IRIS

Integrated and Replicable Solutions for Co-Creation in Sustainable Cities

Project Acronym:	IRIS
Project Full Name:	Integrated and Replicable Solutions for Co-Creation in Sustainable Cities
Grant Agreement:	No 774199
Project Duration:	5 years (starting 1 October 2017)

Deliverable 4.5

Implementation and integration of core CIP components

Work Package:	WP4: City Innovation Platform (CIP)
Task:	T4.4 Implementation and integration of core CIP components
Lead Beneficiary:	NCA
Due Date:	31 December 2019 (M24)
Submission Date:	31 December 2019 (M27)
Deliverable Status:	Final
Deliverable Style:	Other
Dissemination Level:	PU
File Name:	D4.5 - Implementation and Integration of core CIP components.pdf

This project has received funding from the European Union’s Horizon 2020 research and innovation program under grant agreement No 774199

Authors

Surname	First Name	Beneficiary
Roux	Stephane	NCA
Boufares	Faycal	NCA
In the Implementation and integration of core CIP components contributed NCA, CIV, CERTH (Panagiotis Tsarchopoulos and Maria Koutli), UTR, GOT and METRY.		

In case you want any additional information or you want to consult with the authors of this document, please send your inquiries to: irissmartcities@gmail.com.

Reviewers

Surname	First Name	Beneficiary
Chateau	Alain	NCA
Bontekoe	Eelke	UU
Lampropoulos	Ioannis	UU
Arjen	Hof	CIV

Version History

Version	Date	Modifications made by
0.1	April 26th 2019	First draft created by Nice
0.2	May 20th 2019	Review &1 Introduction &2.2 Methodology
0.4	June 12th 2019	Organization of chapters 2 & 3 with all the technical components
0.4A	June 18th 2019	Update 0.4 after internal NCA review
0.5	June 28th 2019	SCO approval of the organization / presentation of the document
0.6	Sept 13th 2019	End writing of chapter 2 & 3
0.7	Sept 20rd 2019	End writing of chapter 4 & 5
0.8	Sept 28th 2019	Final draft for NCA Review
0.9	Oct. 28th 2019	New version based on the reviewers' comments
1.0	December 2019	Final version

Disclaimer

This document reflects only the author's view. Responsibility for the information and views expressed therein lies entirely with the authors. The Innovation and Networks Executive Agency (INEA) and the European Commission are not responsible for any use that may be made of the information it contains.

Executive Summary

The City Innovation Platform (CIP) collects, manages and exchanges data for the development of new applications and services. The CIP and its components will manage large volumes of data and information coming from standard protocol or vertical solutions.

It is necessary to accompany the functional guidelines by technical guidelines intended for the IT teams to facilitate the deployment and the evolution of the architecture in different technological and infrastructural environment including use cases from the Lighthouse cities.

The document is divided in 3 parts. This distribution is reflected in the document within setting up chapters dedicated to each part.

Part 1: Chapter 2 Methodology

Being general, this part is readable by anyone wishing to have a first technical and methodological draft of the implementation of the CIP platform.

This part of the document presents the method implemented to integrate the different FIWARE software bricks, selected to build a CIP platform

It also aims to present Open Source Alternatives to some FIWARE software bricks to allow a broader view of the possibilities offered.

Part 2: Chapter 3 CIP Architecture - Presentation

This more technical part is intended for more experienced people reading technical information and using scripting languages.

It focuses on detailing for each tool implemented, its purpose and the reasons for its choice.

It also provides links to FIWARE standard installation and / or procedure and also specific links for the components that will be part of the CIP.

Part 3: Chapter 4 Implementation procedure

It summarizes the deliverables of this task - for the IRIS project. These deliverables are the basis to perform the linked tasks.

Table of Contents

EXECUTIVE SUMMARY 3
TABLE OF CONTENTS 4
COPYRIGHT 7
LIST OF FIGURES 7
LIST OF TABLES 7
LIST OF ABBREVIATIONS AND ACRONYMS 8
LIST OF TERMS – ABBREVIATIONS AND DEFINITIONS 9
1. INTRODUCTION 11
1.1. PREREQUISITES 11
1.2. SCOPE, OBJECTIVES AND EXPECTED IMPACT 11
1.3. CONTRIBUTIONS OF PARTNERS 11
1.4. RELATION TO OTHER ACTIVITIES 12
1.4.1. PREVIOUS RELATION 12
1.4.2. JOINT RELATIONSHIP DURING THE EXECUTION OF ACTIVITIES 12
1.4.3. NEXT RELATION 12
1.5. STRUCTURE OF THE DELIVERABLES 13
2. METHODOLOGY 14
2.1. GLOBAL APPROACH 14
2.1.1. IMPLEMENTATION SCENARIO WITH FIWARE TOOLS 15
2.1.2. IMPLEMENTATION SCENARIO WITH COMBINATION ALTERNATIVE AND FIWARE TOOLS 15
2.2. CIP ARCHITECTURE – HIGH-LEVEL PRESENTATION 16
2.2.1. IMPLEMENTATION PRINCIPLES 16
2.2.2. SYNTHETIC PRESENTATION OF THE TECHNICAL COMPONENTS 17
2.2.2.1. IDM & AUTHZFORCE (SECURITY & PRIVACY FRAMEWORK) 17
2.2.2.2. API MANAGEMENT 17
2.2.2.3. CONNECTORS 17
2.2.2.4. IOT AGENTS - DEVICE MANAGEMENT - EXCHANGE FORMAT - IDAS 18
2.2.2.5. DATA BASE STORAGE AND PUBLICATION 18
2.2.2.6. DATA VISUALIZATION / ANALYZING / MONITORING 18
2.2.2.7. GIS SERVER 19
2.2.2.8. ACCOUNTING / PAYMENT / BILLING 19

2.2.3. MAPPING TOOLS TO IMPLEMENT AT LEAST FOR A CIP ARCHITECTURE / FIWARE CONCEPTS 19

2.2.4. IRIS ARCHITECTURE 20

3. CIP ARCHITECTURE – PRESENTATION 21

3.1. *DESCRIPTION OF THE STUDIED TOOLS* 21

3.1.1. ORION – CONTEXT BROKER COMPONENT 22

3.1.2. IDM & AUTHZFORCE - SECURITY & PRIVACY 23

3.1.2.1. IDM - KEYROCK 25

3.1.2.2. AUTHZFORCE 26

3.1.2.3. PEP-PROXY 27

3.1.2.4. IDM – KEYCLOAK 27

3.1.3. API MANAGEMENT 29

3.1.3.1. APINF 29

3.1.3.2. API UMBRELLA 30

3.1.3.3. 3SCALE 30

3.1.4. CONNECTORS 31

3.1.4.1. FIWARE - QUANTUM LEAP 31

3.1.4.2. FIWARE - CYGNUS 32

3.1.4.3. FIWARE-STH COMET 32

3.1.5. IOT AGENTS - DEVICE MANAGEMENT - EXCHANGE FORMAT 33

3.1.5.1. IOT AGENT NODE LIB 33

3.1.5.2. JSON – IOT AGENT JSON 34

3.1.5.3. UL – IOT AGENT UL 34

3.1.6. DATABASE – STORAGE AND PUBLICATION 35

3.1.6.1. CRATEDB 35

3.1.6.2. MONGODB 36

3.1.6.3. ELASTICSEARCH 37

3.1.6.4. LOGSTACH 37

3.1.6.5. POSTGRES 38

3.1.6.6. REDIS 39

3.1.6.7. CKAN 40

3.1.7. DATA VISUALIZATION / ANALYZING / MONITORING 41

3.1.7.1. KIBANA 41

3.1.7.2. GRAFANA 41

3.1.8. GEOGRAPHICAL TOOLS 42

3.1.8.1. 3D CITY DB 42

3.1.8.2. GEOSERVER 43

3.1.9. ACCOUNTING / PAYMENT / BILLING 44

3.1.9.1. BIZ ECOSYSTEM 44

4. IMPLEMENTATION PROCEDURE 46

4.1. INTRODUCTION 46

4.2. LAUNCHING THE INSTALLATION PROCEDURE 46

4.3. TEST CASE 48

5. CONCLUSIONS 49

6. REFERENCE 50

6.1. *LIST OF HYPERTEXT LINKS USED IN THE DOCUMENT* 50

Copyright

List of Figures

Figure 1 - Implementation approach with FIWARE Tools 15
Figure 2 - Implementation approach with FIWARE and Alternate Tools 15
Figure 3 - FIWARE Platform: Technical Architecture 16
Figure 4 - CIP Architecture - Technical components implemented during IRIS project 16
Figure 5 - CIP Architecture – Minimum tools to run a smart city application 19
Figure 6 - CIP Architecture – Tools implemented in the IRIS project 20
Figure 7 - Template used to describe the studied tools 21
Figure 8 - Orion Context Broker relation with other components 23
Figure 9 - Level 1: Authentication only (Fiware Account is Keyrock IDM) 24
Figure 10 - Level 2: Authentication & Basic Authorization 25
Figure 11 - Level 3: Authentication & Advanced Authorization (using XACML). 25
Figure 12 - Keycloak - Identification process – Directory Service 28
Figure 13 - Keycloak – Identification process - Social network 28
Figure 14 - Keycloak - Standard protocols 29
Figure 15 - Fiware Business Ecosystem Architecture 45
Figure 16 - Implementation procedure : Main menu 47

List of Tables

Table 1 - Abbreviation 8
Table 2 - Definitions 10
Table 3 - Methodology - Global Approach 14
Table 4 - List of security requirements (ref to D4.2 - Figure 22 PMRM services / Page 22 /) 24

List of Abbreviations and Acronyms

Abbreviation	Definition
ADE	Application Domain Extension
API	Application Programming Interface.
BIM	Building information model
CIM	City Information Model
CIP	City Innovation Platform
CKAN	Comprehensive Kerbal Archive Network
CVS	Computer Vision Syndrome
DCAT-AP	Data Catalogue Vocabulary -Application Profile
DPA	Data Protection Authority
DPO	Data Protection Office
EIP-SSC	European Innovation Partnership on Smart Cities and Communities
EU	European Union
FIWARE	Future Internet-ware
GDPR	General Data Protection Regulation
GML	Geography Markup Language
IAM	Identity & Access Management
IoT	Internet of Things
JSON	JavaScript Object Notation
OUP	Open Urban Platforms (also working group of EIP-SCC)
PbD	Privacy by Design
PEAR	Privacy Enhancing ARchitecture
PET	Privacy Enhancing Technology
PIA	Privacy Impact Assessment
PMRM	Privacy Management and Reference Model and Methodology
PRIPARE	Preparing Industry to Privacy-by-design by supporting its Application in Research
RDF	Resource Description Framework
SLA	Service Level Agreement
USEF	Universal Smart Energy Framework
WP	Work Package
XML	Extensible Mark-up Language

Table 1 - Abbreviation

List of Terms – Abbreviations and definitions

Abbreviation	Definition
API	A software intermediary that allows for distinct applications or systems to interact with one another.
BIM	Also called object-based model is a three-dimensional model with data information.
Capability	The abstract representation of what is needed to produce an outcome along with goals and metrics for that outcome.
CIP-component	The following five components of the City Innovation Platform: 1. Data management framework 2. Data market 3. Security and privacy 4. Platform management Proprietary systems connectivity (federated solution)
CVS	This is a program that lets a code developer save and retrieve different development versions of source code. It also lets a team of developers share control of different versions of files in a common repository of files
CKAN	This is an Open source catalogue system for open data portals.
Data Portal	A software solution (usually a website) that presents a catalogue of searchable and downloadable datasets in a user-friendly and uniform way.
Data Set	A collection of data that can be downloaded and processed further.
DCAT-AP	This is an RDF vocabulary designed to facilitate interoperability between data catalogues published on the web. The Application Profile is developed by the European Commission for interoperability optimisation between European Data Portals
ESPRESSO	EU-project (2016-2017) that identified a collection of open standards for smart cities that work well together ("conceptual standards framework") and have been proven
FIWARE	"This is an open software- and standards framework
Functional requirement	Functional requirements describe the desired end function of a system to assure the design is adequate and meets user expectations. In this document also used as "Design Principles".
Interoperability	The ability of different information technology systems and software applications to communicate, exchange data, and use the information that has been exchanged.
JSON	This is a lightweight data-interchange format
Linked Data	A method of publishing structured data so that it can be interlinked and become more useful through semantic queries, facilitating the sharing of machine-readable data on the web.
Metadata	Data about data

Open data	Data carrying an open licence stating it can be freely used, re-used and redistributed by anyone, for any purpose.
Open & Agile Smart Cities	Open & Agile Smart Cities (OASC) is a global initiative connecting cities, advocating de facto standards, and sharing best practices.
Reference architecture	A template that offers a common language and support for standards, specifications and patterns, a list of functions and interfaces (APIs) and their interactions with each other.
RDF	This is a standard model for data interchange on the web.
Role	Responsibility and/or activity of a stakeholder within the CIP
Stakeholder	A person or group with specific interest in the CIP.
Technical requirement	Technical requirements define what is required to deliver the desired function or behaviour from a system to a user's standards.
TM-Forum	Is a neutral, non-profit member organization.
USEF	This is an international common standard that ensures smart energy technologies and projects are connectable at lowest cost.
XML	This is a mark-up language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable.

Table 2 - – Abbreviations and definitions

1. Introduction

1.1. Prerequisites

To fully understand the content of this document and benefit the most of it, it is recommended to read first the following documents:

- D4.2: Document describing the functional aspects of the architecture in accordance with the Functional & Technical requirements “interoperable and open solutions, standards and new business models”;
- D4.4: Document translating the use cases of the LH Cities into a technical architecture as a reference for each implementation of the CIP within LH Cities and Follower cities.

1.2. Scope, objectives and expected impact

Task 4.5 of Work Package 4 is described in the IRIS project management plan as “Implementation and integration of core CIP components”.

The deliverable describes all information about “Technical components / Configuration management / Deployment procedure” to implement a generic API, Data models, and support for the FIWARE platform components that are required for the development of integrated solutions.

The document describes the products, technologies, procedures necessary to install and support a CIP platform, and to ensure that the system components are compatible and comply to support for the FIWARE platform components.

It also aims to facilitate future developments of integrated solutions “services / applications” to handle a large volume of data and information coming from various domain in the scope defined in the requirements.

It provides information applicable to the different parties involved in the IRIS project in accordance with all the common decisions taken to build the current architecture.

Specific implementations of this architecture are beyond the scope of this document and must be specified and developed by each city.

This document completes the current documentations describing the functional and technical architectures.

1.3. Contributions of partners

The Implementation and Integration of core CIP components is a joint effort between the three LH Cities.

Each of the partners has contributed with a set of reflection, technical research and development activities, to provide and share its results by comparative tests (Advantages / disadvantages) with the different components implemented.

The FIWARE concept is built around a set of software components that integrate with each other based on standards for data models and APIs.

In the case of software components, different candidate solutions with the same objective have been studied.

Several comparative tests were carried out on "Open Source" tools to propose a basic option and alternative options.

1.4. Relation to other activities

1.4.1. Previous relation

This document is based on the work done upstream and the documents previously delivered. These documents provide all “functional and technical” expectations to build and implement the architecture of the CIP.

This document builds on the work done in D4.1, D4.2 and D4.4.

D4.1 provided a baseline on the current situation in each major city.

D4.2 described the functional reference architecture, based on an analysis of stakeholders, capabilities and components.

D4.4 gives a description of the technical components, data processes and APIs that will be used to build the CIP.

1.4.2. Joint relationship during the execution of activities

This document includes the work done by the different teams involved in the architecture project.

This distribution on different teams made it possible to test several tools for the same technical base and to propose in certain cases several options during the deployment.

To propose to each candidate city a bespoke architecture, the document presented in chapters

- The technical solution and the tools implemented by the city of Nice;
- Possible alternatives with complementary tools tested or replacement.

In the following chapters

- 2.2 - CIP Architecture – High-level presentation;
- 3 - CIP Architecture – Presentation.

1.4.3. Next relation

The platform built on open source technical components, is associated with a set of procedures which allow each city to implement an architecture compatible with the management and sharing of data in real time.

It is the basis for the future activities to guarantee the success of the tasks:

- T4.5 – integration of CIP in LH Cities.

For the following work packages:

- WP5: Utrecht LH City demonstration activities;
- WP6: Nice LH City demonstration activities;
- WP7: Gothenburg LH City demonstration activities.

In its exploitation phase, it will provide a vision of the architecture state, thus facilitating the evolution of the platform

1.5. Structure of the deliverables

3 types of deliverables were managed during the execution of the task T4.4:

Documentary components

- D4.5 – Implementation and Integration of Core CIP Components.

Technical Components - 1 link to a GIT including

- Tools, framework, open source software, from reference architecture guaranteeing the use and the exploitation of the platform.

Internal Documents of Nice city for monitoring the progress of the project

- MNCA – IRIS - WP4 T4.4 - Nice - WSRyyymmdd - Support.pptx;
- MNCA – IRIS - WP4 T4.4 - Nice - Action Plan & Deliverables & Planning.xlsm.

Only type 1 & 2 components are provided as part of the activities and deliverables of task 4.5.

This deliverable “Document D4.5” describes the different stakeholders, capabilities and technical components of the Platform, focusing on generic Principles that support the implementation of the core CIP components.

Chapter 2 - Methodology.

This section lists the different tools used for the implementation of the architecture and also presents the methodology used to progressively test the integration of the bricks implemented.

Chapter 3 - CIP Architecture – Detailed Presentation

This section presents in detail the technical components used in the implementation of the architecture foundation.

For each component, the reader can find information related to the reasons for this choice, difficulties encountered in its implementation, and links to different procedures that can provide advice and ease of installation.

Chapter 4 – Implementation procedure

This section describes how to implement the set of basic software to create a CIP platform

Chapter 5 – Conclusion

The document is not intended to define the standard reference architecture to be implemented, but rather to recommend the possible technical solutions and the software components to be used according to what has been tested according to the conditions of implementation by the MNCA and the other partner cities.

Chapter 6 – Reference

The chapter provides additional information in the understanding of the document.

2. Methodology

2.1. Global Approach

To respect an open source approach in FIWARE tooling context, the chosen method took place over 2 periods, each period including several steps.

- The steps of the first period were executed sequentially and only once.
- Each step of the second period was iterative by the sequential implementation of FIWARE tools, then alternate tools, depending on the scenario and the schedule defined at the end of the first period.

Period / Steps		Activities
Study and analysis of the requirements to determine the target tools	Select FIWARE tools regarding the target	• Mapping for each type of architecture components (DBMS / API / Security / ...) the available FIWARE tools and the prerequisites for their implementation; • How to implement data models and how to connect the different components to create an optimal platform
	Critical study of the tools and contribution	• Analyze individually the target requirements of each tool and in its entirety those of the platform; • Analysis of behaviors between tools through data flows; • Highlight the advantages and disadvantages of each tool and its integration in the global architecture.
	Alternate Solution	• Identify the possible alternate tools for each FIWARE tool, their add-value (scalability, maintainability, sustainability, performance, documentation, ...).
	Scenario and schedule	• Design and detail different scenarios and planning for implementation of FIWARE tools at first, then substitution of alternate tools.
Implementation of the solution	Implement FIWARE Tools	**FIWARE tool** • Integrate individually each FIWARE tool selected and validated its proper execution and its gateways with the other tools installed; • Perform a compliance analysis to ensure a minimum acceptable solution.
	Implement Alternate Tools	**Alternative tool** • Replace the FIWARE Tool by the alternate tools selected and validated their good executions and the gateways with the other tools installed; • Perform a POC to decide the final choice based on a comparative compliance analysis with the FIWARE tool.

Table 3 - Methodology - Global Approach

2.1.1. Implementation Scenario with FIWARE Tools

The scenario chosen for the implementation of FIWARE tools followed a strict rule. The implementation was done by iteration in order to test the correct functioning of the components and to perform non regression tests on the components already installed.

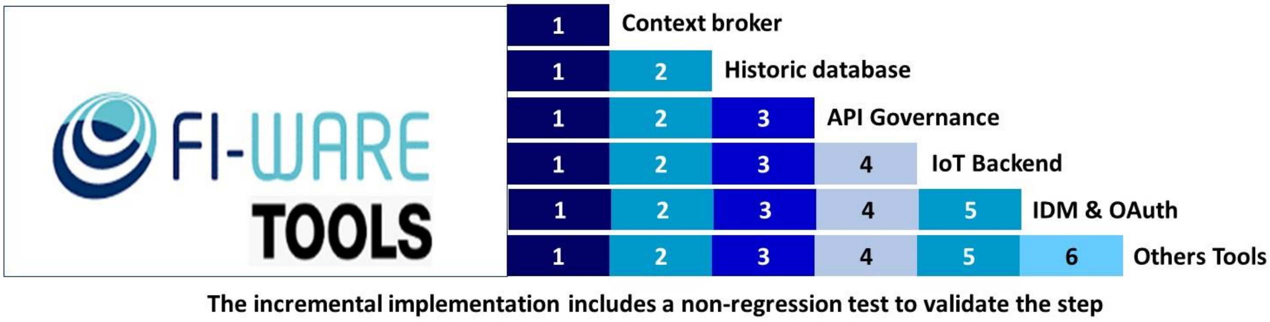


Figure 1 - Implementation approach with FIWARE Tools

No other implementation shall start if the previous implementation had not been validated in its globality.

This individual approach has allowed each step of tool integration to have a feedback “ROI” and capitalize in a plan of continuous progress for the next integration.

2.1.2. Implementation Scenario with combination alternative and FIWARE Tools

The scenario chosen for the implementation of the alternative tools followed the same rules.

According to the analysis, the choice of an alternative tool was 0 to 2 possible per FIWARE tool.

In order to study the best performance and scalability between different tools, several combinations of tool assemblies have been tested.

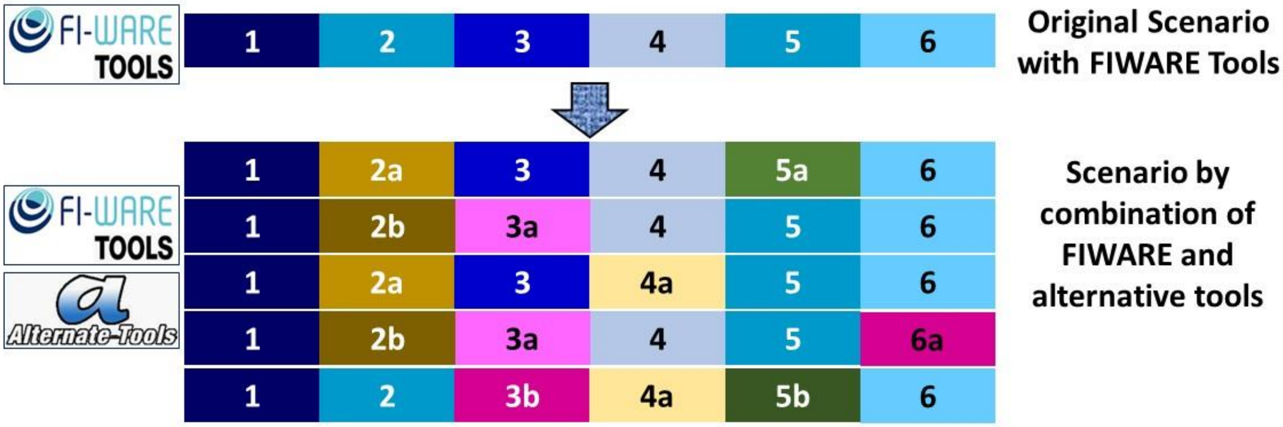


Figure 2 - Implementation approach with FIWARE and Alternate Tools

2.2. CIP Architecture – High-level presentation

2.2.1. Implementation principles

The architecture for the Platform includes the minimum components to use to run a smart city application in the scope of the requirements based on the FIWARE platform.

The Picture below shows the building blocks that can be used in an open urban platform below.

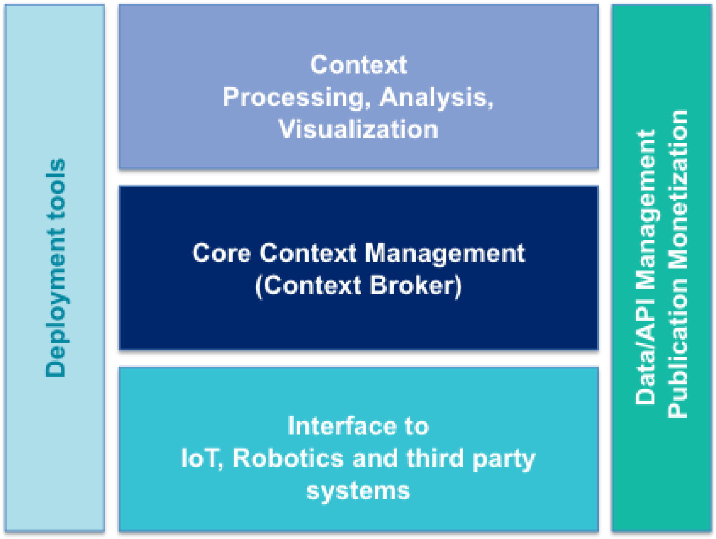


Figure 3 - FIWARE Platform: Technical Architecture (Source FIWARE[1])

As part of the IRIS project, the CIP architecture implemented as well as the technical components used are presented in the diagram below.

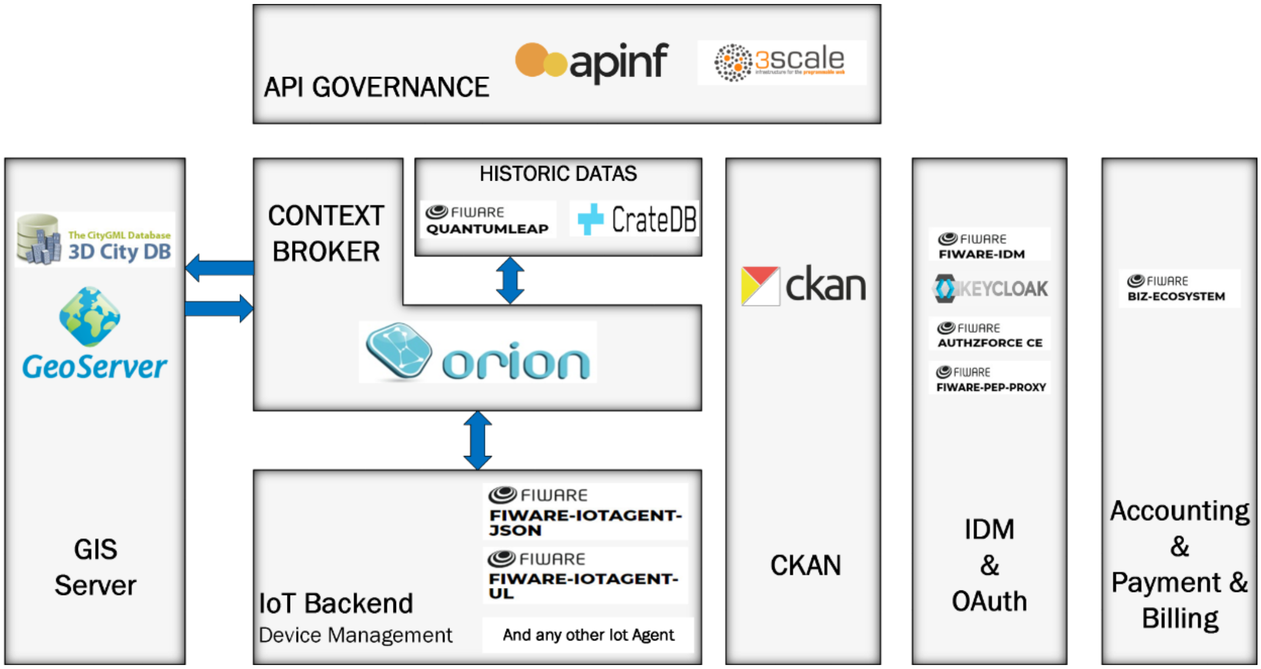


Figure 4 - CIP Architecture - Technical components implemented during the IRIS project

[1] https://www.fiware.org/developers/catalogue/

2.2.2. Synthetic presentation of the technical components

This section lists by theme the technical components implemented or tested in the CIP architecture during IRIS project. The next chapter describes with more detail, the architecture and the technical components.

The gathering of the technical components is formalized on the following topics:

- IDM & Authzforce (Security & Privacy)
- API Management
- Connector
- IoT Agents - Device Management - Exchange Format
- Data Base Storage and Publication
- Data Visualization / Analyzing / Monitoring
- Geographical tools
- Accounting / Payment / billing

2.2.2.1. IDM & Authzforce (Security & Privacy framework)

To reduce exposure to different threats and attack methods, while accepting a dynamic and open environment, FIWARE's internet security architecture focused on tools as:

Tools implemented for CIP Architecture	Alternative or complementary tools studied
PEP-PROXY	
AUTHZFORCE	
FIWARE KEYROCK - IDM	KEYCLOAK
	3scale

2.2.2.2. API Management

The APIs studied to publish, promote and supervise application programming interfaces are:

Tools implemented for CIP Architecture	Alternative or complementary tools studied
APInf	3scale
API Umbrella	

2.2.2.3. Connectors

This is connectors in charge of persisting certain sources of data in certain configured third-party storages, creating real time view or a historical view of such data:

Tools implemented for CIP Architecture	Alternative or complementary tools studied
Quantum Leap	FIWARE Cygnus / FIWARE STH Comet
Orion	

2.2.2.4. IoT Agents - Device Management - Exchange Format - IDAS

Solutions for collecting and transporting data as the communication channel protocol from IoT devices support exists with several standard file formats. Based on the standard data model defined by FIWARE[2], the accepted formats in the IoT backend are:

Tools implemented for CIP Architecture	Comments
IoT agent-Node Lib	Template to build custom agents
IoT agent-JSON	Bridge between JSON & NGSI interface of a context broker
IoT agent-UL	Bridge used to communicate devices using the Ultralight 2.0 protocol and NGSI Context Brokers

2.2.2.5. Data Base Storage and Publication

Core function of the architecture, it supports multiple sources and data formats to account for increased information delivered by connected objects in an evolving and constantly ecosystem changing. It allows to store data through an open portal to share, search and make accessible public and private data in different ways in the reference architecture.

Two options for storing the data are proposed:

- An option storing the last known value of an attribute, to always have the last known information,
- An option to save all the values of this attribute to follow its evolution to restore statistics in the form of graph or table.

The databases selected are listed below:

Tools implemented for CIP Architecture	Alternative or complementary tools studied
MongoDB	
FIWARE CRATEdb	MongoDB / HADOOP
Elasticsearch / Logstach	
Redis	

2.2.2.6. Data Visualization / Analyzing / Monitoring

The tools proposed are:

Tools implemented for CIP Architecture	Alternative or complementary tools studied
Grafana	
Kibana	

[2] https://www.fiware.org/developers/data-models/

2.2.2.7. GIS Server

The tools proposed are:

Tools implemented for CIP Architecture	Alternative or complementary tools studied
GeoServer	
3DCityDB	

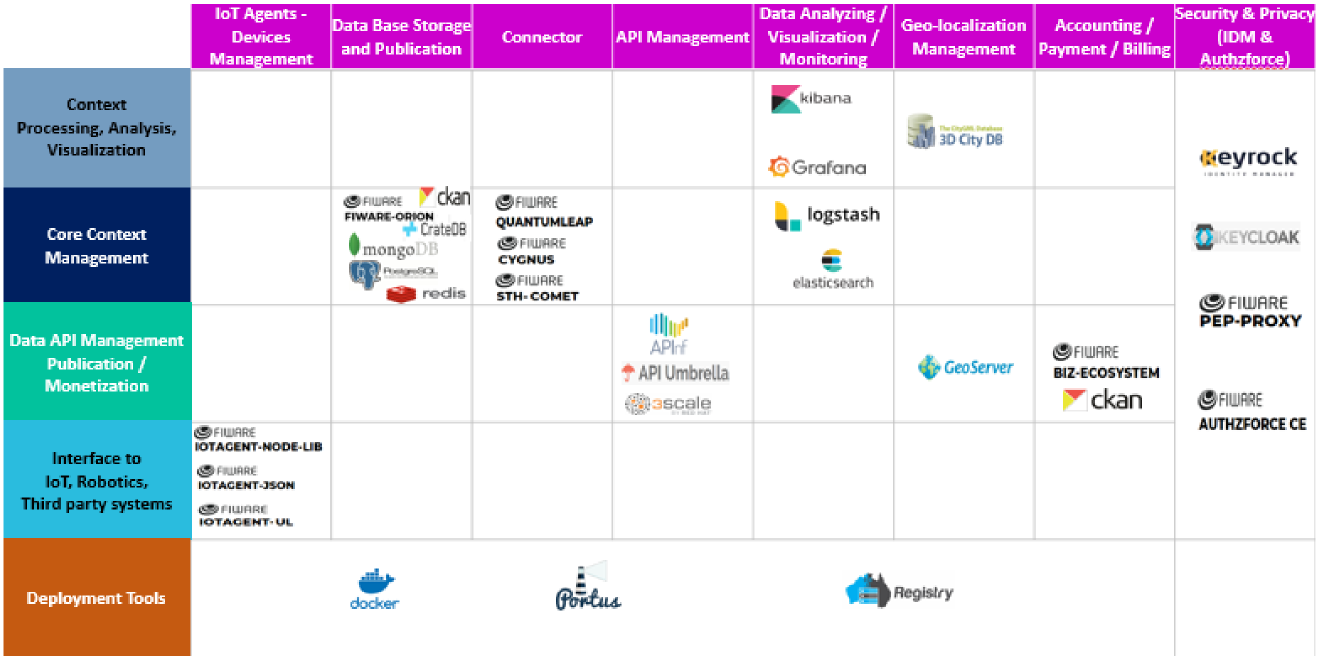
2.2.2.8. Accounting / Payment / billing

The tools proposed are:

Tools implemented for CIP Architecture	Alternative or complementary tools studied
FIWARE BIZ	3Scale

2.2.3. Mapping tools to implement at least for a CIP architecture / Fiware concepts

This section presents the positioning of the technical solutions implemented by Nice on the CIP architecture to meet the minimum requirements required to run a smart city application.



	IoT Agents - Devices Management	Data Base Storage and Publication	Connector	API Management	Data Analyzing / Visualization / Monitoring	Geo-localization Management	Accounting / Payment / Billing	Security & Privacy (IDM & Authzforce)
Context Processing, Analysis, Visualization					kibana Grafana	The CityGML Database 3D City DB		Keyrock IDENTITY MANAGER
Core Context Management		FIWARE FIWARE-ORION ckan CrateDB mongoDB PostgreSQL redis	FIWARE QUANTUMLEAP FIWARE CYGNUS FIWARE STH-COMET		logstash elasticsearch			KEYCLOAK
Data API Management Publication / Monetization				APInf API Umbrella 3scale BY RED HAT		GeoServer	FIWARE BIZ-ECOSYSTEM ckan	FIWARE PEP-PROXY FIWARE AUTHZFORCE CE
Interface to IoT, Robotics, Third party systems	FIWARE IOTAGENT-NODE-LIB FIWARE IOTAGENT-JSON FIWARE IOTAGENT-UL							
Deployment Tools		docker	Portus		Registry			

Figure 5 - CIP Architecture – Minimum tools to run a smart city application

2.2.4. IRIS architecture

The figure below shows the articulation of the different tools selected and implemented during the IRIS project.

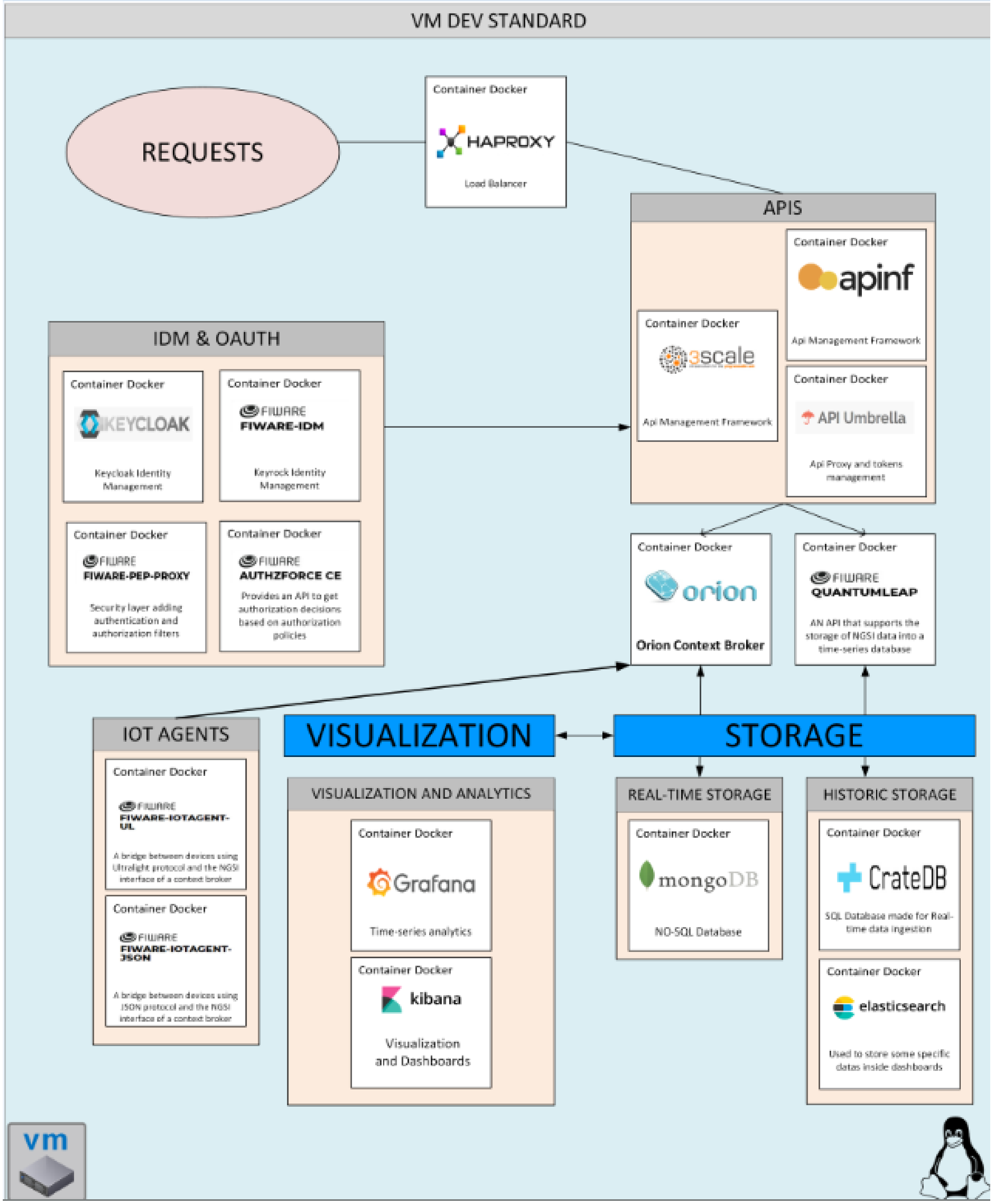


Figure 6 - CIP Architecture – Tools implemented in the IRIS project

3. CIP Architecture – Presentation

The CIP architecture has several components. For each of them a detailed description is given, it includes links to the publisher's documentation, to the associated FIWARE documentation and finally to the installation procedure as part of the IRIS project.

3.1. *Description of the studied tools*

For each tool studied and mentioned in the previous section, the chapter presents in detail the technical information needed to understand it. Links refer to official sites for up-to-date additional information

A common formalism of presentation has been adopted according to the model below. It contains for each tool:

- Its goal and the reason to choose it
- How to install the component in a CIP architecture with a link if used in the IRIS project

Standard installation	Link to editor tool if develop by an editor
Implementation procedure with FIWARE	Link to FIWARE documentation
Implementation procedure with IRIS project	Link to guide

- Software requirements (if known), Additional information (if necessary). Information presentations are formalized in the model below:

Tool Logo & Goals	
Reasons for choosing this component	
How to install the technical component	*Standard Installation* *Implementation procedure with FIWARE environment* • Admin Guide • User Guide *Implementation procedure if used within CIP Architecture project (IRIS)*
Software / Hardware Requirements	
Additionnal information	

Figure 7 - Template used to describe the studied tools

As proposed, the document presents also alternative tools tested during the CIP architecture project but not implemented. In this case, no installation procedure is described.

3.1.1. Orion – Context broker component

orion FIWARE **FIWARE-ORION**	Orion Context Broker allows to manage the entire lifecycle of context information including updates, queries, registrations and subscriptions. It is an NGSIv2 server implementation to manage context information and its availability. https://fiware-orion.readthedocs.io/en/master/
Reasons for choosing this component	This component is mandatory to the Fiware platform: this is the data exchange bus of the platform. The reference implementation of the Publish/Subscribe Context Broker GE.
How to install the technical component	*Standard Installation* https://fiware-orion/master/admin/install *Implementation procedure with FIWARE environment* • Admin Guide https://fiware-orion/master/admin • User & Programmers Guide https://fiware-orion/master/user *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	Orion context broker is developed in C++. The size of the Orion container image is 260 MB. *Host* • 2 CPU cores • 4 GB RAM *Operating system* • CentOS 7.4.1708 but it should work also in any later CentOS • RedHat 7.x version. *Database* • MongoDB 3.6 is required to run either in the same host where Orion Context Broker is to be installed or in a different host accessible through the network.

Additionnal information

Context Broker as a central component interact with many other components such as IOT Agents, Cosmos (Cygnus and Comet), CEP, Wirecloud

Orion doesn't provide "native" authentication nor any authorization mechanisms to enforce access control.

Authentication / Authorization are provided using the FIWARE PEP Proxy GE. / Keyrock IDM security component.

To provide high availability and scalability, the Orion component is deployed as a service through a Docker stack file, meaning that several container instances of Orion will run to provide HA, scalability.

The following schema shows the place of Orion context broker in the Fiware architecture:

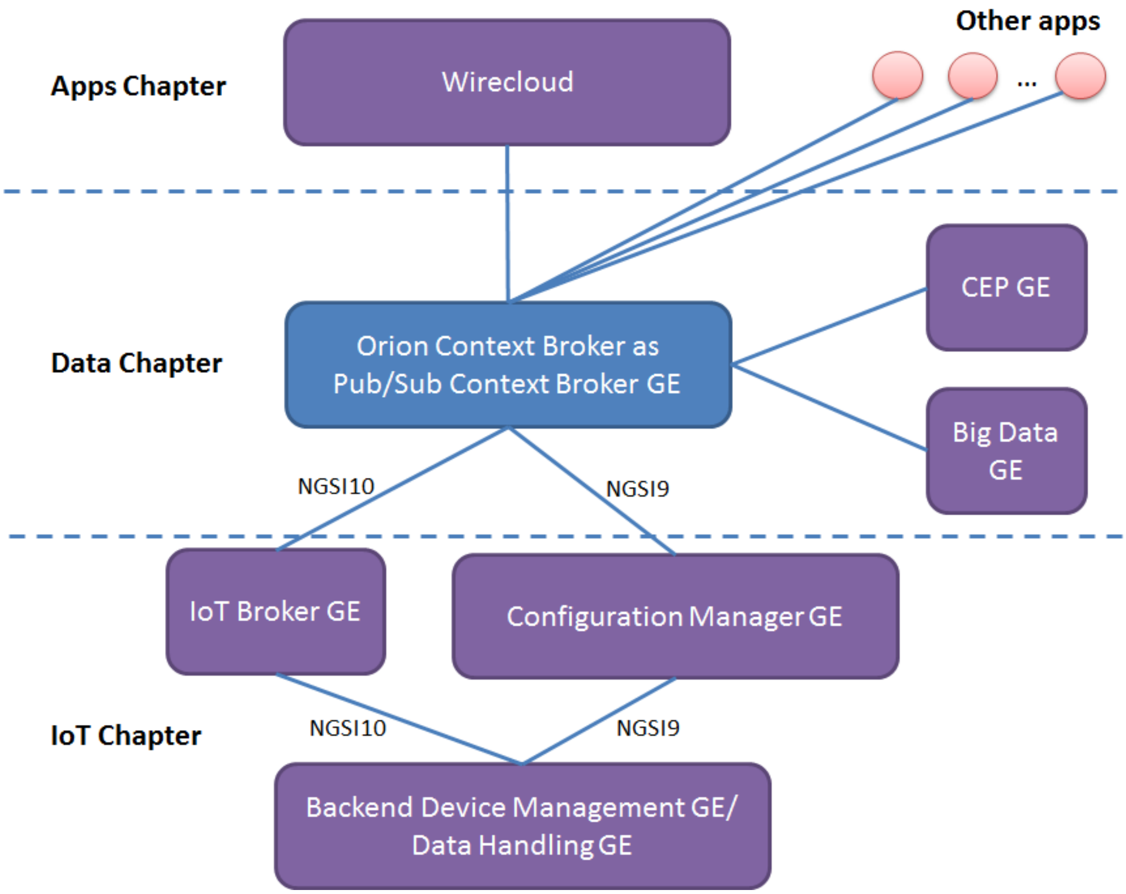


Figure 8 - Orion Context Broker relation with other components

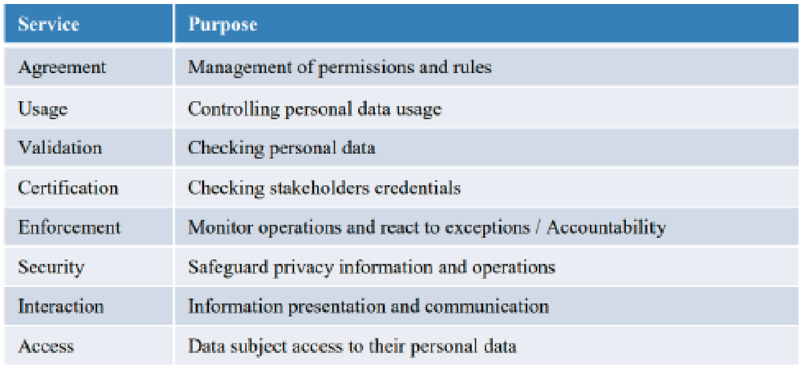
This component is central to the platform, because it's the dorsal of the Fiware architecture

It can become a bottleneck in case of high traffic, so a particular attention is needed on response time, through specific monitoring.

3.1.2. IDM & Authzforce - Security & Privacy

The security component is central to the platform, and so, used by most of other components from the platform to provide them authentication and authorization management, such as Orion Context Broker, CKAN, CEP, as well as Business ecosystem.

This section presents the tools that introduce identity management in FIWARE. The software solutions proposed meet with the requirements set within the specifications.



Service	Purpose
Agreement	Management of permissions and rules
Usage	Controlling personal data usage
Validation	Checking personal data
Certification	Checking stakeholders credentials
Enforcement	Monitor operations and react to exceptions / Accountability
Security	Safeguard privacy information and operations
Interaction	Information presentation and communication
Access	Data subject access to their personal data

Table 4 - List of security requirements (ref to D4.2 - Figure 22 PMRM services / Page 22 /)

This component is the Fiware GE responsible for Fiware platform global user identity & access rights management. It is composed of 3 elements:

- ***Identity Manager : KeyRock***

  The Keyrock module provides identity management, the integrated web UI is provided through Horizon front-end, which allows user accounts management.

- ***Access Control : AuthZForce***

  AuthZForce is the PDP (Policy Decision Point). It provides roles and access rights management based on XACML.

- ***GE module to add security to backend applications : PEP Proxy***

  The PEP Proxy components provides KeyRock security to other backends of the platform such as Orion context broker.

The 3 components together offer 3 levels of security:

- ***Level 1 Authentication***

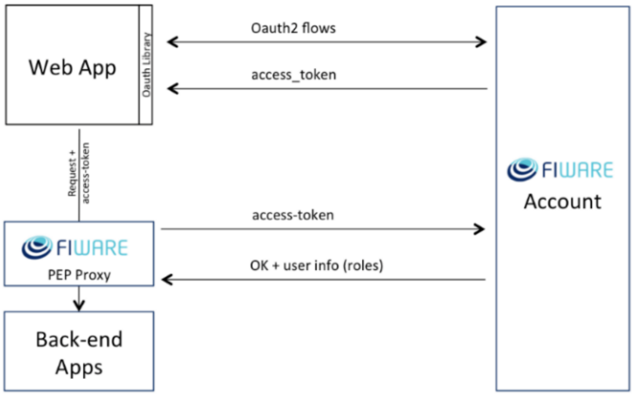


Figure 9 - Level 1: Authentication only (Fiware Account is Keyrock IDM)

- ***Level 2 Basic Authorization***

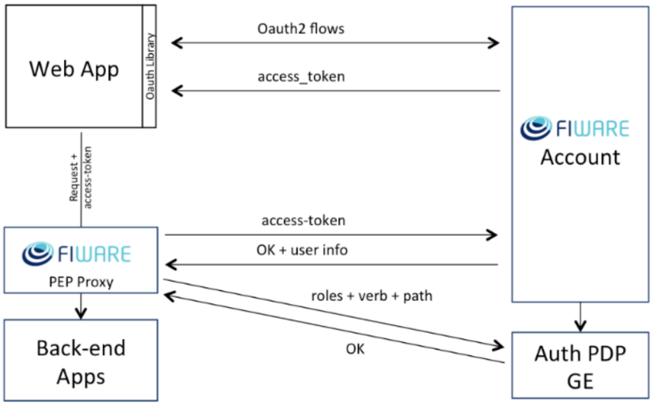


Figure 10 - Level 2: Authentication & Basic Authorization

- ***Level 3 Advanced Authorization***

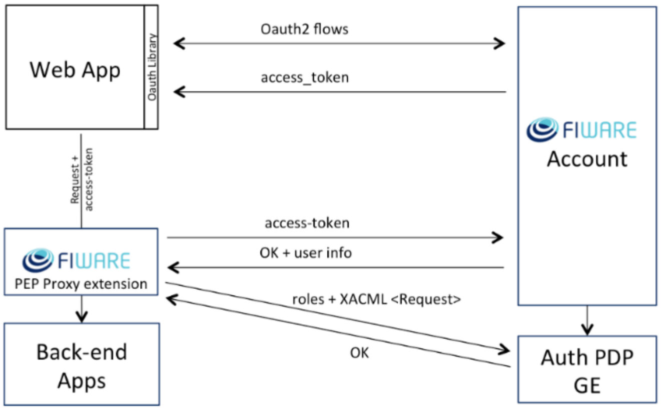


Figure 11 - Level 3: Authentication & Advanced Authorization (using XACML).

3.1.2.1. IDM - KEYROCK

Keyrock IDENTITY MANAGER FIWARE FIWARE-IDM	Keyrock is a FIWARE component responsible for Identity Management (IDM). The tool enables the authentication and authorization security for services and applications and can be associated with other security FIWARE components as PEP Proxy and Authzforce or other tools. https://fiware-idm/identity-manager-keyrock
Reasons for choosing this component	This component adds an identity management feature. The use of Keyrock (in conjunction with other security components such as Proxy PEP and Authzforce) allows authentication and authorization security to be added to the various CIP services.
How to install the technical component	*Standard Installation* N/A *Implementation procedure with FIWARE environment* • Admin Guide *https://fiware-idm/Keyrock/install* • User Guide

	https://fiware-idm/Keyrock/user_guide *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	This tool is based on a JavaScript environment and SQL databases. To execute the tool, several requirements are mandatory https://fiware-idm/software-requirements
Additionnal information Keyrock plays the role of PAP (Policy Administration Point) and PIP (Policy Information Point) in architecture. It sets rules for data usage.	

3.1.2.2. AuthZforce

FIWARE **AUTHZFORCE CE**	AuthzForce is the reference implementation of the authorization program for obtaining authorization decisions via an API and PEP proxy (Wilma) https://authzforce-ce-fiware/Introduction
Reasons for choosing this component	The use of Authzforce (in conjunction with other security components such as Proxy PEP and Keyrock) allows authentication and authorization security to be added to the various CIP services.
How to install the technical component	*Standard Installation* N/A *Implementation procedure with FIWARE environment* • Admin Guide https://authzforce-ce-fiware/Installation-AdministrationGuide • User Guide https://authzforce-ce-fiware/UserAndProgrammersGuide *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	• CPU frequency: 2.6 GHz min • CPU architecture: i686/x86_64 • RAM: 4GB min • Disk space: 10 GB min • File system: ext4 • Operating System: Ubuntu 16.04 LTS • Java environment
Additionnal information AuthzForce is the reference implementation of the Authorization PDP Generic Enabler (formerly called Access Control GE). Indeed, as prescribed by the GE specification, this implementation provides an API to obtain authorization decisions based on authorization rules and authorization requests from PEP.	

3.1.2.3. PEP-PROXY

FIWARE PEP-PROXY	The PEP Security Proxy is the first security step. It can be combined with other security components such as Keyrock, keycloak and Authzforce to apply application access control. https://fiware-pep-proxy.readthedocs.io/en/latest/
Reasons for choosing this component	The use of Proxy PEP (in conjunction with other security components such as Authzforce and Keyrock) allows authentication and authorization security to be added to the various CIP services.
How to install the technical component	*Standard Installation* N/A *Implementation procedure with FIWARE environment* • Admin Guide https://fiware-pep-proxy/admin_guide/ • User Guide https://fiware-pep-proxy/user_guide/ *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	No requirements

Additionnal information

The PEP proxy provides a layer of security to add authentication and authorization filters to GE FIWARE software and any primary service. This is the PEP (Police Enforcement Point) of the FIWARE chapter on security. Thus, with GE PDPs for identity management and permissions, it ensures the security of FIWARE servers.

3.1.2.4. IDM – KEYCLOAK

KEYCLOAK	Keycloak is an open source software solution for identity and access management (IDM) that makes it easy to implement application and service security by limiting specific code development https://www.keycloak.org/
Reasons for choosing this component	Keycloak offers services by thematic (authentication, registration, social authentication, user dashboard, ...) and supports the major technologies of the market.
How to install the technical component	*Standard Installation* *https://www.keycloak.org/downloads.html* *Implementation procedure with FIWARE environment* • Admin Guide *https://fiware-idm/build-install*

	• User Guide https://fiware-idm/user_guide/ *Implementation procedure if used within CIP Architecture project (IRIS)* • This component is an alternative to the IDM components in the CIP architecture. All information can be found on this web site: https://www.keycloak.org/server_installation/index.html
Software / Hardware Requirements	• Can run on any operating system that runs Java • Java 8 JDK • zip or gzip and tar • At least 512M of RAM • At least 1G of diskspace • A shared external database like PostgreSQL, MySQL, Oracle, etc. Keycloak requires an external shared database if you want to run in a cluster. Please see the database configuration section of this guide for more information. • Network multicast support on your machine if you want to run in a cluster

Additionnal information

Keycloack is based on a single sign-on (SSO) method that allows a user to access multiple application domains by performing only one authentication

If the identification of a user on a workstation is made through LDAP, Active Directory or own provider, the latter is automatically authenticated to Keycloak without relinquishing its login information.

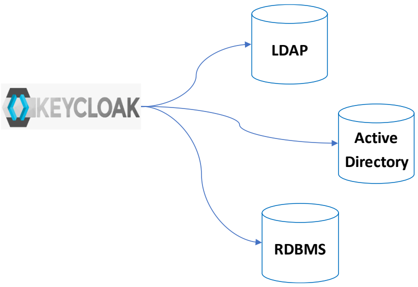


Figure 12 - Keycloak - Identification process – Directory Service

Connection from social networks, as well as identity providers are also possible.

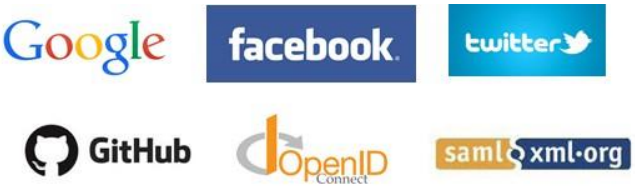


Figure 13 - Keycloak – Identification process - Social network

With a very simple integration process, the establishment of the links is carried out by the Keycloak administration console, by selecting the components to be added as an identification protocol.

No changes should be made to both the security system and the applications

Based on standard protocols, the software supports OpenID Connect, OAuth 2.0 and SAML

Figure 14 - Keycloak - Standard protocols

3.1.3. API Management

The major features are oriented towards APIs allowing the provision of services and tools of:

- Records Management
- Publication,
- Development Portal
- Reporting and analysis.

The FIWARE API management framework relies on the integration of several APIs components such as API-umbrella, APInf, 3Scale.

3.1.3.1. APInf

APInf	APInf Framework is an Open source API management platform with multi proxy and protocol support. It is a Smart City orchestrator to be used with other FIWARE enablers. https://www.apinf.io/
Reasons for choosing this component	Based on API Umbrella, the APInf framework offers a complete set of API management tools. It includes enhanced user interface features for API and consumer management. APInf integrates with FIWARE core technologies, such as Identity Management, NGSI v2 and Business API Ecosystem along with API Umbrella to offer a comprehensive toolset to let various API Owners to run business with their APIs. Public marketplace is one of the essential GUI tools to help API Owners to reach their customer with the showcase (https://fiware-academy./data-publication/apinf).
How to install the technical component	*Standard Installation* https://github.com/apinf/platform#installation *Implementation procedure with FIWARE environment* • Admin Guide https://apinf-fiware/admin-installation-docker • User Guide https://apinf-fiware/user-guide *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	No requirement

Additionnal information

APInf is fully open-source, it is developed in javascript with the [Meteor framework.](#)

3.1.3.2. API Umbrella

API Umbrella	API Umbrella is an open source API management platform for web services. https://apiumbrella.io/
Reasons for choosing this component	The FIWARE API structure integrates the Umbrella API as a basic proxy technology to access FIWARE APIs. APInf and FIWARE are joined to accelerate the integration of key API capabilities to exploit the benefits coming out of "APInf technologies". https://fiware.org/apinf-api-management-fiware-platform/
How to install the technical component	*Standard Installation* https://apiumbrella.io/install/ *Implementation procedure with FIWARE environment* • Admin Guide N/A • User Guide N/A *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	Minimum systeme recommandation is 1.5 GB memory on your computer or server. The API Umbrella runs on the following componant "Docker / Debian / Enterprise Linux / Umbuntu / from source code"

Additionnal information

API Umbrella is developed with Ruby on Rails, it uses MongoDB for data persistence, and Elasticsearch for analytics, the interface is a RESTful API

3.1.3.3. 3scale

3scale BY RED HAT	3scale is a API infrastructure for internal or external users https://www.3scale.net/
Reasons for choosing this component	3Scale is open source and now part of Red Hat. Red Hat® 3scale API Management makes it easy to manage your APIs. Share, secure,

	distribute, control, and monetize your APIs on an infrastructure platform built for performance, customer control, and future growth.
How to install the technical component	*Standard Installation* https://3scale_api_management/2.5/single/installing_3scale *Implementation procedure with FIWARE environment* - Admin Guide N/A - User Guide N/A *Implementation procedure if used within CIP Architecture project (IRIS)* - This component is an alternative to the API Management in the CIP architecture.
Software / Hardware Requirements	No requirement
Additionnal information N / A	

3.1.4. Connectors

3.1.4.1. FIWARE - Quantum Leap

FIWARE QUANTUMLEAP	QuantumLeap is an API that supports the storage of FIWARE NGSIv2 data into **CRATE** Databases, as time series database. https://quantumleap.io
Reasons for choosing this component	QuantumLeap is an API that supports the storage of NGSI FIWARE NGSIv2 data into a time-series database, known as ngsi-tsdb.
How to install the technical component	*Standard Installation* N/A *Implementation procedure with FIWARE environment* - Admin Guide https://smartsdk.github.io/ngsi-timeseries-api/admin/ - User Guide https://smartsdk.github.io/ngsi-timeseries-api/user/ *Implementation procedure if used within CIP Architecture project (IRIS)* Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	No Requirement
Additionnal information N / A	

3.1.4.2. FIWARE - Cygnus

FIWARE CYGNUS	Cygnus is a connector in charge of persisting data in a Orion context. It creates a historical view of these data, in only stored the last attribute values of an entity https://fiware-cygnus.io
Reasons for choosing this component	This an alternative component to the proposed solution.
How to install the technical component	*Standard Installation* N/A *Implementation procedure with FIWARE environment* • Admin Guide https://fiware-cygnus/installation_administration_guide • User Guide https://fiware-cygnus/user_programmer_guide *Implementation procedure if used within CIP Architecture project (IRIS)* • This agent was not used for CIP architecture project.
Software / Hardware Requirements	**N/A**
Additionnal information N / A	

3.1.4.3. FIWARE-STH Comet

FIWARE STH- COMET	It is responsible of the management of chronological data and their historical aggregation according to their evolution in an instance of ORION's "Context Broker. https://fiware-sth-comet.Io
Reasons for choosing this component	This an alternative component to the proposed solution.
How to install the technical component	*Standard Installation* N/A *Implementation procedure with FIWARE environment* • Admin Guide https://fiware-sth-comet/Admin_guide https://smartsdk.github.io/ngsi-timeseries-api/admin/ • User Guide *https://fiware-sth-comet/contribution-guidelines* *Implementation procedure if used within CIP Architecture project (IRIS)*

	- This agent was not used for CIP architecture project.
Software / Hardware Requirements	No requirement
Additionnal information N/A	

3.1.5. IoT Agents - Device Management - Exchange Format

Several solutions exist to collect data from IoT Devices. Each device follows its own protocol when sending its collected data to a platform. This protocol differs from the manufacturers and types of devices within the same manufacturer.A protocol is based on standard file formats as communication channel. The exchange of data can be done through a considerable number of formats. Each format has its strengths and weaknesses depending on the context. The final choice is the format that offers the best balance between cost and relevance in terms of the use of the exchange protocols on the solutions implemented. An IoT agent, a component that groups devices with the same native protocol, uses its own proprietary protocols and disparate transport mechanisms. Also, each IoT agent has a standardized NGSI interface for its interactions with the Context Broker.

3.1.5.1. IoT Agent Node Lib

FIWARE **IOTAGENT-NODE-LIB**	This IoT agent give the method to develop bridge between HTTP/MQTT messaging with a specific format and NGSI and to use the library. https://iotagent-node-lib
Reasons for choosing this component	it is possible to use the Ultralight 2.0 protocol and adapt it to the desired format.
How to install the technical component	*Standard Installation* N/A. *Implementation procedure with FIWARE environment* - Admin Guide https://iotagent-node-lib/installationguide - User Guide https://iotagent-node-lib/usermanual *Implementation procedure if used within CIP Architecture project (IRIS)* - This agent was not used for CIP architecture project. This is a template to create IoT-Agent
Software / Hardware Requirements	Node.js v8 (at least) installed and working on your machine.
Additionnal information N / A	

3.1.5.2. JSON – IoT Agent JSON

JSON FIWARE IOTAGENT-JSON	The IoT agent is a bridge between HTTP/MQTT messaging (with a JSON format) and NGSI. https://fiware-iotagent-json JSON (JavaScript Object Notation), based on a subset of JavaScript, allows the exchange of data in a text format without language. https://www.json.org/
Reasons for choosing this component	The syntax of JSON is very similar to the syntax of JavaScript literal objects. It can be used independently of Scripting language The current programming languages offers JSON syntax analysis and generation functions.
How to install the technical component	*Standard Installation* JSON format is just a support used by executable applications to facilitate the exchange of data. *Implementation procedure with FIWARE environment* • Admin Guide https://fiware-iotagent-json/installationguide • User Guide https://fiware-iotagent-json/UserGuide *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	**No requirement**
Additionnal information N / A	

3.1.5.3. UL – IoT Agent UL

FIWARE IOTAGENT-UL	The IoT agent is a bridge between HTTP/MQTT messaging and NGSI. It uses a lightweight text protocol (Ultralight 2.0) for devices whose bandwidth and memory have limited resources https://fiware-iotagent-ul
Reasons for choosing this component	Simple and low bandwith text protocol that allows IoT devices to transmit data or receive commands.
How to install the technical component	*Standard Installation* N/A *Implementation procedure with FIWARE environment* • Admin Guide

	https://fiware-iotagent-ul/installationguide • User Guide https://fiware-iotagent-ul/userguide *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	No Requirement
Additionnal information Ultralight 2.0 is a lightweight text protocol that reduces the amount of data transferred. The body of the message for information update requests is made up of a list of key-value pairs separated by the symbol " | "	

3.1.6. Database – Storage and Publication

3.1.6.1. CrateDB

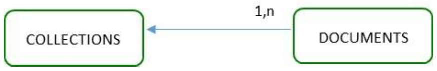
	CrateDB is a distributed SQL database based on a NoSQL database. This build makes it easy to combine the ease of use of SQL queries with the scalability and flexibility of NoSQL data. https://crate.io/
Reasons for choosing this component	CrateDB supports both relational data and nested JSON-documents and allows to • Use of SQL to process any type of data, structured or not. • Execute SQL queries including joins, aggregations and nested JSON attributes. • Store BLOBs like pictures, videos, or large unstructured files.
How to install the technical component	*Standard Installation* • Admin Guide / User Guide https://crate.io/download/ https://crate.io/docs/crate/guide https://crate.io/docs/crate/reference *Implementation procedure with FIWARE environment* • Admin Guide / User Guide https://fiware-tutorials/persisting-querying-time-series-data-cratedb *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	No requirement
Additionnal information N / A	

3.1.6.2. MongoDB

mongoDB	MongoDb is a NoSQL database management system that manipulates structured objects without a specific schema, natively scalable by multiplying the nodes/containers running the software and replicating the data upon the nodes. www.mongodb.com
Reasons for choosing this component	MongoDB is the selected NoSQL database for several Fiware components " Orion Context Broker, IOT agents, Comet STH, Perseo". This structuring allows to evolve in real times the physical representation of the objects without obligation to reconfigure the database. It runs on the following Operating Systems "Linux, MacOS, Microsoft Windows, Ubuntu, Amazon".
How to install the technical component	*Standard Installation* https://mongodb.com/community *Implementation procedure with FIWARE environment* - Admin Guide / User Guide https://fiware-draco/ngsi_mongo_sink *Implementation procedure if used within CIP Architecture project (IRIS)* - To achieve high availability several replicas will be deployed (each replica is a container), see MongoDB documentation Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	This tool is based on a JavaScript environment and SQL databases. To execute the tool, several requirements are mandatory - MongoDB version 3.6 The standard MongoDB incoming port is 27017/TCP. Container RAM memory sizing: 4 GB Storage sizing: - Docker image size is 361 MB - Data have to be stored on a specific Docker volume on the container host, each container running MongoDB should have its own volume on the host node - The size available for data storage, should be at least 300 GB for each storage volume

Additionnal information

With MongoDB, we talk about "Documents" that are stored in "Collections". A "Collection" contains a number of "Documents".



A "Document" may contain a different number of "Fields", which may vary for the same "Collection".

Compared to a relational Database:

- Collections = Tables,
- Documents = Records.

The number of Fields in a Record may vary from one Record to another while they are stored in the same Table

3.1.6.3. Elasticsearch

elasticsearch	Elasticsearch is a database server that facilitates the indexing and searching of structured and non-structured data, geographic data, documents, .. https://www.elastic.co/fr/products/elasticsearch
Reasons for choosing this component	Logstash is one of the three components of the ELK suite: Elasticsearch, Logstash, Kibana. Elasticsearch is the collection database.
How to install the technical component	*Standard Installation* • Admin Guide https://www.elastic.co/elasticsearch/install • User Guide *https://www.elastic.co/elasticsearch/userGuide* *Implementation procedure with FIWARE environment* • Admin Guide N/A • User Guide N/A *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	No requirement

Additionnal information

N / A

3.1.6.4. Logstach

logstash	Logstash is an open source solution that provides a channel for processing server-side data. It simultaneously integrates a multitude of data sources, transforms them and sends them to a storage system. https://www.elastic.co/fr/products/logstash
Reasons for choosing this component	Logstash is one of the three components of the ELK suite: Elasticsearch, Logstash, Kibana. Logstash is the collection and analysis tool.
How to install the technical component	*Standard Installation* • Admin Guide *https://elastic.co/logstash/installing* • User Guide *https://elastic.co/logstash/UserGuide*

	Implementation procedure with FIWARE environment • Admin Guide / User Guide N/A *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	No requirement

Additionnal information

N / A

3.1.6.5. Postgres

	PostgreSQL is an open source object-relational database system that has a proven, reliable and secure architecture that ensures data integrity. it is also composed of additionals features that allow extensions such as PostGIS "geospatial database".. https://www.postgresql.org/
Reasons for choosing this component	PostgreSQL is a free Database Management System (SBGD) available under BSD license. This multi-platform system is widely known and renowned around the world, especially for its stable behavior and for being very respectful of ANSI SQL standards.
How to install the technical component	*Standard Installation* • Install procedure https://www.postgresql.org/download/ • Documentation https://www.postgresql.org/docs/11/index.html *Implementation procedure with FIWARE environment* • Admin Guide / User https://fiware-draco./ngsi_postgresql *Implementation procedure if used within CIP Architecture project (IRIS)* • This agent was not used for CIP architecture project.
Software / Hardware Requirements	PostgreSQL runs on all major operating systems.

Additionnal information

N / A

3.1.6.6. Redis

	Redis is an open source solution which allows the exchange of data. It is a data structures server, supporting different kinds of values, not limited to a simple string. https://redis.io/ https://docs.redislabs.com
Reasons for choosing this component	The city of Nice has implemented this software to take into account in the same tool several types of formats retrieved from different devices. This choice is made to the detriment of the Fiware Node Red -Lib which would have required the development of several IoT agents.
How to install the technical component	*Standard Installation* https://redis.io/download https://installing-upgrading *Implementation procedure with FIWARE environment* N/A *Implementation procedure if used within CIP Architecture project (IRIS)* N/A
Software / Hardware Requirements	Hardware requirements are different regarding DEV / PROD environments. Minimum set up for minimal development and functional testing are : • RAM At least 4GB Recommanded 8 GB • Storage At least 10 GB Recommanded 20 GB For production requirements https://designing-production/hardware-requirements/

Additionnal information

Redis data types

• Binary-safes strings	String with length attached in the string itself
• Lists	Collection of string element
• Sets	Collection of unique and unsorted string elements
• Sorted Sets	Similar to Sets but where every string element is associated to a floating number value
• Hashes	Maps composed of fields associated with values
• Bit arrays	Array data structure that compactly stores bits
• HyperLogLogs	Probabilistic data structure
• Streams	Append only collections of map-like entries that provide an abstract log data type

3.1.6.7. CKAN

	CKAN is a powerful data management system that makes data accessible – by providing tools to streamline publishing, sharing, finding and using data. https://ckan.org/
Reasons for choosing this component	The CKAN GE provides the following features: • Complete catalog system with easy to use web interface and a powerful API • Strong integration with third-party CMS's like Drupal and WordPress • Data visualization and analytics • Workflow support manage their own data publishing • Fine-grained access control • Integrated data storage and full data API • Federated structure : easily set up new instances with common search
How to install the technical component	*Standard Installation* https://ckan.org/download-and-install/ *Implementation procedure with FIWARE environment* • Admin Guide https://fiware-ckan-extensions/installation-administration-guide • User Guide https://fiware-ckan-extensions/user-programmer-guide *Implementation procedure if used within CIP Architecture project (IRIS)* N / A
Software / Hardware Requirements	CKAN runs on a server of normal capacity. The minimum requirements for each machine are: • 2 CPU cores • 4 GB of RAM • 60 GB of disk space

Additionnal information

It can manage large volumes of data and publishing them to provide search functionality to a diverse user population.

Publishing and the search function make it easy to crawl the data to extract the desired content.

Other tools allow to exploit these data for a cartographic representation, dashboard, and tables.

3.1.7. Data Visualization / Analyzing / Monitoring

3.1.7.1. Kibana

	Kibana is a complementary solution to the Elasticsearch search engine. It allows to visualize, filter the data in the form of graph, curve, ... https://www.elastic.co/fr/products/kibana
Reasons for choosing this component	Logstash is one of the three components of the ELK suite: Elasticsearch, Logstash, Kibana. Kibana is the visualization tool.
How to install the technical component	*Standard Installation* • Admin Guide https://elastic.co/kibana/install • User Guide *https://elastic.co/kibana/userGuide* *Implementation procedure with FIWARE environment* • Admin Guide *N/A* • User Guide *N/A* *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	No requirement
Additionnal information N / A	

3.1.7.2. Grafana

	Grafana allows you to query, visualize, alert on and **understand** your metrics no matter where they are stored. Create, explore, and share dashboards with your team and foster a **data driven culture**. https://grafana.com/
Reasons for choosing this component	Grafana is a tool that allow to quickly create dashboards to visualize real-time data from the CIP.
How to install the technical component	*Standard Installation* *https://grafana.com/docs/* *Implementation procedure with FIWARE environment* • Admin Guide N/A • User Guide

	N/A *Implementation procedure if used within CIP Architecture project (IRIS)* • This agent was not used for CIP architecture project.
Software / Hardware Requirements	No requirement
Additionnal information N/A	

3.1.8. Geographical tools

3.1.8.1. 3D City DB

The CityGML Database 3D City DB	3D City Database is a free 3D geo solution. It allows to store, represent, and manage models of virtual cities in 3D in connection with other spatial relational database systems. https://www.3dcitydb.org/3dcitydb/
Reasons for choosing this component	This component allows to store 3D city model and a rich interaction with 3D visualization models, e.g. highlighting of 3D objects on mouse-over and mouse click as well as hiding of selected 3D objects Dynamic loading and unloading of large 3D visualization models in the form of tiled KML/glTF datasets exported from the 3DCityDB using the Importer/Exporter. Adding and removing an arbitrary number of data layers like 3D visualization model (KML/glTF), 3DTiles datasets, WMS imagery layer, and Cesium digital terrain model
How to install the technical component	*Standard Installation* https://github.com/3dcitydb/3dcitydb https://github.com/3dcitydb/tutorials *Implementation procedure with FIWARE environment* • N/A *Implementation procedure if used within CIP Architecture project (IRIS)* • N/A *Implementation procedure if used within CIP Architecture project (IRIS)* • This agent was not used for current CIP architecture project.
Software / Hardware Requirements	Software is Open Source and released under the terms of the **Apache License, Version 2.0** • Oracle DBMS >= 10g R2 with Spatial or Locator option • PostgreSQL DBMS >= 9.3 with PostGIS extension >= 2.0

Additionnal information

- Spatial relational database schema for semantic 3D city models
- All thematic modules from CityGML 2.0 included
- Five different Levels of Detail (LODs)

- Appearance data (e.g. textures, colors) in addition to flexible 3D geometries
- Supports generic and prototypical 3D objects
- Complex digital terrain models (DTMs)
- Web Feature Service (WFS) interface
- Works with Oracle Spatial 10g, 11g, and 12c (Spatial and Locator), or PostGIS 2.0 or higher
- Docker containers available for quick setup

3.1.8.2. GeoServer

	Geo Server is an opensource server to share geospatial data from any major spacial datasource using open standard. https://geoserver.org/
Reasons for choosing this component	GeoServer is a Java-based software server that allows users to view and edit geospatial data. Using open standards set forth by the Open Geospatial Consortium (OGC), GeoServer allows for great flexibility in map creation and data sharing. It can be used in conjunction with 3D City DB in 3D city scenario.
How to install the technical component	*Standard Installation* • Admin Guide https://geoserver.org/installation *Implementation procedure with FIWARE environment* • N/A *Implementation procedure if used within CIP Architecture project (IRIS)* • This agent was not used for current CIP architecture project.
Software / Hardware Requirements	No requirement

Additionnal information

N / A

3.1.9. Accounting / Payment / billing

3.1.9.1. BIZ ecosystem

FIWARE BIZ-ECOSYSTEM	This component is built from a FIWARE Business framework and a set of APIs provided by TMForum. It facilitates monetization throughout the cycle, from initial offering to billing, settlement and information sharing https://business-api-ecosystem.io
Reasons for choosing this component	This is the Market Place component of the smart city platform, that allows to monetize applications, APIs and also data that are exposed through the smart city platform open data component.
How to install the technical component	*Standard Installation* N/A *Implementation procedure with FIWARE environment* • Admin Guide https://business-api-ecosystem/installation • User Guide https://business-api-ecosystem/user-guide *Implementation procedure if used within CIP Architecture project (IRIS)* • Refer to : &3.3 - implementation procedure
Software / Hardware Requirements	**TM Forum APIs and Revenue Sharing Service (RSS) requirements** • Java 8 • Glassfish 4.1 • MySQL 5.5 => to replace with PostgreSQL/Citus (or in extreme case use Galera Cluster and MariaDB if hard to change from MySQL to PostgreSQL) **Charging Backend requirements** • Python 2.7 • MongoDB • wkhtmltopdf **Logic Proxy requirements** • NodeJS 4.5.0+ (Including NPM)

Additionnal information

The Business API Ecosystem is not a single software repository, but it is composed of different applications which work coordinately to provide a complete solution .

Business Ecosystem Charging Backend:

- Process the different pricing models, the accounting information, and the revenue sharing reports.
- Calculate amounts to be charged, charge customers, and pay sellers, using this above information.

Business Ecosystem Revenue Sharing Service:

Distributes revenues originated by the usage of a given service among the involved stakeholders.

It distributes part of the revenue generated by a service between the Business API Ecosystem instance provider and the Service Provider(s) responsible for the service, a service here refers to both final applications and backend application services.

Business Ecosystem Logic Proxy:

Is the endpoint for accessing the Business API Ecosystem:

- it orchestrates the APIs validating user requests, including authentication, authorization, and the content of the request from a business logic point of view.
- it serves a web portal that can be used to interact with the system.

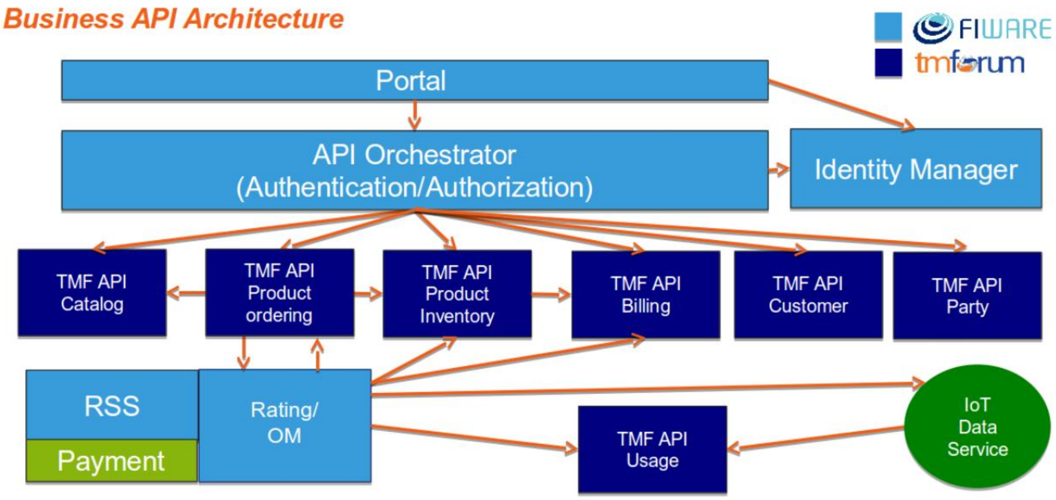


Figure 15 - Fiware Business Ecosystem Architecture

4. Implementation procedure

4.1. Introduction

This chapter details the procedure for implementing the set of basic software to create a CIP platform.

All the sources needed to install the platform are hosted on a GIT repository at the following address: https://github.com/orgs/IRIS-Smart-Cities/teams/cip-platform

In order to download the sources of the GIT repository, you must first install git on your machine by running the following command:

```
apt-get update && apt-get install git
```

You also need to install docker Docker engine, Docker allow us to deploy applications quickly, ready to use and without boring configurations to do.

Once git installed you have to launch the next command to download the sources:

```
git clone https://github.com/IRIS-Smart-Cities/cip.git .
```

Before launching the installation script, a configuration file allows you to set the environment.

4.2. Launching the installation procedure

To start the installation procedure, it is necessary to execute the following script “./cip/setup.sh”

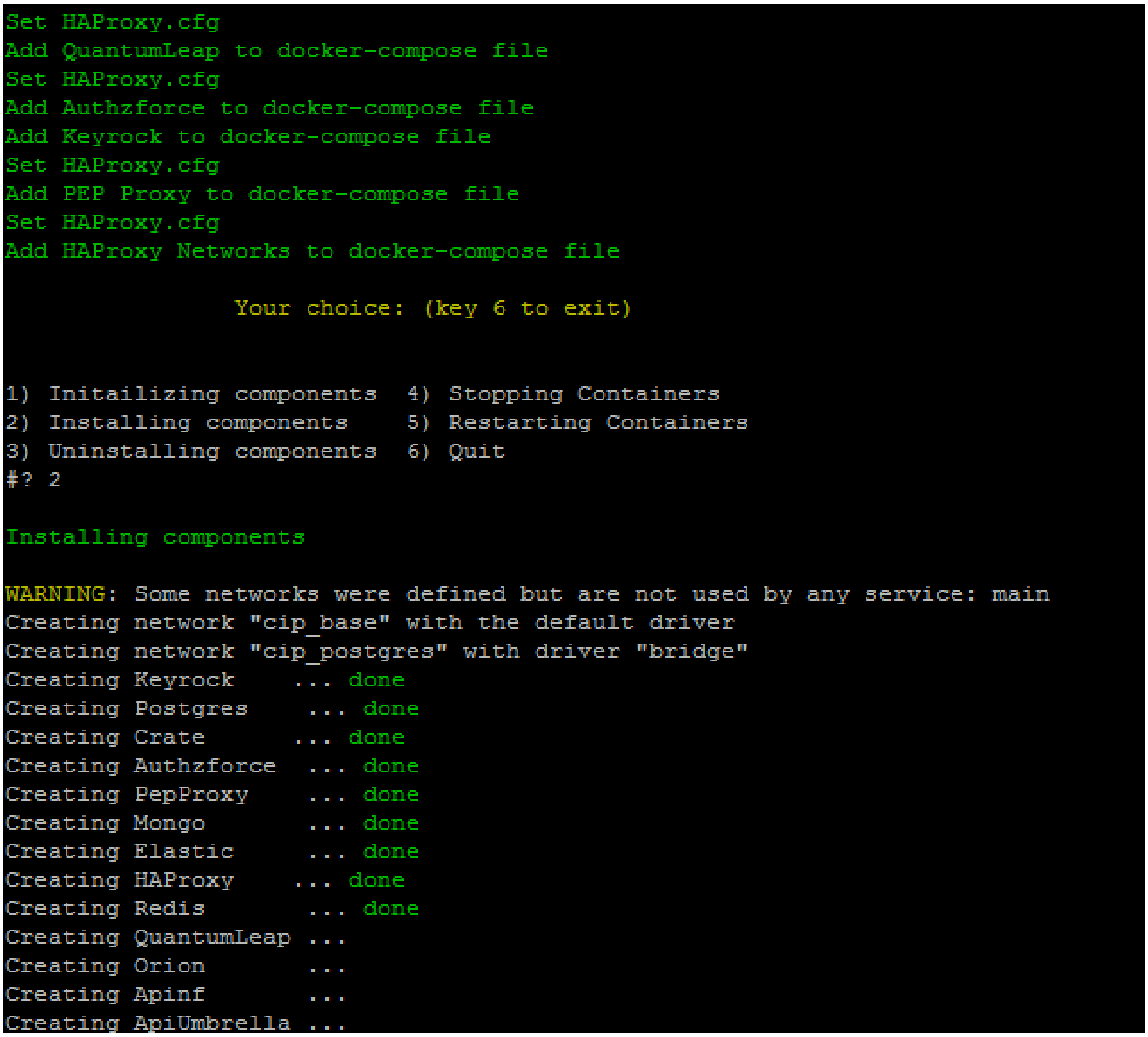
At the start the script display a menu:

1. Initialize the CIP
2. Installation of CIP components
3. Uninstalling CIP components
4. Stopping CIP containers
5. Restarting CIP containers

Option 1 "Initialize the CIP". The steps taken are;

- Certificate Generation
- Docker environment initialization

The figure below shows the beginning of the installation procedure:



```
Set HAProxy.cfg
Add QuantumLeap to docker-compose file
Set HAProxy.cfg
Add Authzforce to docker-compose file
Add Keyrock to docker-compose file
Set HAProxy.cfg
Add PEP Proxy to docker-compose file
Set HAProxy.cfg
Add HAProxy Networks to docker-compose file

                Your choice: (key 6 to exit)


1) Initailizing components  4) Stopping Containers
2) Installing components    5) Restarting Containers
3) Uninstalling components  6) Quit
#? 2

Installing components

WARNING: Some networks were defined but are not used by any service: main
Creating network "cip_base" with the default driver
Creating network "cip_postgres" with driver "bridge"
Creating Keyrock    ... done
Creating Postgres    ... done
Creating Crate      ... done
Creating Authzforce  ... done
Creating PepProxy    ... done
Creating Mongo       ... done
Creating Elastic     ... done
Creating HAProxy    ... done
Creating Redis       ... done
Creating QuantumLeap ...
Creating Orion       ...
Creating Apinf       ...
Creating ApiUmbrella ...
```

Figure 16 - Implementation procedure : Main menu

Option 2 "Installation of CIP components". The script will deploy all the CIP component on that Virtual Machine.

Those components are:

• Database:	MongoDB, PostgreSQL, Elasticsearch, CrateDB, Redis
• Security:	IDM (Authzforce, PEP-Proxy, Keyrock)
• Fiware Components:	Orion, QuantumLeap, IDAS (IotAgents)
• API Management:	APInf, API-Umbrella
• Accounting / Payment	Biz Ecosystem
• Catalog	Ckan
• GIS	Geoserver / 3D City DB
• Other tools:	HAProxy

Option 3 “Uninstalling CIP components” uninstall all components previously installed.

Option 4 “Stopping CIP Containers” stop the CIP containers.

Option 5 “Restarting CIP Containers” restart the CIP containers.

The useful documentation for supporting and using the platform is available at the following address:

https://github.com/IRIS-Smart-Cities/cip/blob/master/README.md

4.3. Test case

Test sets are available to check that the CIP is working properly.

The useful documentation for testing your installation is available at the following address:

https://github.com/IRIS-Smart-Cities/cip/blob/master/README.md

5. Conclusions

The CIP is an open and sustainable urban platform that implements essential features to meet the new challenges of connected cities. For this it must be an open, reusable and reliable platform allowing:

1. Data sharing;
2. Accelerating innovation, standardizing and implementing smart applications;
3. Cost-cutting by the usage of open components.

As a result, the CIP will meet the following requirements:

1. All components of the CIP are based on open source software
2. All components of the CIP are compliant with OASC-commitments
3. All three LH cities use the same common CIP-components
4. Each user of the platform has equal chances to connect to the CIP and use data-sources

This document describes all the components implemented in the CIP and facilitates the deployment by IT teams of the various LH and follower cities of the common platform. It enables the implementation and integration of core CIP components.

This common platform will be the starting point for the developments of the various demonstrators.

6. Reference

6.1. *List of hypertext links used in the document*

All links used in the documents are the property of their respective owners. The use of this information does not imply an authorization for the commercial use of unlicensed products by persons wishing to implement this solution.

FIWARE LINKS

General

FIWARE Catalogue

ORION

Orion website

Orion Installation guide

Orion Admin guide

Orion User guide

KEYROCK

Keyrock website

Keyrock Installation guide

Keyrock User guide

Keyrock Requirements

AuthZforce

AuthZforce website

AuthZforce Installation guide

AuthZforce User guide

PEP-PROXY

PEP-Proxy website

PEP-Proxy Admin guide

PEP-Proxy User guide

IDM

IDM Installation guide

IDM User guide

APInf

APInf Installation guide

APInf User guide

API UMBRELLA

API Umbrella website

QUANTUMLEAP

QuantumLeap Admin guide

QuantumLeap User guide

CYGNUS

Cygnus website

Cygnus Admin guide

Cygnus User guide

COMET

Comet website

Comet Admin guide

Comet Guideline

IoT Agent Node Lib

IoT Agent Node Lib website

IoT Agent Node Lib Installation guide

IoT Agent Node Lib User guide

IoT Agent JSON

IoT Agent JSON website

IoT Agent JSON Installation guide

IoT Agent JSON User guide

IoT Agent UL

IoT Agent UL website

IoT Agent UL Installation guide

IoT Agent UL User guide

CRATEDB

CrateDB website

MongoDB

Mongodb website

Postgres

Postgres website

CKAN

CKAN website

CKAN User guide

BIZ Ecosystem

Biz Ecosystem website

Biz Ecosystem Installation guide

Biz Ecosystem User guide

PRODUCT LINKS

KEYCLOAK

Keycloack website Keycloak Download page

Keycloack Installation and configuration guide

APInf

APInf website

APInf AcademyAPInf Installation guide*API UMBRELLA*

API Umbrella website

API Umbrella Installation guide

3Scale

3 Scale website

3 Scale Installation guide

QUANTUMLEAP

QuantumLeap website*CRATEDB*

CrateDB website

CrateDB Download page

CrateDB Installation guide

CrateDB Reference

MongoDB

MongoDB website

MongoDB Community

Elasticsearch

Elasticsearch website

Elasticsearch Installation guide

Elasticsearch User guide

Logstach

Logstach website

Logstach Installation guide

Logstach User guide

Postgres

Postgres website

Postgres Download page

Postgres Installation guide

Redis

Redis website

Redis Labs

Redis Download page

Redis Installation guide

CKAN

- CKAN website
- CKAN Installation guide

KIBANA

- Kibana website
- Kibana Installation guide
- Kibana User guide

GRAFANA

- Grafana website
- Grafana User guide

3D CITY DB

- 3D City DB website
- 3D City DB Github Realease
- 3D City DB Tutorials

GeoServer

- GeoServer website
- GeoServer Installation guide

IoT Agent JSON

- IoT Agent JSON website

GITHUB SMARTCITY

- CIP on Github