer.

2.4 REPLICATE

The REnaissance of PLaces with Innovative Citizenship And Technology[5] (REPLICATE) project is a H2020 project. The project focuses on energy efficient buildings, ICT and mobility. Each of the core city will develop an ICT smart city platform. The Florence instance of the platform relies on the semantics-based Km4City platform[6].

WP4 is in contact with the city of Florence to mutually exchange views and knowledge about the respective platform models and implementations.

2.5 FIWARE

The FIWARE[7] Community is an independent open community whose members are committed to materialise the FIWARE mission, that is: "to build an open sustainable ecosystem around public, royalty-free and implementation-driven software platform standards that will ease the development of new Smart Applications in multiple sectors".

The FIWARE Catalogue[8] encompasses a rich set of enablers, i.e., components with reference implementations that allow developers to put into effect various functionalities (data management, service ecosystems, security, IoT etc.).

The FIWARE Catalogue is being monitored by WP4 in order to identify and evaluate opportunities of using FIWARE enablers in the Sharing Cities USP.

[4] http://triangulum-project.eu/
[5] http://replicate-project.eu/
[6] http://www.km4city.org/
[7] https://www.fiware.org/
[8] https://catalogue.fiware.org/

3 USP Reference Model

The main objectives of the Sharing Cities USP are:

- Co-design a **shared reference IT architecture** at European level.
- Enable information and functional **sharing** by building an **interoperable federated IT platform**.
- Utilise **API Economy** best practices.

To facilitate the creation of an USP for three cities, a **reference architectural model** is needed that provides a template for each cities design and architecture. It also provides a common vocabulary with which to discuss implementations, aiming to stress commonality and sharing of functions and components.

The following sections describe the reference model of the USP.

3.1 Logical View

The diagram below depicts the overall logical reference architecture of the USP and the related logical components.

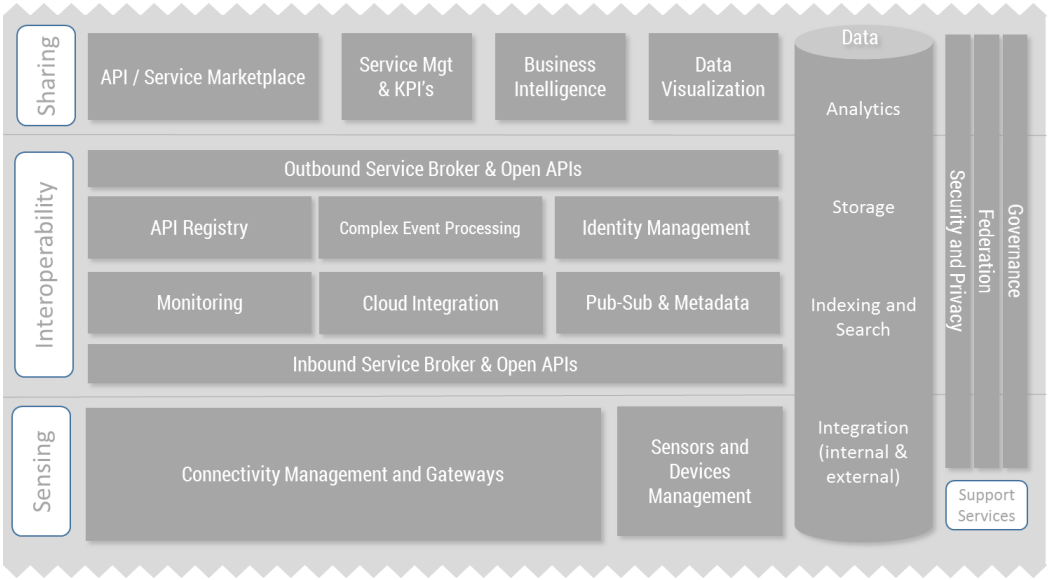

FIGURE 1. USP LOGICAL REFERENCE MODEL

The following logical macro-layer can be identified:

- **Sharing** layer.

- **Interoperability** layer.
- **Sensing** layer.
- **Data** layer.
- **Support services** layer.

Each macro-layer and the related components are described in the following sections.

3.1.1 Sharing layer

This top layer supports the seamless sharing and presentation of platform information and functions to city stakeholders (citizens, city managers, commerce etc.) by a variety of tools: API and service marketplace, business intelligence capabilities, data visualization and dashboards. E.g., end-user applications for citizen engagement developed in WP2 can rely on the services provided by this layer.

The ultimate goal of the Sharing layer is to define standard interfaces and provide end-users with seamless integration with the USP as for data and function access, e.g., let any app developed in one city to be utilised in the other cities with a minimum of changes.

3.1.1.1 API / Service Marketplace

This component is a service catalogue where technical service descriptions are enriched with business oriented information and supports features such as billing, user rating etc.

It relies on the API Registry component of the Interoperability layer (see Sect. 3.1.2.2).

3.1.1.2 Service Management and KPI's

This component provides end-users and USP administrators with proper features to monitor the status and behavior of the services offered by the USP. In particular, it provides specific SLA / KPIs monitoring features.

It relies on the Monitoring component of the Interoperability layer (see Sect. 3.1.2.5).

3.1.1.3 Business Intelligence

This component provides end-users with business logic features operating on data managed by the USP. Such features may include: data analysis (both hot-path and cold-path), machine learning, notification engine.

3.1.1.4 Data Visualization

This component provides end-users with features for building visual representations of the data managed by the USP. E.g., it can support the creation of graphical dashboards. It could also enables the development of 2D, 3D and Augmented Reality user interfaces.

3.1.2 Interoperability layer

This layer provides processing and analytics capability to allow the refinement and consumption of data and information by city or citizen functions. APIs are provided to support standard interfacing with the platform. This layer encompasses also supporting functions such as identity management, service brokering, service monitoring, integration with external cloud-based "XaaS" features etc. Interoperability is supported also cross-cities via a federation gateway to interconnect different instances of USP.

Specific interfaces to the components of this layer may vary in each city because they are tightly linked to the specific tools and solutions used to implement these logical functions.

3.1.2.1 Outbound Service Broker and Open APIs

This component can be considered as a meta-component. I.e., the USP will make data and features available to end-users through API-based open services. Such services will adopt open standards, protocols and data formats for interoperability.

3.1.2.2 API Registry

This component represents a registry of the services offered by the USP. It can be seen as the main entry point of the USP ecosystem from the standpoint of USP consumers, since it provides end-users with a Web interface for browsing the USP registry and search for the different assets available (i.e., APIs, glossaries). Moreover, it provides APIs for browsing the USP assets in the registry.

3.1.2.3 Complex Event Processing

This components supports analysis and tracking of streams of information originated by multiple data sources and related to certain events, internal or external, e.g., data provided in real-time from field sensors. Such processing can suggest or lead to (automated) actions.

3.1.2.4 Identity Management

This component supports identification and authentication of USP users. For this purpose, this component maintains a directory of unique user identifiers together with the related user profiles (e.g., basic set of attributes supporting user categorization).

3.1.2.5 MONITORING

This component aims to track and control the "health" of the USP and of the ecosystem of services it gives access to. Basic monitoring may consist in checking the availability and responsiveness of the USP components. Advanced monitoring can support internal SLAs.

3.1.2.6 CLOUD INTEGRATION

This component provides features for seamless integration of the USP with cloud-based services and with APIs operating in the cloud.

3.1.2.7 PUB-SUB AND METADATA

This component supports topic-based dispatching of information, in the form of simple messages, when some condition occurs in or around the USP.

The publisher, i.e., the provider of information, supplies information about a subject (e.g., availability of certain data) without needing to know anything about the consumers who are interested in that information. The publisher generates this information in the form of messages, called publications that it wants to publish, and defines the topic of these messages.

The consumers of the information, i.e., the subscribers, can create subscriptions that describe the topic that they are interested in. Thus the subscription determines which publications are to be forwarded to the subscribers. Subscribers can make multiple subscriptions.

In the context of the USP, the publisher can be the USP itself, and information to be forwarded is determined by internal events occurring in the platform (e.g., acquisition of new data from the field, availability of a new API). In this case, the subscribers can be USP end-users.

Or, the publisher can be some sensors or devices operating in the field, and the subscriber can be the USP itself. This would give further flexibility in interfacing the USP with the data sources.

3.1.2.8 INBOUND SERVICE BROKER AND OPEN APIS

This functional block offers a single point of contact for publishing and retrieving data in a flexible and standardized way. Real time capabilities are offered to data providers to publish their data according to a set of well-defined protocols and APIs. A Service Broker is able to manage the life cycle of the APIs, monitor them and provide a set of common capabilities (i.e., single sing-on).

The USP should support the following types of interfaces:

- Protocol-based: Industry supported messaging protocols – MQTT, AMQP, COAP (and other industry standards as they become prevalent).

- API-based: Publish API standards – REST, SOAP.
- Preferably some parsers for common legacy device file format data.

The overall principle is to align whenever possible with existing standards and initiatives: EIP SCC – Urban Platform (see Sect. 2.1), ESPRESSO (see Sect. 2.2), FIWARE (see Sect. 2.5), IETF[9] etc. Any new standard to become supported should be an industry standard with wide acceptance on the market.

3.1.3 Sensing layer

This layer supports data collection from the cities sensors and devices (see WP3 interventions), providing connectivity and low level data aggregation when needed. Sensing layer of data feeds for both legacy and new devices, through the use of gateways and directly to more intelligent devices sharing capabilities. It also provides direct integration with city and external data sources that may be open or private in nature.

3.1.3.1 Connectivity Management and Gateways

This component represents the resource interface communication layer composed by all the data providers of Sharing Cities. All existing gateways and devices must comply with APIs or protocols defined by the upper layer. Libraries may be provided to the data providers to ease the integration process.

3.1.3.2 Sensors and Devices Management

This functional block should be able to discover and register resources/devices according to the needs of each data provider, from devices families and hardware versions previously registered in some hardware catalogue. This feature is tightly related to the platform's security requirements, as only registered and approved devices are allowed to ingest data into the USP.

Decoupling data ingestion from the main USP architecture is vital, in order to prevent that new devices and new standards cause constant changes to the core of the platform.

Two principles should support this:

- Abstraction - Increased abstraction further up the stack isolates low level changes.
- Flexibility - An environment should be created to easily 'wire up' new devices.

[9] https://www.ietf.org/

3.1.4 Data layer

This layer is responsible for the persistence of the data, in all its different forms (structured, non-structured, geographic, pair-values etc.), that is used and managed by the USP. It also offers to the other USP components effective and efficient mechanisms to query and retrieve the data, to control the data flow, and to support analytical capabilities of the solution.

3.1.4.1 Analytics

As the USP needs to support predictive capabilities based on Machine Learning solutions to make predictions "as it happens", it is necessary to provide a mechanism for process data in memory as it streams in for live events (in order to take actions in real time), and in parallel, to storing data for long term batch processing (e.g., to rebuild the Machine Learning models to reflect a changing environment).

This architecture (Lambda Architecture) is graphically represented in the diagram below.

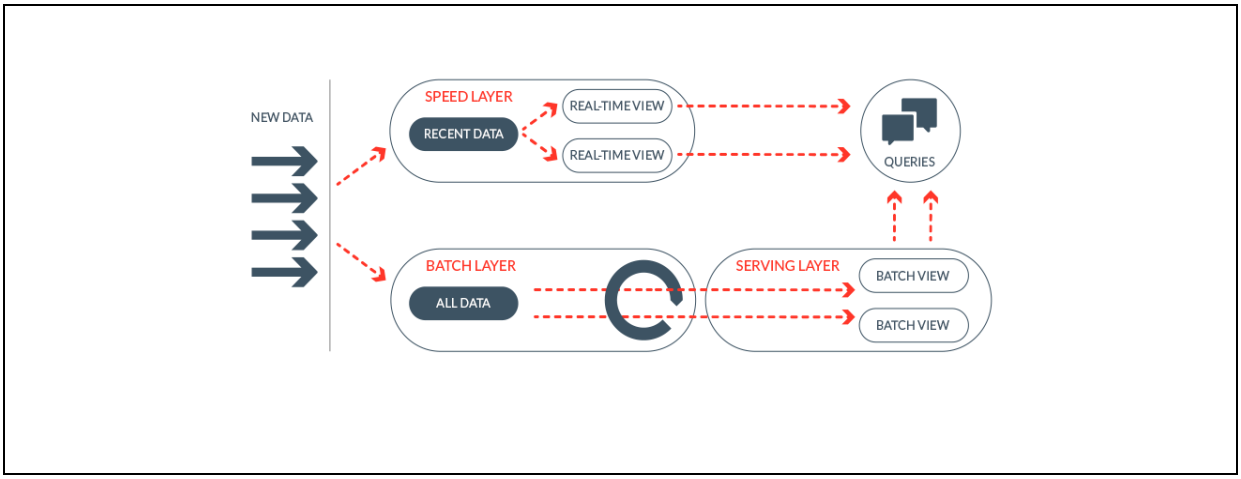

FIGURE 2. LAMBDA ARCHITECTURE

The Analytic Service should also provide a container to deploy different kind of analytic functions into the system and should provide a way to connect the output of these analytic functions into various result streams that can be made available to different visualization elements via the Sharing Layer.

3.1.4.2 Storage

The storage component is commissioned to persist the data that support all the functionalities available in the USP, including, but not limited to: raw event data coming from sensors and gateways through the Sensing Layer, processed events data, imaging, spatial and any other data needed by other USP components.

As within the USP it is necessary to handle different kinds of data sources and data types, the USP needs a polyglot data storage, providing efficient services for, at least, key value pairs, relational data, graphs, geo spatial, imaging, and unstructured data.

The storage service should be designed to support the persistence of every data stage (raw, intermediate and processed), allowing a form of immutability.

Additionally, considering the potential large volume of data that might be produce in a city, the Storage component should be carefully designed to ensure scalability, availability and reliability, in addition to a multi-tier approach, that support hot, warm and cold storage of data.

3.1.4.3 Indexing and Search

In order to manage and use effectively the data contained in the Storage component, it is necessary for the USP to provide an Indexing service that allow applications to query and fetch data quickly enough to offer a good user experience to end-users. As already stated in the Storage component, it is necessary to take into consideration the large amount of data that the USP might handle, so, indexing over clustered or distributed databases might be considered.

Searching service is tightly associated with the indexing functionalities, as it complements the functionality of data retrieval. The searching service should consider the defined Data Ontology as core part of the data retrieval process, and an efficient data caching strategy.

The best way to store and query data would be using time series techniques, this allows us to search and extract meaningful statistics and other characteristics of the data. This also allows for forecasting and predictive analytics that is relevant for the types of scenarios that we will encounter on the project.

3.1.4.4 Integration (internal and external)

Given the polyglot and multi-tier nature of Storage layer it is critical to centralize complexity of data access from rest of the platform by having this virtualized data access layer that hides the location and form of data from various consumers and provides a consistent and unified access to data; this is the objective of the Internal Integration component.

The consumer should not be aware if the requested data is stored inside a cache, a relational database or from archived blobs, all of this must be handled on the server side.

The External Integration component provides resources and tools to ingest data into the platform from external sources, or to make these data available without ingesting, but setting up live connections for real time queries to be used by other USP components. Typical external data sources that will be handled by this component are: weather, traffic, transportation, population etc.

As part of the tools that will be made available, this component must provide support for the following items:

- Data validation workflows / data quality - which implies exception handling workflow.
- Flexible scheduling - Data ingestion can happen on demand, based on predefined schedule or based on events.

- Batch and real time - This layer should be able to deal with high speed and high volume data ingestion.
- Transforms - Allow the transforming data into the Data Ontology defined by USP.

3.1.5 Support services layer

This layer encompasses transversal services that can globally support the operation of the USP.

3.1.5.1 GOVERNANCE

An USP is a multi-stakeholder service environment enabling and fostering data management and interoperability between different parties. In addition to the reference model and technical guidelines, a proper governance model is needed: roles, processes etc. The USP governance guidelines will be developed in the course of the project.

3.1.5.2 FEDERATION

Federation mechanisms will be put in place that enable interoperability between different USP instances (e.g., to support collection of monitoring indicators as defined in WP8).

3.1.5.3 SECURITY AND PRIVACY

Proper guidelines, operating both at technical and process level, will ensure that security and privacy issues are addressed in the USP.

3.2 Focus Modules

Within the scope of the Sharing Cities project a subset of the complete reference model, i.e., three clusters of components have been identified which will best enable sharing and exploit the consortium members capabilities and support their respective aspirations: **Device**, **Data** and **API**.

These three **Focus Modules** are being developed into detailed designs which will provide their respective functions. The Focus Modules will be then translated into a logical architecture and finally solution designs, using common components and context. The relationship between the key design artefacts is shown below.

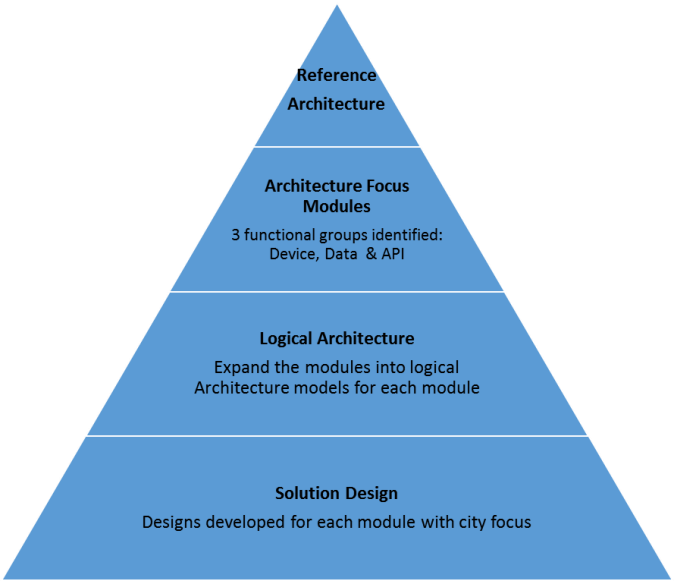

FIGURE 3. USP DESIGN HIERARCHY

WP4 partners reviewed each cities existing platform capability and identified the three areas with the most capability and the best potential for enabling and sharing the USP.

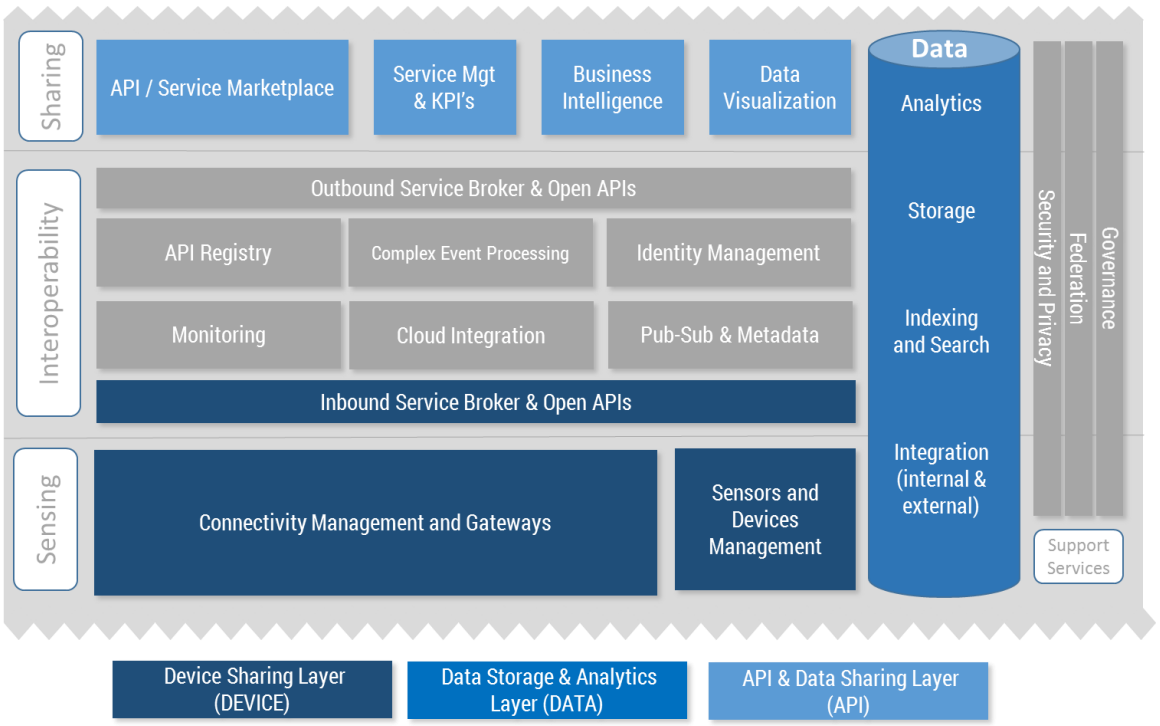

FIGURE 4. FOCUS MODULES OF THE USP REFERENCE MODEL

The **API and Data Sharing Layer** aims to make data collected and processed by the USP, as well as services and functionalities provided by the USP, available to cities consumers via standard APIs in a homogeneous and regulated way. Thanks to this API based framework offered by the USP, city stakeholders can exploit USP features to, e.g.,

- Develop end-user applications supporting citizen engagement.
- Build dashboards that city managers can use for monitoring purposes.
- Support the project monitoring and evaluation framework.
- Build other solutions and tools addressing city needs and requirements in various domains.
- Support a marketplace for APIs and data to generate value for the city.
- Enable application portability and reuse between cities.
- Reduce the cost and time of city application development.
- Provide an open, standards based platform for city applications development.
- Accelerate and reduces cost, complexity and risk of design and procurements by providing a specification for tenders.
- Provide a framework for the development of citizen engagement tools for WP2.

This approach is fully in line with the API Economy trends, in fact it aims to reduce the cost and time of city application development, ease exploitation of integrated heterogeneous data sources, as well as enable application portability and reuse also between cities. Through the API and Service Sharing Layer, the USP ecosystem enables generating tangible value for the smart city.

The **Data Storage and Analytics Layer** is responsible for the persistence of the data, in all its different forms (structured, non-structured, geographic, pair-values etc.), that is used and managed by the USP. This layer should also offer to the other USP components easy to use and efficient mechanisms to query and retrieve the data, to control the data flow, and support analytical capabilities of the solution. Other features include:

- Provide standard data models, ontologies and analytics.
- Enhance the cataloguing and access to data.
- Enable authorised access to datasets and subsets based on an access control policy.
- Enable city stakeholders to make better decisions based on rich data and standard analytics.
- Support the gaining of value from data.
- Enable the sharing of data between cities as determined by data collector.
- Enable efficient transit of data between the Device and API layers including accommodating real-time and historic data.

To enrich the data available in the USP, this layer has integration capabilities that make possible to feed the USP with external data sources, for example, weather data, traffic data, population date, etc. that, jointly with the sensors data ingested through the Device Sharing Layer, will allow city stakeholders to exploit them in apps, dashboards, predicting models and others.

The **Device Sharing Layer** deals with logical connection of any device to the USP in any city. Key features include:

- Connect any device or data source to the platform, which complies to USP specification.
- Include device control, management and feedback loop.
- Support distributed data capture.
- Provide guidelines and specifications for design and procurement including quality, security etc.
- Accelerate and reduces cost, complexity and risk of design and procurements.
- Provide a specification for the procurement of devices by WP3.

The Focus Modules identified above are the foundation of the “Sharing through Architecture” approach adopted in WP4 (see Sect. 4.1). In fact the instantiation of the common USP reference model in each city is taking into particular account the sharing potential of actual tools and solutions, as described in the following Section.

4 Instantiation and Sharing of the USP

This section describes how each core city plans to implement locally – and share with the other cities – the USP reference model described in the previous Section. Details about the design and technical aspects of the implementation are provided in D4.3.

4.1 Sharing through Architecture

As pointed out before in this document, the definition of a common USP reference model plays a key role in achieving actual sharing between cities and partners. In particular, the logical view of reference model is a tool that each city can use in order to:

- Define how they are actually going to implement each logical components, i.e., by relying on which solutions.
- How these solution can be actually shared with other cities, i.e., identify opportunities of collaboration and reuse.

In the remainder of this section each city provides an overview of their USP reference architecture with respect to the logical reference model described in Sect. 3, with a special focus on the main characteristics and licensing scheme of each component.

The following steps for achieving actual sharing of approaches and solutions are already being addressed in the project:

- More detailed technical descriptions of USP components realisation (see D4.3).
- Use of “data capture tables” to identify standards support by sensors and applications (see D4.3).
- Identification of common technical standards to be adopted.
- Link and collaboration with wider initiatives, e.g., EIP SCC – Urban Platform and ESPRESSO (see Sect. 2).

4.2 London / Greenwich

The Greenwich USP is been implemented on the top of two products:

- Concirrus Connect
- NEC Cloud City Operation Centre (CCOC)

The first one, Concirrus Connect, act as the “ingestion layer” to the whole platform. All the data sources (sensors, external live data sources, legacy systems, etc.) will be integrated with Concirrus Connect using the different technical approaches: MQTT, CoAP, API integration etc.; subsequently, Concirrus Connect will stream the received data to NEC CCOC.

The second component, NEC CCOC, will provide all the functionalities for the Interoperability, Sharing, Data and Supporting layers, as described in the USP Reference Model (see Sect. 3).

The integration between Concirrus Connect and NEC CCOC is based on the FIWARE NGSI 9/10 protocol[10]. Concirrus uses this protocol to ingest the data received from all data sources to NEC CCOC, and it manages the data cleaning processes and persistence in one of the storage systems available (see details below).

The diagram below shows a general overview of the Greenwich USP based on those products, as just explained.

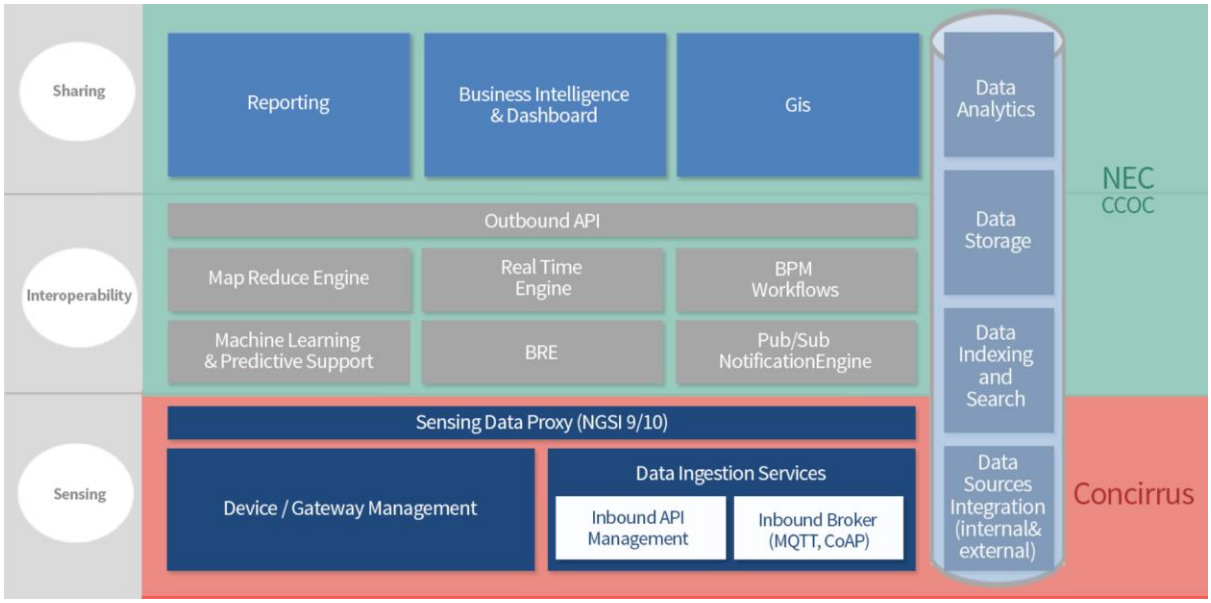

FIGURE 5. OVERVIEW OF THE GREENWICH USP

The different components of the solution are supported by different software packages, in some cases commercial solutions, and in other based on Open Source software.

The diagram below shows the products used for each of the functionalities in the diagram above. The software components in light green are all Open Source software.

It is important to mention that even when using Open Source software, NEC and/or Concirrus have customised those products to meet the specific needs within a smart city scenario. On the other side, is relevant to emphasize that as smart city technologies are in constant change and evolution, this diagram is a permanent "ongoing work". It can be changed at any time, to ensure that the best alternative is offered to the end-user.

[10] https://www.fiware.org/

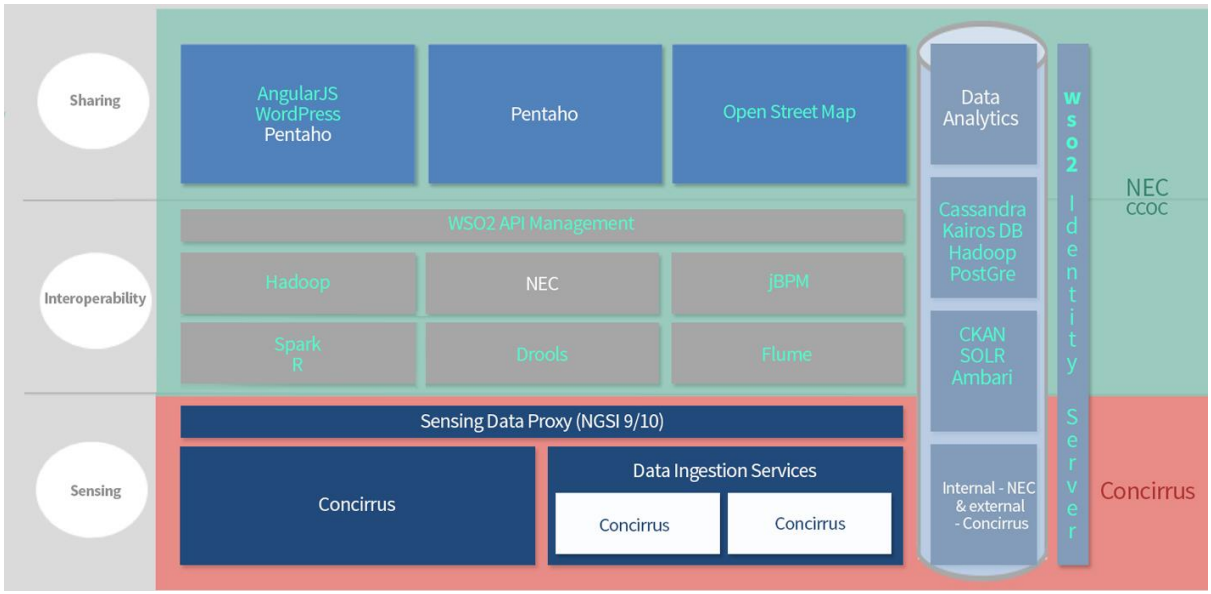

FIGURE 6. GREENWICH USP VS. USP REFERENCE MODEL

A brief description of each component and the associated license model is presented in the table below:

TABLE 1. LIST OF COMPONENTS – LONDON/GREENWICH USP

USP Layer	USP Function	Component	Provider	License
Sharing	BI and Dashboard	Pentaho	Hitachi	Proprietary
Sharing	Visualisation tools	Angular	Google	Open Source
Support Features	Analytic Data Storage	Cassandra	Apache	Open Source
Support Features	Analytic Data Storage	KairosDB	Kairos	Open Source
Support Features	Raw Data Storage	Hadoop	Apache	Open Source
Support Features	Open Data Storage	PostGre SQL	PostGRE SQL	Open Source
Support Features	Open Data Catalog Management	CKAN	CKAN	Open Source
Support Features	Enterprise Search Engine	SOLR	Apache	Open Source
Support Features	Big Data Management	Ambari	Apache	Open Source
Support Features	Identity Server	WSO2 Identity Server	WSO2	Open Source
Sharing	GIS	Open Street Map	Open Street Map	Open Source
Sharing	CMS Module	WordPress	WordPress	Open Source
Interoperability	Business Rule Engine	Drools	Apache	Open Source

Sharing	Reporting Platform	Pentaho	Hitachi	Proprietary
Interoperability	Business Process Management	jBPM	Apache	Open Source
Interoperability	Pub/Sub Notification Engine	Flume	Apache	Open Source
Interoperability	Map Reduce Engine	Hadoop	Apache	Open Source
Interoperability	Real Time Processing Engine	CEP	NEC	Proprietary
Interoperability	Machine Learning support	Spark, R	Apache	Open Source
Sensing	Sensing Data Proxy (NGSI)	Concirrus Connect	Concirrus	TBD
Sensing	Inbound API	Concirrus Connect	Concirrus	Proprietary
Sensing	Inbound Broker	Concirrus Connect	Concirrus	Proprietary
Sensing	Device/Gateway Management	Concirrus Connect	Concirrus	Proprietary

4.3 Lisbon

This section describes a mix of the “as is” situation with the near future platform planned for the city of Lisbon.

Currently, the following products are the basis of the platform but they are not yet integrated:

- SDB – Service Delivery Platform (from Altice Labs)
- Mobi-me (from CEIIA)

SDB is a typical broker for exposing APIs while Mobi-me is a Business Intelligence and Reporting tool.

More in detail, these and other open source components are represented in the following reference architecture picture and are described in following subsections.

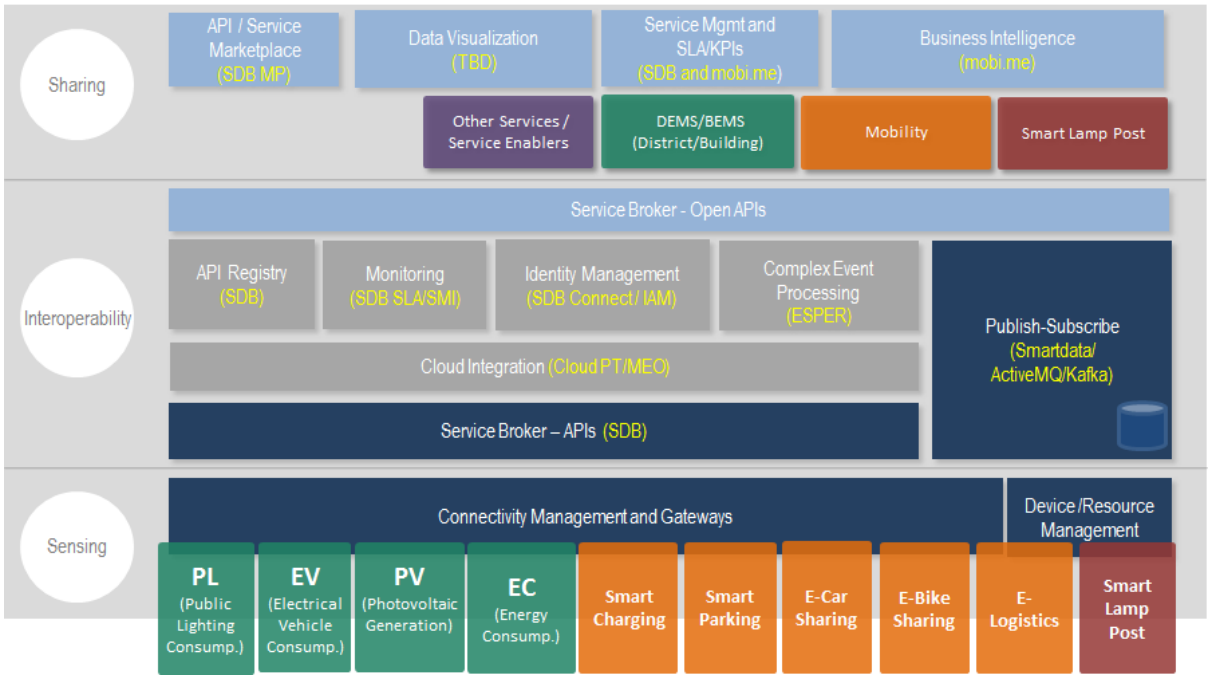

FIGURE 7: LISBON USP TECNHOLOGIES

Figure 7 lists (in yellow) the following list of components:

- Service Delivery Broker Suite which contains:
  - SDB Marketplace
  - SDB (outbound and inbound service brokers)
  - SDB Connect
  - SDB Monitoring
  - SDB CEP (ESPER)
- Pub/sub Broker - Smartdata
- MEO Cloud
- Business Intelligence - Mobi.me
- Visualisation – To be defined (there are already several proprietary solutions available but the city of Lisbon will be using an open source tool as a basis for visualisation).
- Data Sources/Gateways (layer of integration between WP4 and WP3) – APIs will be defined according to the available sensing resources identified in WP3.

4.3.1 SDB Suite

4.3.1.1 SDB Marketplace

The aim of SDB Marketplace is to provide an Ecosystem that would facilitate the deployment and distribution of APIs and Apps for business applications through a Web friendly portal[11]. MO-BIZZ marketplace is the ideal solution even to small companies who usually develop innovative business apps (e.g., SMEs and start-ups). It is imperative that they achieve quick time-to-market and reduce time and resources on deployment and infrastructure support. These types of companies can easily become partners of MO-BIZZ and on-board online their solutions. As soon as they are accepted by MO-BIZZ their new apps and/or APIs will be available to their clients in the marketplace

4.3.1.2 Service Delivery Broker (SDB)

Service Delivery Broker (SDB) is a proven, reliable, standard-base and cloud-scalable SOA (Service-Oriented Architecture) product's suite that comprises SDB Runtime, SDB Backoffice, SDB Support Services, SDB Marketplace and SDB Connect products:

- SDB Runtime provides mediation between service enablers and their consumers (applications or other service enablers) providing a loosely coupled, highly distributed integration network platform, with enterprise-strength performance, scalability and manageability. Combining messaging exchange, data transformation, intelligent routing and many several transversal functionalities, to reliably connect and coordinate service enablers according with the services' management decisions.
- SDB Backoffice is a Web application that provides a user interface for the services' lifecycle management, product lifecycle management, access management and SDB Runtime parameterization;
- SDB Support Services are a set of SOA services that support most of the product logic and therefore can be replaced or extended by other SOA services that implement the interface or easily integrate the SDB with other applications.
- SDB Marketplace is a Web application that gives the operator the possibility to sell applications and services to third parties with a suitable business model.
- SDB Connect manages identity and access permissions, addressing two main scenarios: API-based third-party application ecosystems, federation and interoperability with multiple identity sources.

4.3.1.3 SDB Connect

SDB Connect offers a single-sign-on capability by providing the following functionalities:

[11] https://market.MO-BIZZ-project.eu/en/

- API Registry
- API Life Cycle Management
- Basic ID Management

SDB Connect delivers advanced identity and access management features to address two main scenarios:

- API based third-party application ecosystems; it concludes a comprehensive OAuth 2.0 Authorization Protocol solution, providing functionalities such as:
    - Client application management (via the SDB Backoffice Application);
    - An OAuth 2.0 Authorization Server for access token issuance;
    - Integration of access token processing into the SDB API Gateway.
- Federation and interoperability with multiple identity sources; for this purpose SDB Connect provides an Identity Gateway to mediate and manage the connections with external identity sources, supporting multiple identity providers, credential stores, interaction protocols and security token formats.

4.3.1.4 SDB MONITORING (SMI/SLA)

The SDB monitoring module supports different tasks:

- Service and API provisioning – When a product of a SDB Marketplace product offer is purchased, it is often necessary for the provider of that product offer – the service provider – to be notified so that it has a chance to provision the necessary items for the product. Product provisioning is requested by the SDB Marketplace through a call to the SMI endpoint of the service provider. The service provider is responsible for performing all the steps that are necessary to make the product available to the customer.
- Monitoring and configuration – By means of a SMI (Service Management Interface) to manage cloud / digital service or application it a) performs a set of operations to allow activation and deactivation of services and check their health status; b) supports and reports event-driven service failures.
- SLA Management – A generic SLA management system for the Java programming language that automates many domain independent tasks such as template publishing, offer validation, agreement creation and monitoring, and guarantee evaluation.

4.3.1.5 SDB COMPLEX EVENT PROCESSING (CEP)

SDB CEP (as is) is based on ESPER which is an open-source Java-based software product for Complex Event Processing (CEP) and Event Stream Processing (ESP) that analyses series of events for deriving conclusions from them.

ESPER enables rapid development of applications that process large volumes of incoming messages or events, regardless of whether incoming messages are historical or real-time in nature. ESPER filters and analyses events in various ways, and respond to conditions of interest. ESPER is not limited to running on a single

machine and can run in any architecture and any container, as it has no dependencies on external services and does not require any particular threading model and does not require any external storage.

4.3.1.6 SMARTDATA (PUBLISH/SUBSCRIBE)

The Smartdata platform is a service oriented architecture aligned with principles of service provisioning approach (offer and reutilization of services) from the TM Forum Service Delivery Framework, based on consumer-producer concepts where multiple processes synchronization is crucial. It provides a multi-tenant and open platform to collect, enquire, route and process information produced in an IoT domain. A Consumer is the entity for whom the information is relevant (i.e., a mobile application); a Producer is any entity that produces information (i.e., a sensor).

4.3.1.7 MEO CLOUD

MEO Cloud is Altice/Portugal Telecom's business cloud solution, encompassing an array of products and services that leverage Portugal Telecom's expertise in the Information Technology and Information Systems industry. As a cloud platform it includes IaaS, PaaS and SaaS components in its architecture. Therefore, partners deploying their applications can make use of the MEO Cloud infrastructure according to their needs.

4.3.1.8 LIST OF COMPONENTS

The following table summarizes the characteristics of the components of the Lisbon USP and maps them to the USP reference model.

TABLE 2. LIST OF COMPONENTS – LISBON USP

USP Layer	USP Function	City	Component	Provider/Responsible	WP	Status	License
Service	Marketplace	Lisbon	SDB Marketplace	Altice Labs	WP4	As-Is	Proprietary
Service	Service Mgmt and SLA	Lisbon	SDB Monitoring	Altice Labs	WP4	As-Is/to-be	Proprietary
Service	Busines Intelligence	Lisbon	mobi.me	CEIIA	WP4	As-Is	Proprietary
Service	Dashboard	Lisbon	Dashboard	CML	WP4	To-be	Proprietary
Interoperability	Inbound and Outbound Brokers	Lisbon	SDB API Gateway	Altice Labs	WP4	As-Is	Proprietary
Interoperability	Identity Management	Lisbon	SDB Connect	Altice Labs	WP4	As-Is	Proprietary
Interoperability	Monitoring	Lisbon	SDB SMI	Altice Labs	WP4	As-Is	Proprietary
Interoperability	CEP	Lisbon	SDB CEP	Altice Labs	WP4	As-Is	Open
Interoperability	Publish/Subscribe	Lisbon	Smartdata	Altice Labs	WP4	As-Is	Open
Interoperability	Cloud integration	Lisbon	MEO cloud	altice Labs/PT	WP4	As-Is	Proprietary
Interoperability	Data Storage	Lisbon	Smartdata DB/ Big Data	Altice Labs	WP4	As-Is	Open
Service/Interoperability	Data Analytics	Lisbon	Data Analytics	Altice Labs	WP4	As-Is	Open
Sensing	Connectivity Mgmt and GTWs	Lisbon	Energy Consumption	EDP	WP4/WP3	As-Is/to-be	Proprietary
Sensing	Connectivity Mgmt and GTWs	Lisbon	EV Location & Consumption	EDP, EMEL, CEIIA	WP4/WP3	As-Is/to-be	Proprietary
Sensing	Connectivity Mgmt and GTWs	Lisbon	Public Lighting	EDP	WP4/WP3	As-Is/to-be	Proprietary
Sensing	Connectivity Mgmt and GTWs	Lisbon	Network	EDP	WP4/WP3	As-Is/to-be	Proprietary
Sensing	Connectivity Mgmt and GTWs	Lisbon	Mobility	EMEL	WP4/WP3	As-Is/to-be	Proprietary
Sensing	Connectivity Mgmt and GTWs	Lisbon	Bike Sharing	EMEL	WP4/WP3	As-Is/to-be	Proprietary
Sensing	Connectivity Mgmt and GTWs	Lisbon	Parking	EMEL	WP4/WP3	As-Is/to-be	Proprietary
Sensing	Connectivity Mgmt and GTWs	Lisbon	City Info	CML	WP4/WP3	As-Is/to-be	Proprietary

4.4 Milan

The Milan USP serves as the "nervous systems" of the interventions that the project will implement in Milan's smart district "Porta Romana / Vettabbia" (see Sharing Cities project proposal).

The design of Milan's USP relies on the following criteria:

- Leverage local infrastructures and investments, already in place or planned in the short, medium and long term.
- Enhance and evolve USP components.
- Ensure that such evolution is continuous and sustainable over time, e.g., also beyond the end of the Sharing Cities project.

In the light of the criteria mentioned above, the pillars of Milan's USP are:

- Siemens IT's **Monet EMS** solution.
- Milan's **Interoperability Platform** managed by the Municipality.
- The **E015 digital ecosystem**[12].

Such pillars are actual solutions already operating before – and independent from – the Sharing Cities project. Milan partners will properly integrate and enhance such resources in order to address project goals.

In addition to actual IT solutions, it is worth mentioning some inspiring principles that inform the realisation of Milan's USP as well:

- API Economy, i.e., the approach based on APIs and microservices.
- The support to various interaction patterns to address the different communication needs towards/from the platform and provide better scalability, dynamic network topology etc., in particular the push/pull and publish-subscribe patterns.
- The federation mechanisms that will enable interoperability between different USP instances.

The following figure summarizes the overall logical structure of Milan's USP.

[12] http://www.e015.regione.lombardia.it/PE015/ (in Italian).

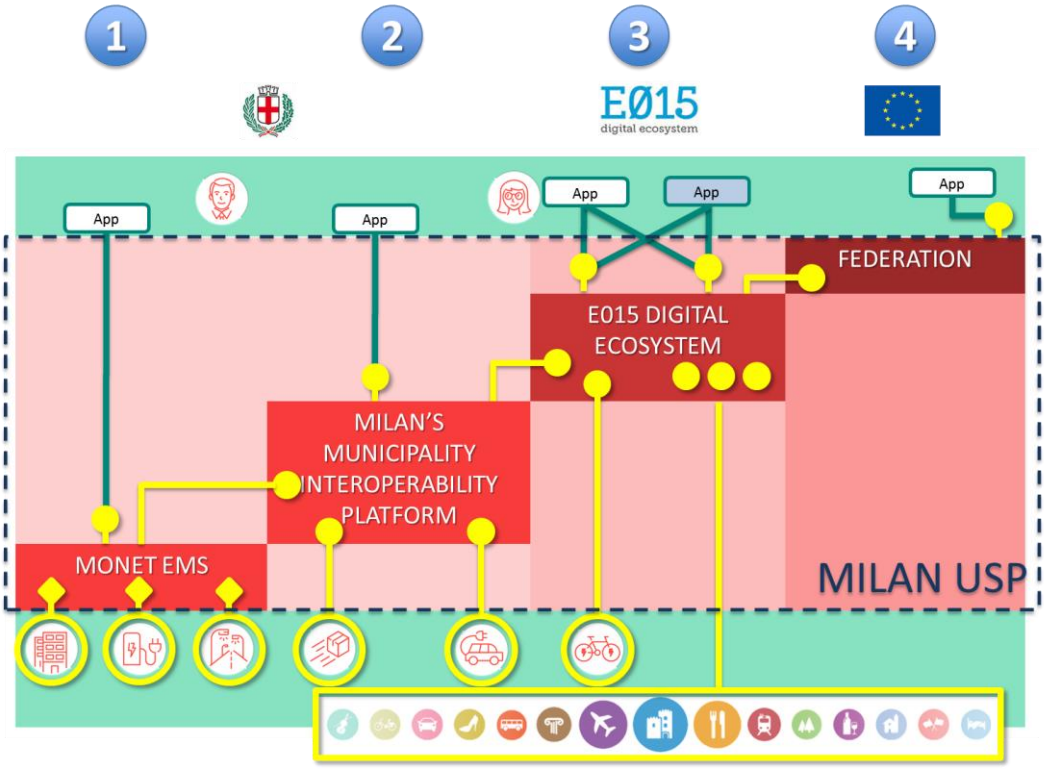

FIGURE 8. MILAN USP LOGICAL STRUCTURE

Four different vertical "stages" can be identified, that incrementally realize the overall USP picture:

1. The **Monet EMS** component by Siemens IT can integrate different data sources deployed in the field (e.g., sensors and devices), in particular the ones related to smart interventions in private and public buildings. Monet in fact is able to support and adapt specific IoT communication protocols to collect data according to different interaction patterns (e.g., MQTT), store and elaborate data, expose data via standard REST API to the following USP stages. Monet also supports visualization features for end-users (e.g., dashboards for city managers) regarding data acquired from the field and managed.
2. The **Interoperability Platform of the Municipality of Milan** enables, fosters and governs interoperability at the IT level within the Administration and at City level. The platform is composed of a set of components – most of which are open source – able to collect and process different kinds of information from different sources, internal or external. The platform enables the realization and management of different kinds of APIs. In particular, this platform can integrate and manage as APIs both the data sources conveyed by the Monet EMS and other data sources from project interventions that can be exposed natively as APIs (e-vehicles, logistics etc.). The platform also provides vital features such as identity management, business intelligence etc. (see below). APIs managed by this platform enable the realization of added-value applications for the end-users.

3. The **E015 digital ecosystem** enables API-based interoperability at regional/national level. E015 is a multi-stakeholder digital ecosystem that exploits the notion of API economy, which provides full, bidirectional and direct interoperability among autonomous distributed applications that access and exploit shared data-driven services. The APIs realized in Sharing Cities and managed through the Interoperability Platform of the Municipality of Milan can be exposed to E015 in order to reach a wider group of real-world potential users (e.g., start-ups or organization willing to adopt Sharing Cities data and features to develop innovative solutions for the end-users). In turn, the wide set of APIs already available in E015[13] can be exploited by Sharing Cities partners in order to enrich information and knowledge base for building project solutions.
4. In order to enable **federation** between different USP instances (starting from the USPs of the three core cities) proper components and mechanisms will extend the E015 digital ecosystem. Potential users of this feature are WP2 and WP8 partners.

The diagram below gives an overview of how the solutions mentioned above support the USP logical reference model.

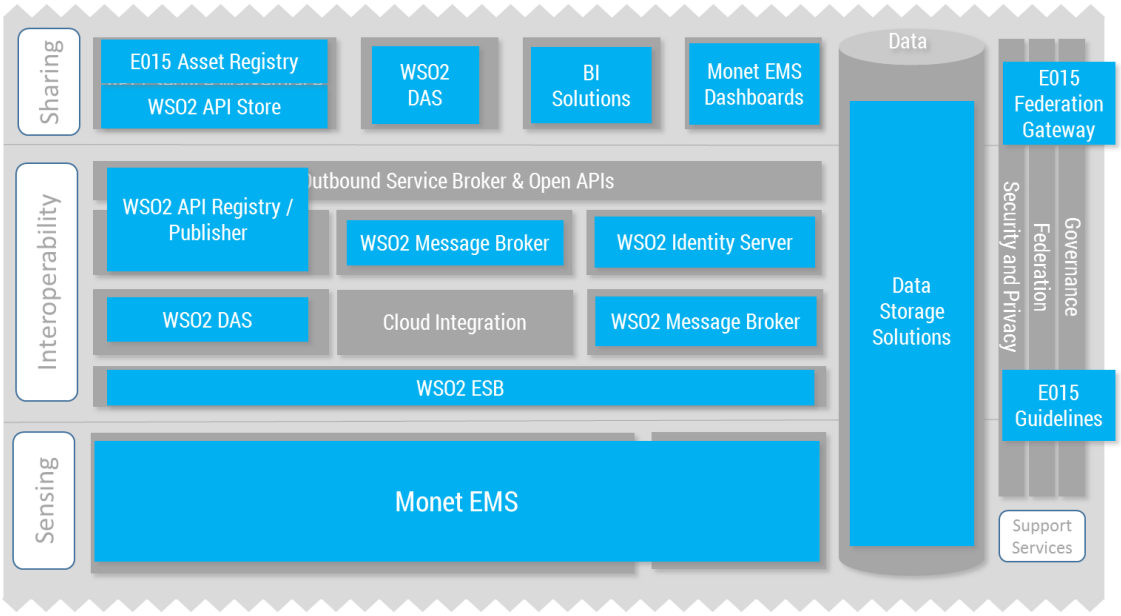

FIGURE 9. MILAN USP VS. USP REFERENCE MODEL

[13] http://www.e015.regione.lombardia.it/PE015/esplora-i-contenuti/i-servizi (in Italian).

The components of the Milan USP are supported by different software tools, open source or commercial. A brief summary of each USP component and the associated information is presented in the table below.

TABLE 3. LIST OF COMPONENTS – MILAN USP

USP Layer	USP Function	Component	Provider/Resp.	License
Sharing	API / Service Marketplace	WSO2 API Store	CdM	Open Source
Sharing	API / Service Marketplace	E015 Asset Registry	Cefriel	Proprietary
Sharing	Service Management & KPI's	WSO2 Data Analytics Server	CdM	Open Source
Sharing	Business Intelligence	*CdM BI Solutions (see D4.3)*	CdM	Proprietary
Sharing	Data Visualization	Monet Dashboards	Siemens IT	Proprietary
Interoperability	Outbound Service Broker & Open APIs	WSO2 API Registry / Publisher	CdM	Open Source
Interoperability	API Registry	WSO2 API Registry / Publisher	CdM	Open Source
Interoperability	Complex Event Processing	WSO2 Message Broker	CdM	Open Source
Interoperability	Identity Management	WSO2 Identity Server	CdM	Open Source
Interoperability	Monitoring	WSO2 Data Analytics Server	CdM	Open Source
Interoperability	Pub-Sub & Metadata	WSO2 Message Broker	CdM	Open Source
Interoperability	Inbound Service Broker & Open APIs	WSO2 Enterprise Service Bus	CdM	Open Source
Sensing	Connectivity Management and Gateways	Monet EMS	Siemens IT	Proprietary
Sensing	Sensors and Device Management	Monet EMS	Siemens IT	Proprietary
Data	Storage, Analytics etc.	*Storage solution to be identified*	CdM	-
Support Services	Governance	E015 Process and Technical Guidelines (Extended)	Cefriel	Open Source
Support Services	Federation	E015 Federation Gateway	Cefriel	-
Support Services	Security and Privacy	*Addressed by all USP components*	*ALL*	-

More information about the components of the Milan USP can be found in D4.3.

5 Conclusions and Next Steps

This document presented the current reference model of the Sharing Cities USP and how cities are instantiating it to support the smart city interventions encompassed by the project. This is a “living document” that will be kept updated during the life of the project in order to acknowledge evolutions of the model, of the state-of-the-art and of the solutions developed in each city.

Next steps of WP4 activities regarding the USP reference model definition, instantiation and sharing include the following activities:

- Detailed technical descriptions of USP components design and realisation, including interfaces.
- Identification of the specific USP stakeholders.
- Monitoring of the “sharing through architecture” approach.
- Selection of common technical standards to be adopted.
- Link and synergic collaboration with related European initiatives.
- Definition of a reference glossary of terms related to the USP.