



# D3.8 Hosting Environment and IoF2020 Lab

## WP 3

November 15<sup>th</sup>, 2019

Development and provision of an experimental infrastructure serving experimentation and validation in the IoF2020 use cases.



## DOCUMENT IDENTIFICATION

<b>Project Acronym</b>	<b>IoF2020</b>
<b>Project Full Title</b>	Internet of Food and Farm 2020
<b>Project Number</b>	731 884
<b>Starting Date</b>	January 1 <sup>st</sup> , 2017
<b>Duration</b>	4 years
<b>H2020 Call ID &amp; Topic</b>	IOT-01-2016
<b>Date of the DoA</b>	23.08.2018
<b>Website</b>	<a href="http://www.iof2020.eu">www.iof2020.eu</a>
<b>File Name</b>	D3.8-HostingEnvironment-IoF2020Lab
<b>Date</b>	November, 15 <sup>th</sup> 2019
<b>Version</b>	112
<b>Status</b>	Final
<b>Dissemination level</b>	PU: Public
<b>Authors</b>	José M. Cantera (Lead author), Harald Sundmaeker, Rita Campos, Francisco de la Vega, Isabel Neira, Daoud Urdu, Ignacio Gómez, Francisco Maroto, Jason Fox, Karen Vega
<b>Contact details of the coordinator</b>	George Beers george.beers@wur.nl

## PROJECT SUMMARY

**The internet of things (IoT) has a revolutionary potential. A smart web of sensors, actuators, cameras, robots, drones and other connected devices allows for an unprecedented level of control and automated decision-making. The project Internet of Food & Farm 2020 (IoF2020) explores the potential of IoT-technologies for the European food and farming industry.**

The goal is ambitious: to make precision farming a reality and to take a vital step towards a more sustainable food value chain. With the help of IoT technologies higher yields and better-quality produce are within reach. Pesticide and fertilizer use will drop and overall efficiency is optimized. IoT technologies also enable better traceability of food, leading to increased food safety.

IoF2020 involves 33 use-cases organised around five trials (arable, dairy, fruits, meat and vegetables) develop, test and demonstrate IoT technologies in an operational farm environment all over Europe, with the first results that were realised in the first quarter of 2018.

IoF2020 uses a lean multi-actor approach focusing on user acceptability, stakeholder engagement and the development of sustainable business models. IoF2020 aims to increase the economic viability and market share of developed technologies, while bringing end-users' and farmers' adoption of these technological solutions to the next stage. The aim of IoF2020 is to build a lasting innovation ecosystem that fosters the uptake of IoT technologies. Therefore, key stakeholders along the food value chain are involved in IoF2020, together with technology service providers, software companies and academic research institutions.

Led by the Wageningen University and Research (WUR), the 100+ members consortium includes partners from agriculture and ICT sectors, and uses open source technology provided by other initiatives (e.g. FIWARE). IoF2020 is part of Horizon2020 Industrial Leadership and is supported by the European Commission with a budget of €30 million.

## EXECUTIVE SUMMARY

This document identifies and summarizes the main system design, configuration and deployment options for Digital Farming solutions based on **Open Platforms** that have to be considered by the technological solutions proposed to implement the different use cases, so that **scalability, interoperability, replicability, evolvability** and **maintainability** are achieved, allowing for the building of a robust and sustainable Internet of Things (IoT) ecosystem around the **IoF2020** project, and ultimately, around Smart Farming in Europe. All these assets can be grouped (and deployed) under the umbrella of the *IoF2020 Lab*, an integrated development environment for interoperable and composable Smart Agri-Food solutions in a **System of Systems**.

The **System of Systems** is a novel approach for the deployment of integrated, multi-vendor Digital Farming Solutions that is intended to maximize scalability, interoperability, maintainability and sustainability aspects. It is envisaged that no single company will be able to provide the best solution for all the challenges faced. Furthermore, the smart agri-food domain is very broad, specialized and diverse. In fact, it is usually needed to count with different solution developers to tackle each farmer's user story, which may involve multiple applications working at the same time.

In addition, there is an opportunity to **integrate innovative solutions coming from different parties**. It will be based on the integration of information generated by different solutions to build a holistic picture of what is going on at a farm. As a consequence, *Farm Management Information Systems* will be able to provide users an integrated, holistic view, encompassing information from different verticals and mashing-up the best of breed user interfaces. In the end, Context Information associated to a farm will be enriched with contributions coming from different vertical solutions (**System of Systems**), all of them able to share data among each other and enabling a further optimization of processes, saving time, money and resources.

The figure below shows an overview of the overall approach and context behind the System of Systems approach for Digital Farming. Expected benefits of the System of Systems approach are numerous, it:

- Enables an open and competitive marketplace of compatible farm management systems and vertical smart farming solutions.
- Lowers costs to achieve interoperability of vertical solutions or their integration with farm management systems.
- Enables a healthy public-private collaboration ecosystem around Digital Farming.
- Improves user friendliness: easing the management around a single stop shop offered by Farm Management Information Systems.
- Enables modularity by adding platform components parallel to business needs.

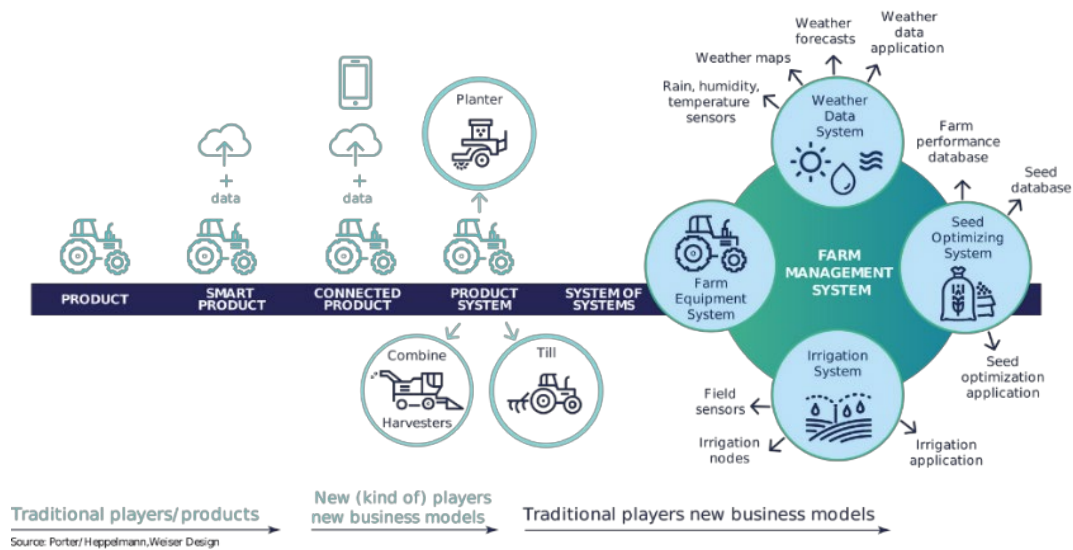


Figure 1: System of Systems overall approach.

To realize the System of Systems vision, the availability of shared, well-adopted information models is a key interoperability mechanism for enabling a global market for IoT-enabled Digital Farming. Such models provide an essential element in the common technical ground needed for standards-based innovation, enabling practical replicability and portability of smart agri-food solutions. This allows for sector-specific focus in a procurement or development process, while maintaining cross-domain consistency.

In order to realise the IoF2020 vision of replicable and interoperable Digital Farming solutions, it is necessary to count with off-the-shelf, integrated and configurable **Open Platforms** that can solve major development challenges, while offering an architecture compliant with the Minimum Interoperability Mechanisms (MIMs). This document describes FIWARE and 365FarmNet (described in Deliverable **D3.6**), as they can be used to materialize the System of Systems vision.

Another important need in a Digital Farming environment, based on System of Systems, is how to monetize data or services. Different IoF2020 partners offer open software components that can be instantiated to experiment with these innovative services. Deliverable **D3.6** already described them, and this report will introduce their role, configuration and deployment options in a System of Systems environment.

The final aim of this work is to help use case stakeholders, product owners and developers to design, deploy and configure Digital Farming Systems of Systems around Open Platforms, leveraging common technology enablers, software components, and related architectures that guarantee the creation of a sustainable ecosystem of portable solutions for the Farm and Food sector.

# TABLE OF CONTENTS

<b>1</b>	<b>Introduction .....</b>	<b>11</b>
1.1	Approach and Methodology .....	12
1.2	The IoF2020 Lab.....	14
<b>2</b>	<b>Overview of Use Cases Usage of Components .....</b>	<b>15</b>
2.1	Use Case Overview .....	15
2.2	Use Cases relying on Open Platforms.....	16
<b>3</b>	<b>The Context Broker in Seven IoT Solutions .....</b>	<b>19</b>
3.1	UC1.2 Precision Crop Management .....	19
3.2	UC1.4 Farm Machine Interoperability .....	20
3.3	UC3.3 Automated Olive Chain.....	20
3.4	UC4.2 Chain-Integrated Greenhouse Production.....	21
3.5	UC5.1 Pig Farm Management .....	22
3.6	UC5.2 Poultry Chain Management .....	23
3.7	UC5.3 Meat Transparency and Traceability .....	24
3.8	Summary.....	25
<b>4</b>	<b>System of Systems Approach .....</b>	<b>26</b>
4.1	Overall concept and architecture .....	26
4.2	Context Broker .....	27
4.3	Harmonized Data Models .....	28
4.4	Materialization of the SoS approach.....	30
4.5	Node-Red enabler for the SoS approach .....	32
4.6	SoS case of study implemented using Node-Red .....	34
4.7	Weather Station data integration in a SoS approach .....	36
4.8	Service Monetization – CoatRack.....	42
4.9	Data marketplace.....	45
4.10	Configurable Dashboards .....	48
4.11	Conclusions.....	49
<b>5</b>	<b>Guidelines .....</b>	<b>51</b>
5.1	Security Privacy and Trust.....	51
5.2	Usability.....	52
<b>6</b>	<b>Added Value for different Business Models.....</b>	<b>59</b>
6.1	Added Value of Reusable Components.....	59
6.2	Added Value of the System of Systems Approach .....	59

<b>7</b>	<b>Conclusions .....</b>	<b>63</b>
<b>8</b>	<b>Appendix A: Node-Red FIWARE Extension Nodes .....</b>	<b>64</b>
<b>9</b>	<b>Appendix B: Step-by-Step Explanation of using CoatRack.....</b>	<b>67</b>

## LIST OF TABLES

Table 1:	Usage of main FIWARE components in initial use cases.....	17
Table 2:	Usage of main FIWARE components in open call use cases. ....	17
Table 3:	Data models supporting a system of systems approach.....	29
Table 4:	Activities to develop a weather system.....	37
Table 5:	Weather data models.....	39
Table 6:	Synergies between weather and other systems.....	41
Table 7:	UX/UI Questions on an Agile Project.....	54
Table 8:	Usage Testing.....	55
Table 9:	Usability Analysis (Example) .....	56

## LIST OF FIGURES

Figure 1:	System of Systems overall approach. ....	5
Figure 2:	IoF2020 vision and approach and the context of this deliverable (IoT Lab).....	11
Figure 3:	Systems of Systems overall approach.....	13
Figure 4:	Overview of all use cases of the IoF2020 project.....	15
Figure 5:	The overall WP3 vision of the system of systems. ....	18
Figure 6:	Functional architectural view of the UC1.2's solution. ....	19
Figure 7:	Functional architectural view of the UC1.4's solution. ....	20
Figure 8:	Functional architectural view of the UC3.3's solution. ....	21
Figure 9:	Functional architectural view of the UC4.2's solution.....	22
Figure 10:	Functional architectural view of the UC5.1's solution. ....	23
Figure 11:	Functional architectural view of the UC5.2's solution. ....	24
Figure 12:	Functional architectural view of the UC5.3's solution. ....	25
Figure 13:	Systems of Systems approach using FIWARE technologies .....	27
Figure 14:	Orion Context Broker high level architecture.....	28
Figure 15:	Animal Data Model preliminary analysis (source <i>Sharebeef</i> use case). ....	29
Figure 16:	Materialization of the System of Systems approach using Context Broker.....	31
Figure 17:	Materialization of the System of Systems approach (pull mode).....	32
Figure 18:	Materialization of the System of Systems approach (push mode). ....	32
Figure 19:	Node-Red FIWARE Extension Package Overview .....	33
Figure 20:	Case of Study for the SoS approach (push mode).....	34
Figure 21:	Node-Red flow that realizes NGSI Agent 1 (NGSIV2) .....	35
Figure 22:	Node-Red flow that realizes NGSI Agent 2 (MQTT).....	35
Figure 23:	Node-Red flow that realizes NGSI Agent 3 (HTTP REST).....	36
Figure 24:	Proposed architecture of a Weather information System .....	38
Figure 25:	Example of Property Graph for Weather Stations in Agri-Food (Work in progress).....	41
Figure 26:	Schematic interaction of service providers and consumers, facilitated by CoatRack. ....	43
Figure 27:	General architecture overview of CoatRack for Service Monetization. ....	43
Figure 28:	Overview of the System of systems architecture featuring Data Marketplace and Configurable Dashboards. ....	45
Figure 29:	CKAN Extensions .....	47
Figure 30:	Creation of a new tenant.....	47



Figure 31:	Adding users to tenants .....	48
Figure 32:	Greenhouse dashboard showcased in the Prague IoF2020 Event.....	49
Figure 33:	Security by design.....	51
Figure 34:	SPT from a WP3 perspective. ....	52
Figure 35:	Design Thinking Processes .....	55
Figure 36:	Context Broker Config Node UI. ....	64
Figure 37:	NGSI Entity Node UI. ....	64
Figure 38:	NGSI Dataset Node UI.....	65
Figure 39:	NGSI Subscription Node UI .....	65
Figure 40:	NGSI Update Node UI .....	66
Figure 41:	CoatRack Homepage.....	67
Figure 42:	GitHub Account Login. ....	67
Figure 43:	Permissions between GitHub and CoatRack.....	68
Figure 44:	New User Registration. ....	68
Figure 45:	Getting Started Wizard.....	69
Figure 46:	Getting Start Wizard (Service Name) .....	69
Figure 47:	Getting Started Wizard (Service URL).....	70
Figure 48:	Getting Started Wizard (Payment Policy). ....	70
Figure 49:	Download and test of the Service Gateway.....	71
Figure 50:	Coatrack Dashboard (Testing Own Service) .....	71
Figure 51:	Service Access Policy Update .....	72
Figure 52:	Service Set as Public.....	72
Figure 53:	List of Public Services.....	73
Figure 54:	Get access to service button. ....	73
Figure 55:	Subscription to Public Service. ....	74
Figure 56:	API Keys to access the Services. ....	74
Figure 57:	CoatRack Dashboard.....	75

## ABBREVIATIONS

AEF	Agricultural Industry Electronics Foundation
AgGateway	Non-profit organization for industry's transition to digital agriculture
COTS	Commercial off the Shelf Product
D	Deliverable
ETSI	European Telecommunications Standards Institute
EU	European Union
FIWARE	Future Internet Software Initiative
GS1	Global Standards One
IoF 2020	Internet of Food and Farm 2020
IoT	Internet of Things
ISO	International Organization for Standardization
ITU-T	International Telecommunication Union - Telecommunication Standardization Sector
LSP	Large Scale Pilot
MVP	Minimum Viable Product
SME	Small and Medium sized Enterprise
UC	Use Case
UN/CEFACT	United Nations Centre for Trade Facilitation and Electronic Business
WP	Work package

# 1 Introduction

One of the main objectives of the IoF2020 project is to provide a robust technological offer around Open Platforms (realising a reference architecture) for standards-based, interoperable and replicable solutions that deliver added-value functionalities to various stakeholders in the food and farms domain. In other words, to convince and demonstrate use case developers the advantages of relying on open platforms, to facilitate and trigger collaboration and synergies and as input for further awareness with external stakeholders (see figure 2). The final goal is enabling an Internet of Things (IoT) *ecosystem* around the IoF2020 project and ultimately around Smart Farming in Europe.

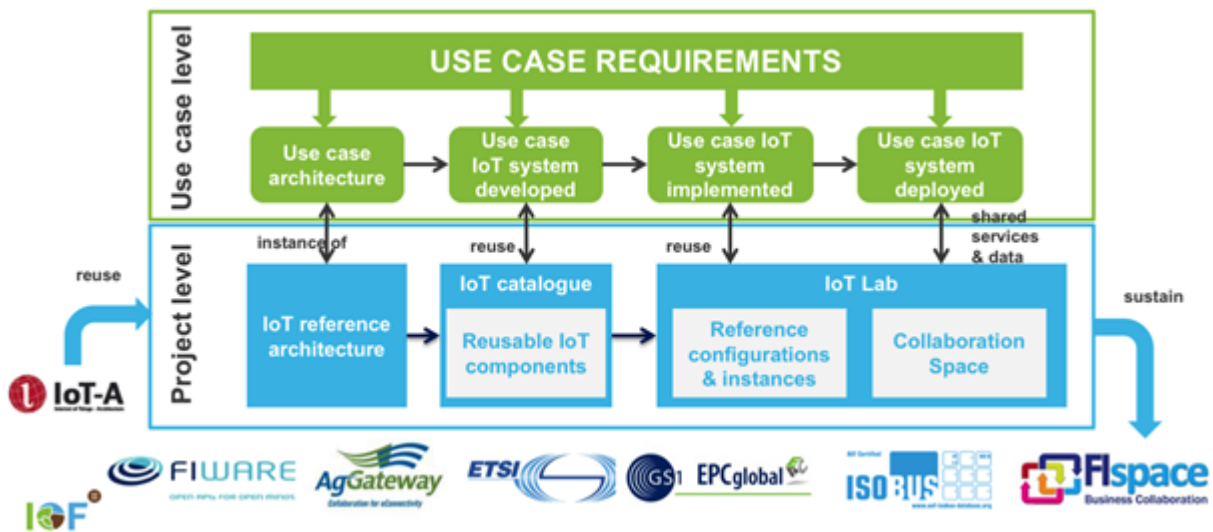


Figure 2: IoF2020 vision and approach and the context of this deliverable (IoT Lab).

This document identifies and summarizes the main system design, configuration and deployment options for Digital Farming solutions based on **Open Platforms** that have to be considered by the technological solutions proposed to implement the different use cases, so that **scalability, interoperability, replicability, evolvability** and **maintainability** are achieved, allowing to build a robust and sustainable Internet of Things (IoT) ecosystem around the **IoF2020** project, and ultimately, around Smart Farming in Europe. Particularly, to make use case developers aware of the system design and associated deployment and configuration options (box labelled as IoT Lab in Figure 2), *relying on Open Platforms*, and to facilitate and trigger sustainability, collaboration and synergies that involve multiple application providers and internal and/or external stakeholders.

To better understand the context around this deliverable, it is encouraged to read **D3.2** which contains the technological analysis and architecture of the different solutions proposed to address the IoF2020 use cases. Besides, **D3.9** should be read so that a proper understanding of the use case *synergies* is gained. The recommendations made by **D3.3** have served as a valuable input for *Task 3.3, Open Platforms*, which is closely related to *Task 3.4*. In fact, it has guided the definition and selection of platforms (some of them described by **D3.6**) and components during the upcoming implementation of the use cases to be developed. The recommendations made by **D3.6** have provided a “common ground” to leverage the IoF2020 IoT-based innovations based on Open Platforms in the upcoming phases of the project, both within each use case (including IoF2020 Open Call winners), spawning across multiple use cases or even beyond the traditional limits of the agri-food and farming sector.

*Chapter 2* makes a summary and an update on the usage of Open Platforms and associated components within IoF2020 trials and use cases. *Chapter 3* provides a summary of the **main lessons learnt** from the usage, deployment and configuration of Open Platforms and components for IoT-based digital farming and ecosystems. *Chapter 4* introduces the concept of **System of Systems** which is a novel approach for the deployment of integrated, multi-vendor Digital Farming Solutions that is intended to maximize scalability, interoperability, maintainability and sustainability aspects, towards the realization of an IoT Lab (see Figure 2) for the Agri-food sector led by the results of the IoF2020 project. *Chapter 5* provides additional guidelines on security, privacy and usability that can be complementary and orthogonal to the previous aspects

discussed. *Chapter 6* discuss about the business value provided by the approaches suggested by this document

Finally, the last chapter captures conclusions, outlook and action plan during next phases of the project.

## 1.1 Approach and Methodology

In order to provide a robust technological offer around Open Platforms (realising a reference architecture) for standards-based, interoperable and replicable solutions, WP3 has been working on:

- Identifying the key and common Minimal Interoperability Mechanisms, MIMs (named as Interoperability Points by **D3.3**) that apply to the different use cases and trials, allowing secure and controlled exchange of information and capabilities across heterogeneous components (*Deliverable 3.3*).
- In close collaboration with WP2, identifying the requirements (both functional and non-functional) of the use cases and trials (*Deliverable 3.7*)
- Identifying reusable IoT components that can be part of, leverage or complement the offer around Open Platforms (*Deliverable 3.7*).
- Promoting, enhancing and extending Open Platforms and related configurations that can solve major development challenges, while offering an architecture compliant with the MIMs (*Deliverable 3.6*).

The Open Platforms referred by **D3.6** address different functional and non-functional requirements and use cases (as described by **D3.7**) that have to be tackled when deploying a Smart Food and Farming solution, in accordance with the Minimal Interoperability Mechanisms (MIMs) described by **D3.3**. In fact, MIMs are of paramount importance and have resulted from the analysis made by D3.3 of the different trials and use cases conducted, identifying and generalizing the main architectural layers and standards.

Even though **D3.3** presented the proposed architectural view and MIMs of single, vertical solutions intended to solve specific problems, it is envisaged that no single company will be able to provide the best solution for all the challenges faced. Furthermore, the smart Agri-food domain is very broad, specialized and diverse. In fact, it is usually needed to count with different solution developers to tackle each farmer's user story, which may involve multiple applications working at the same time. For instance, the technology, devices and applications needed for silo monitoring (sensors based on 3D cameras), are very different than those required for livestock control (collars worn by animals). Also, the graphical user interfaces to be devised are quite different. The latter needs maps and other intensive geospatial data, whereas the former needs a compelling graphical representation of silos and their filling levels.

Therefore, it is envisaged that no single company will be able provide the best solution for all Agri-Food challenges. Furthermore, there is an opportunity to **integrate innovative solutions coming from different parties**. It will be based on the integration of information generated by different solutions to build a holistic picture of what is going on the farm. As a consequence, *Farm Management Information Systems*, optionally deployed under a cloud instance of the IoT Lab as suggested by Figure 2, will be able to provide users an integrated view, encompassing information from different verticals and mashing-up the best of breed user interfaces. In the end, Context Information associated to a farm will be enriched with the contributions coming from different vertical solutions (**System of Systems**), all of them able to share data among each other and enabling a further optimization of processes, saving time, money and resources.

The figure below shows an overview of the overall approach and context behind the System of Systems approach for Digital Farming. The expected benefits of the System of Systems approach are:

- Enables an open and competitive marketplace of compatible farm management systems and vertical smart farming solutions.
- Lower costs to achieve interoperability of vertical solutions or their integration with farm management systems.
- Enables a healthy public-private collaboration ecosystem around Digital Farming.
- Improves user friendliness: easing the management around a single stop shop offered by Farm Management Information Systems.

- Enables modularity by adding platform components parallel to business needs.

To realize the System of Systems vision, the availability of shared, well-adopted information models is a key MIM for enabling a global market for IoT-enabled Digital Farming. Such models provide an essential element in the common technical ground needed for standards-based innovation, by making replicability and portability of smart agri-food solutions practical. This allows for sector-specific focus in a procurement or development process, while maintaining cross-domain consistency. **D3.6** described extensively the approach to Data Modelling followed in IoF2020. In this report, it will be shown how harmonized Data Models can be used to face the challenges posed by the System of Systems approach.

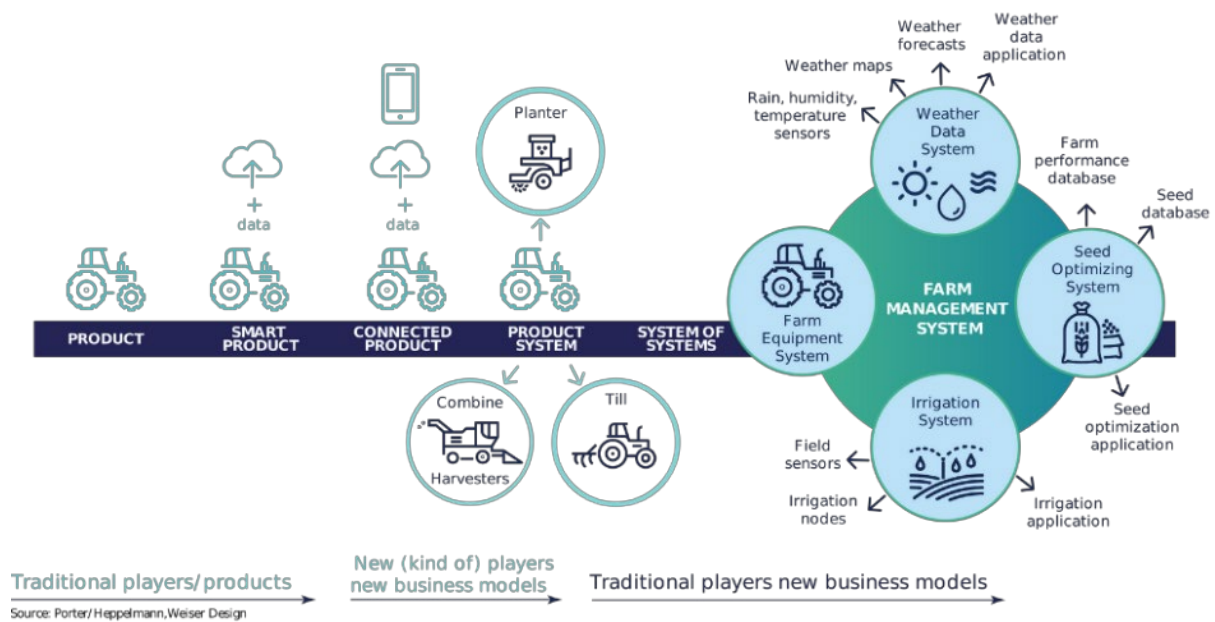


Figure 3: Systems of Systems overall approach

Besides, in order to realise the IoF2020 vision of replicable and interoperable Digital Farming solutions, it is necessary to count with off-the-shelf, integrated and configurable **Open Platforms** that can solve major development challenges, while offering an architecture compliant with the MIMs. Among others, FIWARE open source components, and existing FMIS products, for instance 365FarmNet (described in **D3.6** and **D4.9**) and AkkerWeb (described on **D4.9**), can be used to materialize the Systems of Systems vision. For a more detailed view of the Agri-food platform market offering, **D4.9** can be consulted.

Another important need in a Digital Farming environment, based on System of Systems, is how to monetize data or services. Different IoF2020 partners are offering open software components that can be instantiated to experiment with these innovative services. **D3.6** already described them, and this report will introduce their role, configuration and deployment options in the System of Systems environment.

The final aim of this work is to help use case stakeholders, product owners and developers to design, deploy and configure Digital Farming Systems of Systems around Open Platforms, leveraging common technology enablers, software components, and related architectures that guarantee the creation of a sustainable ecosystem of portable solutions for the Farm and Food sector. In the end, that will foster:

- the flourishing of a marketplace composed by different vertical solutions capable of interoperating and integrating into a broader system of farm management,
- the identification and development of IoT reusable components and reference configurations and compositions in the framework of a common architecture

The final result of this work will allow WP2 developers to make informed decisions about the usage and deployment of Open Platforms (in a cloud instance of the IoT Lab as suggested by Figure 2) in a System of Systems configuration that will foster interoperability, evolvability and maintainability, all for the benefit of use case owners, developers, stakeholders and the farm and food domain.

## 1.2 The IoF2020 Lab

The **IoF2020 Lab** is an *integrated development environment* that allows the publication, development and (optionally) deployment of Smart Agri-Food solutions following a **System of Systems** approach. The IoF2020 Lab is composed by a series of different assets (developed and enhanced under WP3) aimed at facilitating the publication, analysis, design, development and deployment of Digital Farming solutions within a System of Systems configuration:

- The IoF2020 IoT Catalogue
- The FIWARE Context Broker
- The Data Marketplace component
- The Service Monetization component
- The Smart Agri-Food Data Models
- The hosting facilities for proof of concept, offered by the *FIWARE Lab* cloud hosting environment
- Other satellite components (Node-Red connector, Weather Data Enabler, other additional FIWARE GEs etc.)

The **IoF2020 IoT Catalogue** is a web-based catalogue for Smart Agri-Food solutions based on Internet-of-Things (IoT), and available at <https://www.iotcatalogue.com>. The IoT-Catalogue brings IoF2020 Lab users and technology providers together, from the domain needs to Smart Agri-Food products (and back) via validated solutions with components, assembly guides, and more. The IoT-Catalogue shows value propositions, highlighting the added value to the company/user. In addition, it shows Digital Farming problems in easy to understand terms, describing issues needing a technological solution, i.e. different kind of functions needed to get the problems solved, where they will be applied and domain depicting its area of interest.

The **FIWARE Context Broker** is a data broker, supporting ETSI NGSI-LD standards, which allows to publish, consume and subscribe to data (and associated metadata) coming from multiple devices and sources. In fact, it allows applications to get access to (harmonised) data entities, regardless data sources. The broker may store data in the short to medium term using a data store based on mongoDB.

The **Data Marketplace** is representing a complete framework, which supports data management, including the means for visualizing, publishing, and monetizing data while enforcing access control and managing data usage terms and conditions.

The **Service Monetization** is enabling providers and users of services to offer and also use different services, while assuring that no details of a business interaction are stored in a central system, but the history of service usage, as well as to assure a proper authentication and authorisation of service access.

The **Smart Agri-Food Data Models** enable semantic interoperability. In fact, the availability of shared, well-adopted information models is a key interoperability mechanism for enabling a global market for IoT-enabled Digital Farming. Such models provide an essential element in the common technical ground needed for standards-based innovation, by making replicability and portability of Smart Farming solutions practical. This allows for a sector-specific focus in a procurement or development process, while maintaining cross-domain consistency.

The **hosting facilities** offered by the *FIWARE Lab* allow developers to deploy their Smart Agri-Food solutions in a cloud hosting environment which is privacy-friendly, free of charge, but at the same time facilitates the automated deployment of the main software components (Context Broker, Data Marketplace, etc.). The FIWARE Lab is a non-commercial sandbox environment (maintained by the FIWARE Foundation and other stakeholders) where innovation and experimentation based on FIWARE technologies take place. Entrepreneurs and individuals can test the technology as well as their applications on FIWARE Lab.

Other **satellite components** are also part of the IoF2020 Lab. Namely, the Weather Data Enabler currently under development, the Node-Red connector or other FIWARE Generic Enablers (GEs) that can complement the Context Broker (security enablers, historical data enablers, etc.).

More information on the IoF2020 Lab assets can be found along this deliverable and on **D3.6**.



## 2 Overview of Use Cases Usage of Components

IoF2020 has currently **33** use cases, distributed over five different trials (see Figure 4):

- The **arable trial** focuses on wheat, soy bean and potato production and processing in different climate zones. It includes activities across the cropping cycle and machine-to-machine communication. Overall, the use of IoT in arable farming can help to reduce pesticide, fertilizer and energy use, while increasing transparency and food safety.
- The **dairy trial** explores the usefulness of collecting real-time sensor and GPS location data throughout the whole dairy chain, i.e. from grass to glass. It includes monitoring outdoor grazing cows, application of machine learning technologies and cloud-based services.
- The **fruit trial** aims to gather data on pre- and post-harvest losses to increase the yield and quality of fruits, as well as ensure better traceability of fruit products in relation to the protected designation of origin.
- The **vegetables trial** aims to improve the quality and productivity of lettuce and tomatoes in the controlled cultivation and weeding of the vegetables in organic production. The trial includes use cases in different climate conditions, such as fully controlled indoor greenhouses to open-air cultivation.
- The **meat trial** aims to improve the meat production chains value, addressing activities related to management and optimization of meat production, transparency and traceability. The trial includes use cases related to pork, beef and poultry.

### 2.1 Use Case Overview

Each use case is developing an IoT solution for a specific problem. This means IoF2020 will produce a set of **33 IoT solutions** in the agri-food domain. Although the use solutions are individual, there is collaboration among the use cases, and relevant synergies have been identified and followed.

WP3 has been having an active role in identifying synergies among use cases, recognizing needs that can be fulfilled by generic IoT reusable components, and involving use cases in suitable task forces.



Figure 4: Overview of all use cases of the IoF2020 project.

The 33 use cases can be divided into two groups, representing their current maturity level:

- **19 use cases** have started at the beginning of the project, over two years ago, and have already produced at least one minimum viable product (MVP) of their individual IoT solutions.
- **14 uses cases** were added to the project through an **open call**, and have started their activities in *January 2019*. These use cases are currently specifying and starting implementation of their individual solutions.

## 2.2 Use Cases relying on Open Platforms

The identification of synergies among the use cases, starts with the identification of common challenges and use of similar IT and IoT components. WP3 is looking to facilitate a **systems-of-systems approach**, enabling *interoperability and communication among solutions*, supported by IoT reusable components. In this sense, WP3 has been working on **common data models, a data marketplace, service monetization**, among other components. WP3 has been identifying use cases that would benefit the most from adopting these reusable components. The systems-of-systems approach being developed in WP3 (further elaborated by chapter 4) is centred around the use of the **FIWARE Orion context broker**, as an enabler to exchange data among systems. D3.6 describes how the **FIWARE Orion context broker** can be deployed and configured.

Therefore, WP3 is particularly interested in how the individual solutions are using FIWARE components, especially the Orion Context Broker. These use cases have the backbone to adopt the reusable components being developed by WP3 and potentiate the results. The analysis of usage of components has to be differentiated for the two groups of use cases. For the initial use cases, we can study the solutions implemented, while for the open call use cases, we rely on the architectures specified in the work plans.

Seven initial use cases are using the Orion Context Broker in their individual solutions:

- UC1.2 Precision Crop Management
- UC1.4 Farm Machine Interoperability
- UC3.3 Automated Olive Chain
- UC4.2 Chain-Integrated Greenhouse Production
- UC5.1 Pig Farm Management
- UC5.2 Poultry Chain Management
- UC5.3 Meat Transparency and Traceability

The context broker has been identified by these use cases as a component to help solve the following ICT problems:

- Generate new knowledge through data mining;
- Share data with stakeholders;
- Send data to cloud;
- Measure and sense diverse variables on the field; and
- Analyse data.

Use cases UC1.2 Precision Crop Management, UC5.1 Pig Farm Management and UC5.2 Poultry Chain Management have developed full IoT solutions based on FIWARE; not only the context broker, but also IdM Key Rock and PEP Proxy Wilma (among other components). The following chapter of this deliverable elaborates on some of the lessons learned in this development.

The following Table 1 lists the use cases adopting the main identified FIWARE components.



Table 1: Usage of main FIWARE components in initial use cases.

Component	Use Cases
Orion Context Broker	UC1.2 Precision Crop Management UC1.4 Farm Machine Interoperability UC3.3 Automated Olive Chain UC4.2 Chain-Integrated Greenhouse Production UC5.1 Pig Farm Management UC5.2 Poultry Chain Management UC5.3 Meat Transparency and Traceability
IdM KeyRock	UC1.2 Precision Crop Management UC5.1 Pig Farm Management UC5.2 Poultry Chain Management
PEP Proxy Wilma	UC1.2 Precision Crop Management UC5.1 Pig Farm Management

The following chapter of this deliverable details all FIWARE components used by each use case, presenting the architecture of the IoT solutions developed and elaborates on some of the lessons learned during this development.

Studying the work plans of the open call use cases, allows identifying which solutions will adopt these or other FIWARE open platform components. The use cases presented an architecture and listed reusable components in their work plans. From the *14 open call use cases*, **seven** will adopt the Orion Context Broker, representing a quite significant increase in the adoption of this open paradigm.

This supports the WP3 perspective of system-of-systems approach and increases the potential for its demonstration. The following figure enriches the previous one with the open call use cases adopting the main identified FIWARE components.

The following Table 2 lists the open call use cases adopting the main identified FIWARE components.

Table 2: Usage of main FIWARE components in open call use cases.

Component	Use Cases
<b>Orion Context Broker</b>	UC1.5 Potato Data Processing Exchange UC1.6 Data-Driven potato Production UC3.5 Smart Orchard Spray Application UC4.5 Enhanced Quality Certification System UC5.4 Decision Making Optimization in Beef Supply UC5.5 Feed Supply Chain Management UC5.6 Interoperable Pig Tracking
<b>Cygnus</b>	UC1.5 Potato Data Processing Exchange UC4.5 Enhanced Quality Certification System UC5.5 Feed Supply Chain Management UC5.6 Interoperable Pig Tracking

Component	Use Cases
<b>IdM KeyRock</b>	UC1.5 Potato Data Processing Exchange UC4.5 Enhanced Quality Certification System UC5.6 Interoperable Pig Tracking
<b>PEP Proxy Wilma</b>	UC4.5 Enhanced Quality Certification System
<b>CKAN</b>	UC1.5 Potato Data Processing Exchange

IoF2020 will then have thirteen use cases including the Orion Context Broker in their individual IoT solutions, (a little over **40% of all use cases will be using FIWARE generic enablers**, mainly supported by the Orion context broker). This provides excellent opportunities to test the components being developed in the scope of WP3 and the overall system-of-systems approach.

There is then the potential of **combining 14 individual IoT solutions** in a System of Systems approach. We should be able to find a use case able to demonstrate the use of their solution with the addition of services from another solution (enabled by the system of systems approach). The ultimate goal will be to demonstrate a larger system combining the individual IoT vertical solutions, as presented in Figure 5.

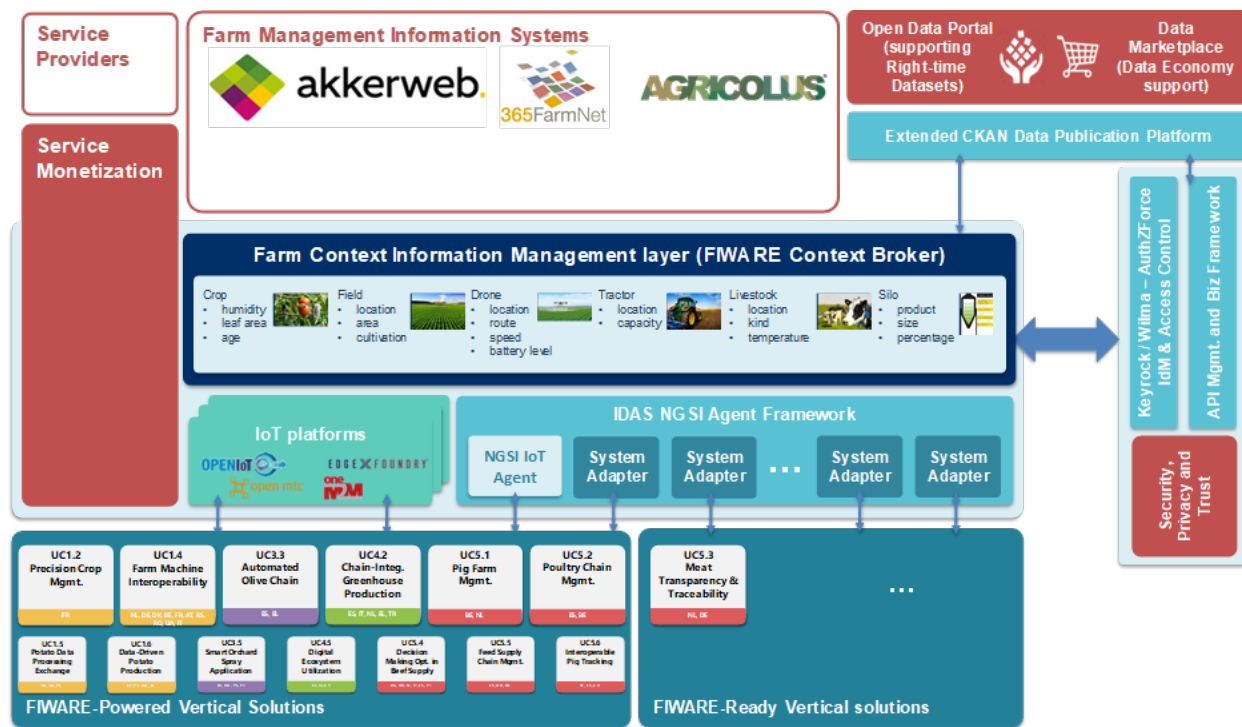


Figure 5: The overall WP3 vision of the system of systems.

### 3 The Context Broker in Seven IoT Solutions

As described in the previous chapter, seven of the initial use cases have adopted the FIWARE context broker in their IoT solutions. Within WP3, it is important to understand the scope of this adoption and, particularly, challenges faced, and lessons learned. This will increase potential of the components and decrease overhead in the development of the new solutions in the open call use cases.

This section describes lessons learnt for the most prominent use cases of IoF2020 using open platforms.

#### 3.1 UC1.2 Precision Crop Management

Nitrogen and water availability are the two main limiting factors in wheat production. By utilizing sensor data embedded in a low-power, long-range network infrastructure and computer modelling for precision crop management, farmers can greatly improve the efficiency of nitrogen and water use in wheat production. This UC demonstrates how farming based on digital technologies can help farmers to reduce the environmental footprint of wheat production, improve their working environment, and save costs through a more efficient use of nitrogen and water.

Figure 6 represents the current functional view of the solution being developed.

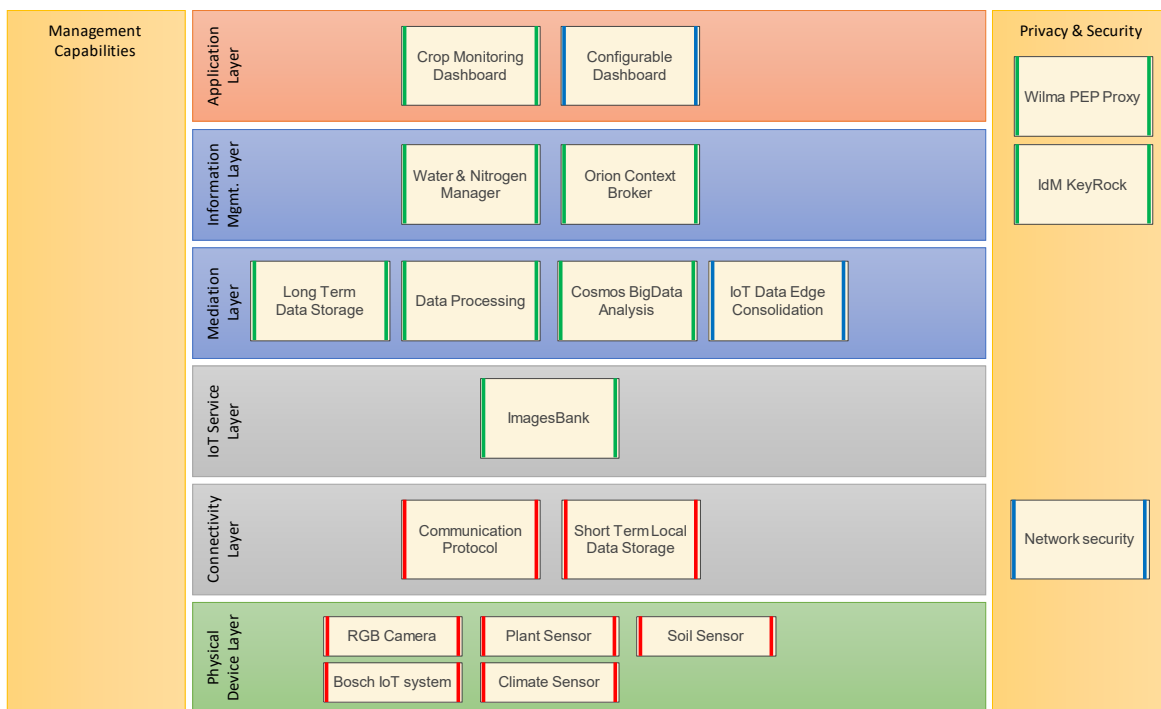


Figure 6: Functional architectural view of the UC1.2's solution.

One of the partners in the use case hosts a FIWARE Lab node with the components Orion context broker, PEP Proxy, and Cosmos. They are also implementing their own PEP proxy and KeyRock instances to manage security access and integration.

The FIWARE components have been rebuilt from source and installed following the reference manual provided on GitHub for each reference implementation, except Orion Context Broker that has been installed from the official FIWARE yum repository.

In this use case, one of the motivations to select FIWARE was to define an open architecture and implement an infrastructure to ensure a long-term sustainability. Such an open architecture has the potential to implement or communicate with other IoT individual solutions.

This use case has opted for their own instances of PEP proxy and KeyRock to replace the FIWARE Lab account management, providing more robust security to their solution.

### 3.2 UC1.4 Farm Machine Interoperability

Farmers have arable machines from different vendors. They want these machines to work seamlessly together. Work orders should be sent from the Farm Management Information System (FMIS) to the appropriate machines and after the tasks have been executed, the work records should be sent back to the FMIS. In addition, the farmer wants to see real-time data of the machine. This use case tries to find solutions for the interoperability of machines and FMIS.

Figure 7 represents the current functional view of the solution being developed.

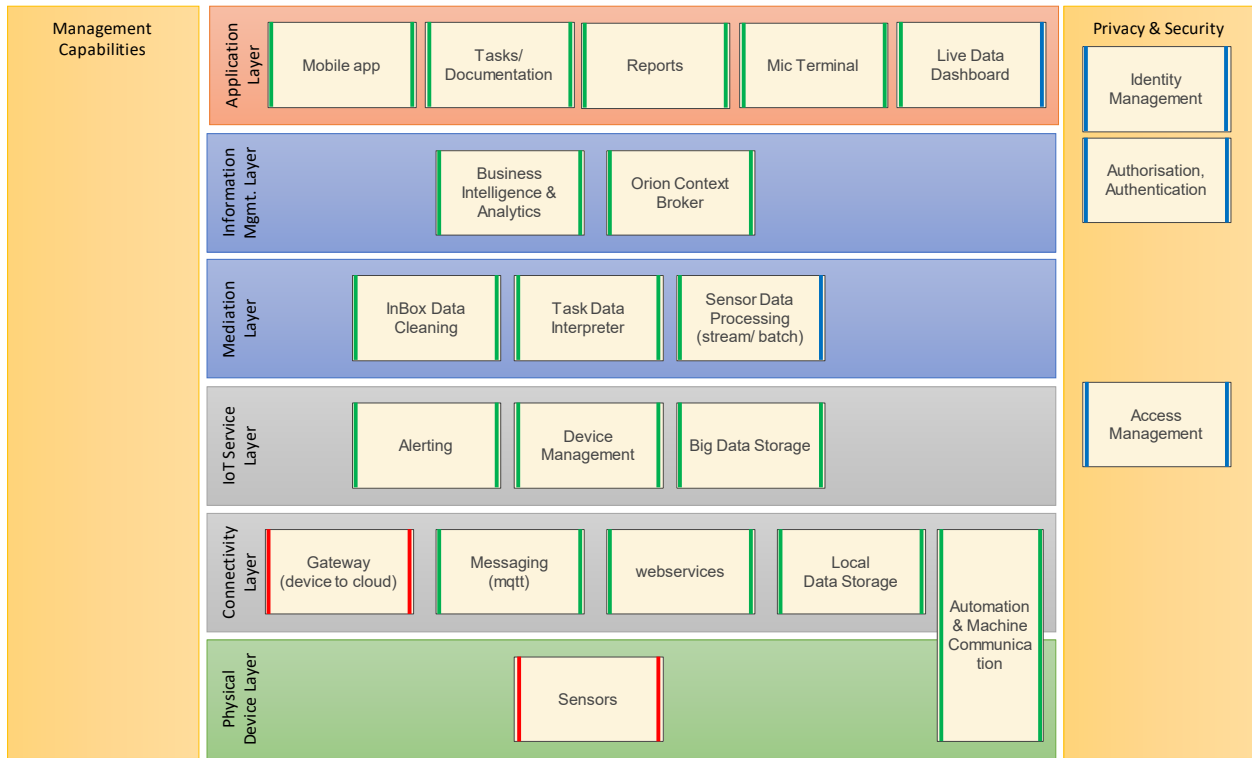


Figure 7: Functional architectural view of the UC1.4's solution.

This use case has been developing a solution for interoperability in real-time, consisting of an overarching framework to harmonise and manage the context information of this specific domain (arable), and implementing AG-equipment specific solutions. In this scope, the use case is focusing on the implementation of a Web-API based on the NGSI v2 adopted by FIWARE and its corresponding NGSI-LD information model, managed by ETSI. The Web-API represents a harmonised starting point to share context information and to connect actors and machinery manufacturers under a common semantic.

The use case has implemented a mobile app-based harvest logistics system that monitors, documents and optimises harvest logistics. This app includes posting and calling entities through proof-of-concept API using NGSI-LD standards.

### 3.3 UC3.3 Automated Olive Chain

The EU is the largest producer (accounting for almost three quarters) and consumer (accounting for almost two thirds) of olive oil in the world. In 2014, the EU produced 2.482.500 tons of olive oil and 794.000 tons of table olives, while consuming 1.731.000 tons and 530.500 tons respectively. However, increasing competition from other countries and the rapid decline in olive plantations caused by the bacterium *Xylella fastidiosa* puts the olive sector under pressure. Therefore, this use-case explores how IoT technologies can help face these challenges. Figure 8 represents the current functional view of the solution being developed.

Figure 8 represents the current functional view of the solution being developed.

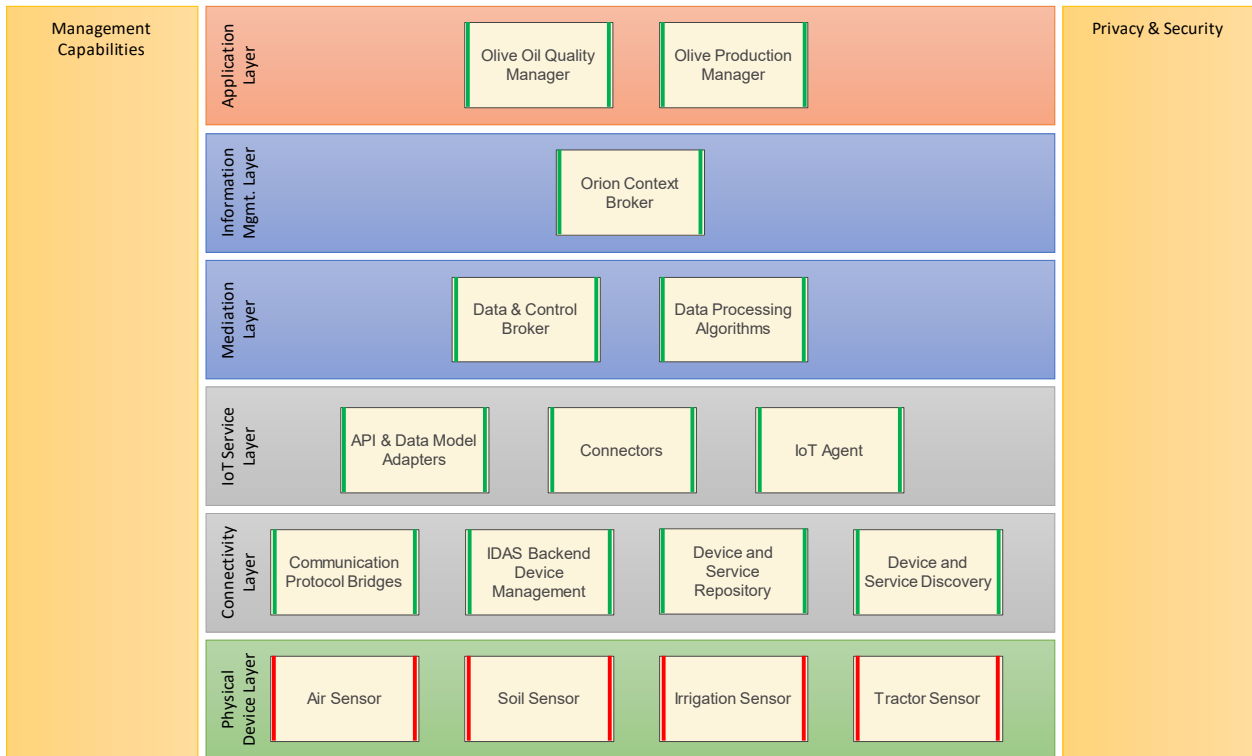


Figure 8: Functional architectural view of the UC3.3's solution.

In this use, the Orion Context Broker was adopted to facilitate data sharing among the several stakeholders involve.

### 3.4 UC4.2 Chain-Integrated Greenhouse Production

This use case is developing an IoT web-based traceability and decision support system in the greenhouse tomato production involving large amount of data, physical and virtual sensors, models. The solution also includes algorithms focusing on important aspects like water and energy use efficiency, safety, transparency, for both conventional and organic supply chain traceability systems of tomato.

Figure 9 represents the current functional view of the solution being developed.

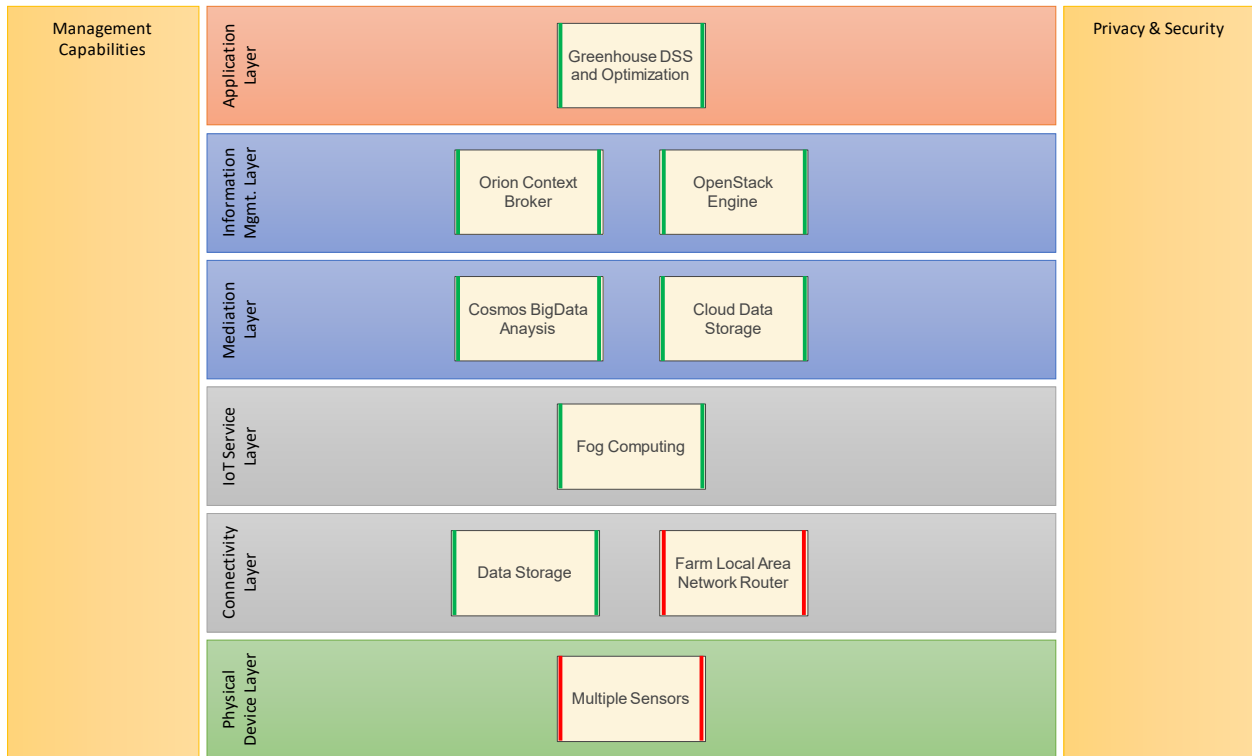


Figure 9: Functional architectural view of the UC4.2's solution

This use case decided on an implementation of the Orion Context Broker as the basis for its IoT platform, to maximise the support for data sharing in the supply chain. This cloud platform powered by FIWARE, where data is anonymised and protected, offers a secure infrastructure and app server for data distribution to users. The use case has faced some challenges in combining two different data modelling approaches: a relational database defined in the use case for traceability issues, and the non-relational approach used in FIWARE. The current solution has completely migrated to a non-relational database using mongoDB.

The use case has also faced challenges related to the integration of different information provider stations, which do not send data directly to the Orion Context Broker. The use case developed a system to check sensor value modifications, verify if there is new data and activate the subscription and notification system of the Context Broker.

### 3.5 UC5.1 Pig Farm Management

The demands for sustainable production in the competitive pig farming industry can only be met when the whole production chain from farm to fork becomes more efficient. To optimise the management of pig farms, data collection and analysis is key. This use-case focuses on linking data across the value chain in order to provide the pig farmers with the necessary information to effectively implement and carry out their management activities. Figure 10 represents the current functional view of the solution being developed.

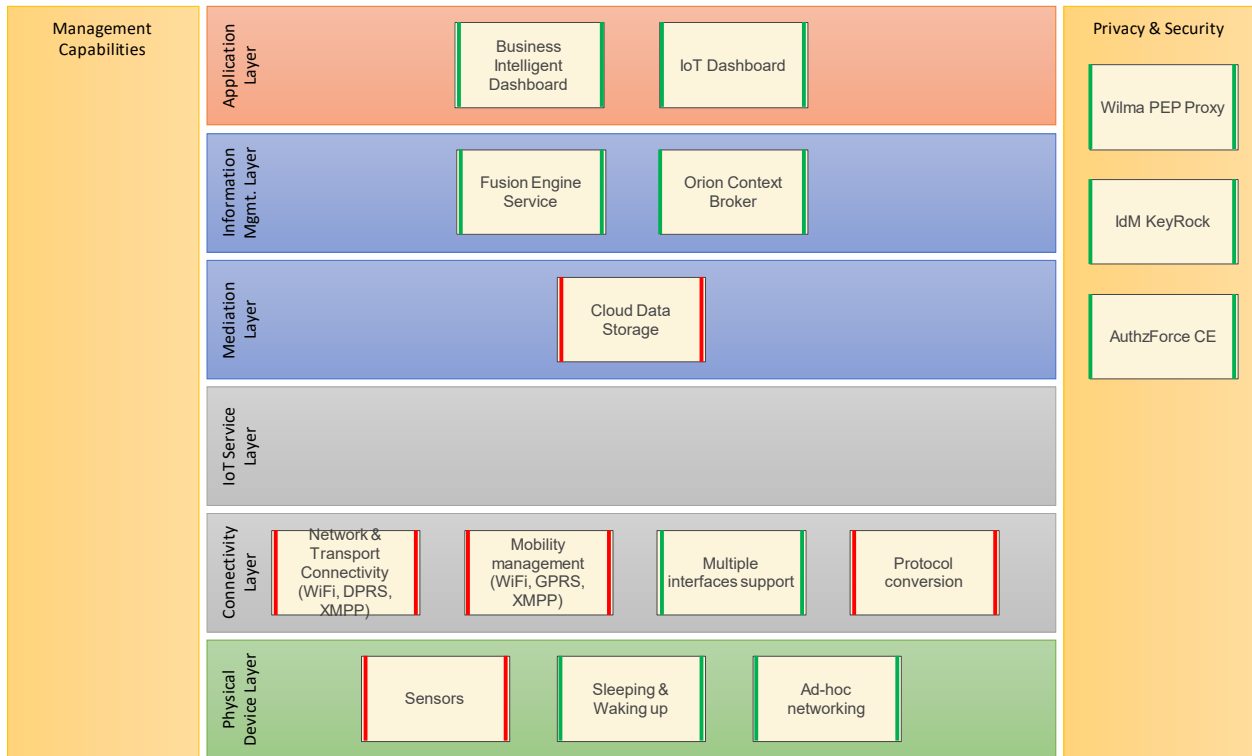


Figure 10: Functional architectural view of the UC5.1's solution.

This use case is developing an IoT platform connected to an Orion Context broker instance, for easy conversion, backup, and uniform availability of data. The Orion Context Broker was selected as a real-time data provider (as pub/sub system), enabling to expose rea-time data to external systems and infrastructures.

The use case faced some challenges to integrate the Orion Context Broker due to some instability issues caused by the under-development status of their architecture.

### 3.6 UC5.2 Poultry Chain Management

The main goal of this use case is to have an efficient growth and improvement of physical condition of poultry to obtain a desired and accurate end weight and physical condition required by the processing plant, respecting the animal welfare. This is done by linking different steps of the value chain (farm, logistics and processing), using sensor technologies and smart data analytics on big data platforms.

Figure 11 represents the current functional view of the solution being developed.

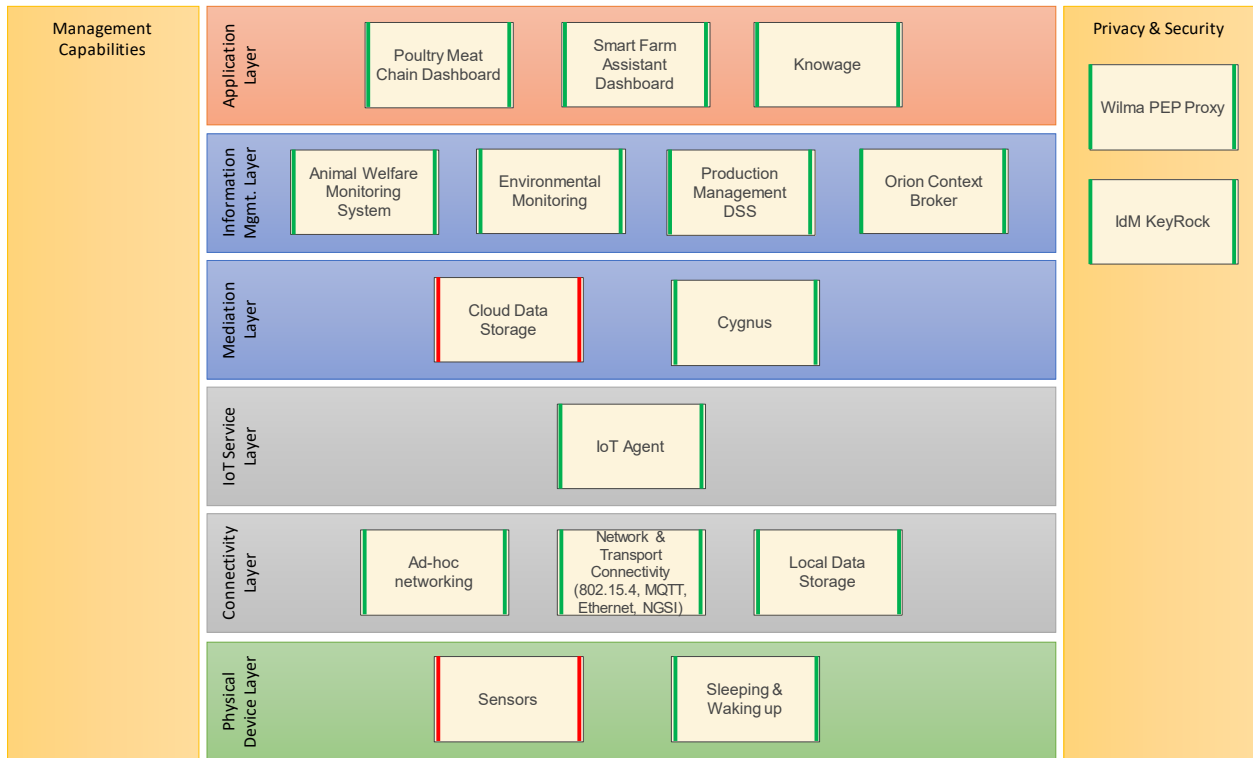


Figure 11: Functional architectural view of the UC5.2's solution.

In this use case, the Orion Context Broker provides the interface between two cloud platforms used, through a NGSI interface, to exchange needed information to perform specific services (data from farms, results of specific models from data analytics etc.). The Orion Context Broker enables the IoT platform to provide real-time data received from the farms.

### 3.7 UC5.3 Meat Transparency and Traceability

The objective of this use case is to develop, deploy, and demonstrate transparency system that enables tracking and tracing in the meat, particularly pork, sector. The use case doesn't deploy its own sensors and IoT platform but will instead collaborate either other UCs to gather transparency data and make it accessible.

Figure 12 represents the current functional view of the solution being developed.



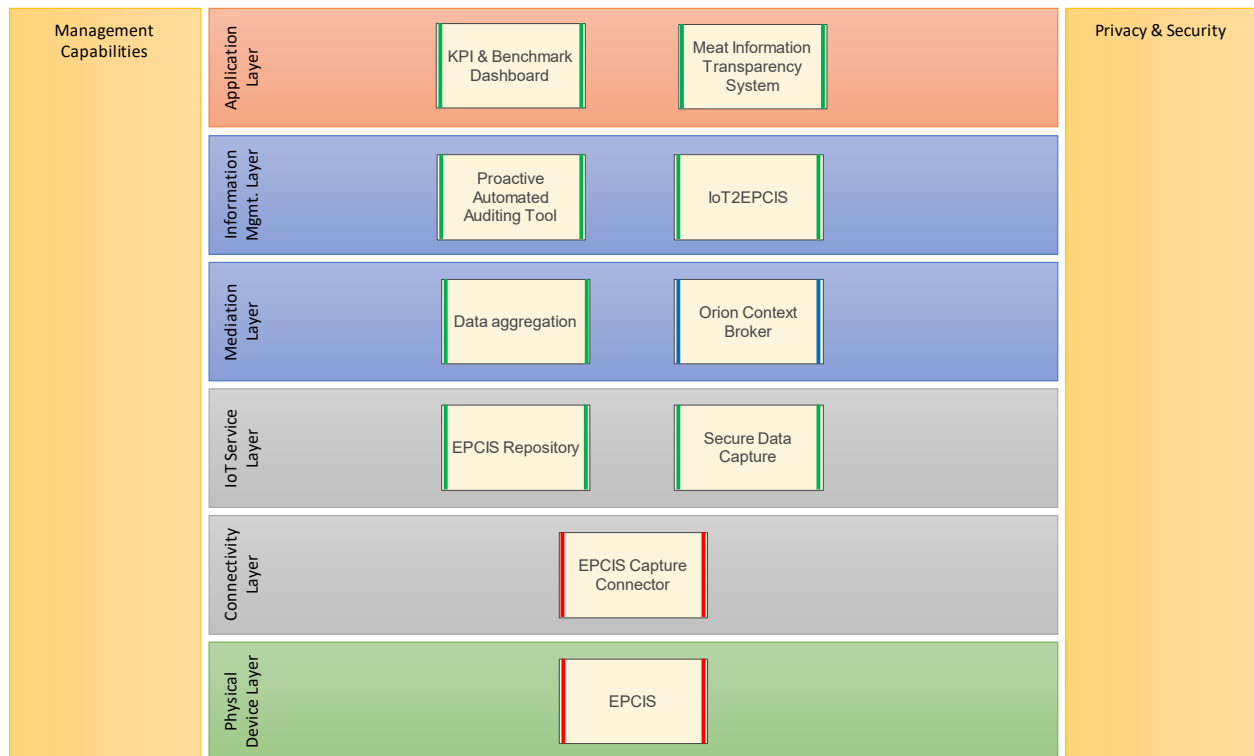


Figure 12: Functional architectural view of the UC5.3's solution.

In this use case, the Orion Context Broker the infrastructure to exchange data with two other solutions, from use cases 5.1 and 5.2. This use case has developed work to harmonise data representation in NGSI and EPCIS.

### 3.8 Summary

In this chapter, we presented the architectures of the six use cases already developing a solution powered by FIWARE. All the use cases are still progressing in the development of their IoT solutions. However, it is possible to draw some conclusions from these use cases:

1. The Orion Context Broker provides an infrastructure for data sharing, enabling connection of different systems, being sensors, equipment or other software solutions.
2. FIWARE offers a set of other generic enablers, e.g. Wilma PEP Proxy and IdM KeyRock, which support privacy and security features, increasing the robustness of the solutions developed.
3. The common use of the Orion Context Broker has already facilitated collaboration among use cases at several levels: developing and agreeing on data model fragments, and exchange of information.
  - a. Harmonisation of data model fragments in use cases of arable trial;
  - b. Exchange of climate station for inside conditions (between UC3.3 and UC4.2);
  - c. Reuse of dashboard to interact with Orion (between UC5.1 and UC5.2);
  - d. Harmonisation of data model and possible exchange of information (between UC5.1 and UC5.3).

These results already show the potential of the system of systems concept, further detailed in the next chapter.

## 4 System of Systems Approach

---

### 4.1 Overall concept and architecture

The figure below shows a picture of the envisaged architecture of integrated (system of systems) smart farm solutions, sharing a common (Context) Information Management Layer. At the bottom of the picture there are different Smart Agri-Food vertical solutions (**AgriApps**) devoted to solve specific problems and which, individually, may have been built using an architecture similar to the one described by **D3.3** and **D3.6**.

There is also the possibility that vertical solutions could have been developed using proprietary approaches, i.e. custom APIs and data models. In that case, an adaptor (named **NGSI Agent**) in the Mediation Layer would be needed. Such an adaptor serves as a bridge between harmonized Information Management APIs and data models of IoF2020 and vertical-specific, proprietary artefacts. This report shows how such an adaptor could be architected, and then implemented using Node-Red technologies.

The main layer in the referred picture is the **Context Information Management Layer**, enabled by NGSI-LD and Harmonized Data Models. It exposes (partially or totally) to the *Farm Management Information System (FMIS)* all the different Entities, Properties and Relationships that capture what is going on in the farm concerning the different verticals involved. Furthermore, each vertical could benefit from information generated by other integrated verticals, yielding to synergies that can bring new applications or insights to end users (farmers, agronomists, etc.). Historical data could be processed using different processing engines (e.g., Hadoop, Spark or Flink) to extract valuable insights or derive smart actions. Complex Event Processing, Advanced AI or machine learning functions can be implemented on top of integrated processing engines. In the end, end users will get access to an integrated view of farming processes from a single stop-point, the FMIS.

In addition, and in collaboration with other platform components (*Wirecloud, Knowage*), an FMIS could offer, globally, different integrated applications, both horizontal or vertical (from the different vertical providers) such as alert management, dashboards, advanced map representations, analytics (prescriptive, predictive or descriptive), KPI monitoring or data mash-ups.

In addition, vertical solutions could also export their data to a **Data Marketplace**, or even benefit from existing open data present in the marketplace. The open data available (through a CKAN data publication platform) could also be exploited directly by FMIS. Open data could also be made available through standard Geo-services, adapted and mashed-up as Context Information or being consumed directly by ad-hoc adaptors.

Last but not least, **API management** and business support functions enable auditing of the system and, potentially, monetization on data access.

The next sections describe the two main enablers of the System of Systems approach, the Context Broker and the Harmonize Data Models. They enable to build the Context Information Management Layer which is the gravitational centre of any System of Systems agri-food implementation.

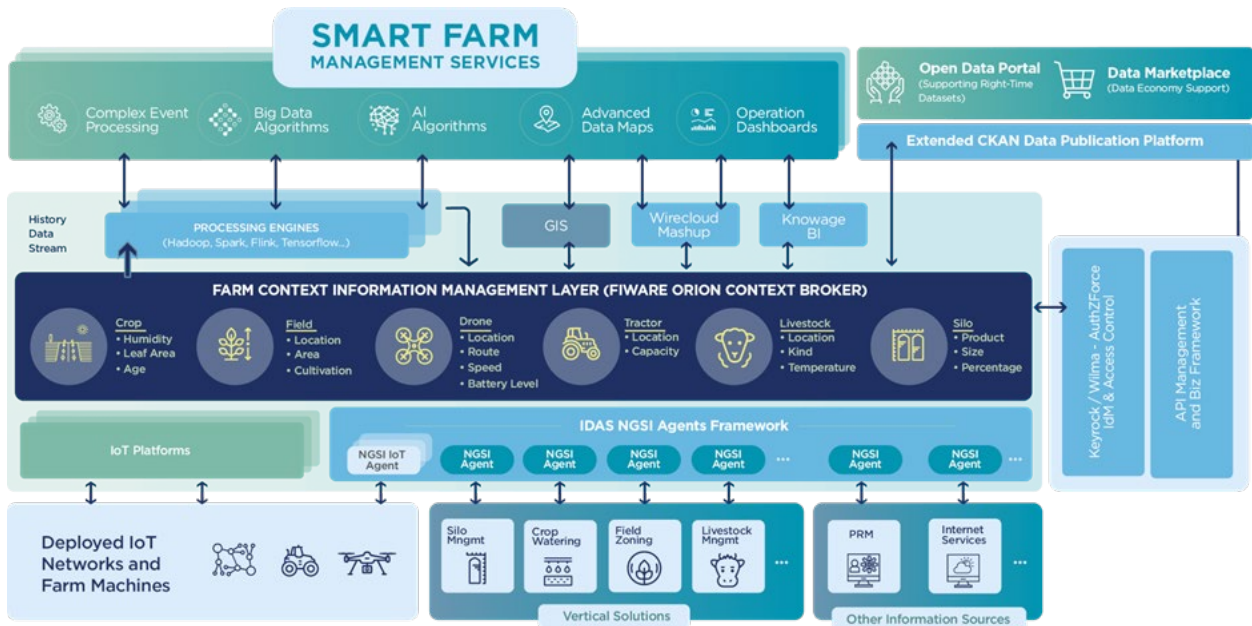


Figure 13: Systems of Systems approach using FIWARE technologies

## 4.2 Context Broker

The most popular implementation of the NGSi interfaces (NGSiV2<sup>1</sup> and NGSi-LD<sup>2</sup>) is the *Orion Context Broker*<sup>3</sup> which uses MongoDB as its underlying data store. The figure below shows an overview of the architecture and functional interfaces supported by this component. It can be deployed using Docker in the most popular platforms.

The Orion Context Broker allows to publish, consume and subscribe to data coming from multiple devices and data sources. In fact, it allows applications to get access to (harmonised) data entities, regardless data sources. The Context Broker may store data in the short to medium term using a data store. The expected uses of Orion are:

- Retention of current instances of harmonized data entities processed from IoT devices and external sources (context data);
- Storage of a window of short term historical harmonized data entities that may be queried directly via a third-party application.
- Storage of any Analytics and Intelligence results which become additional context data that can be queried or mashed up with other IoT data or external data sources.
- Register context provider (source) applications, e.g. a temperature sensor within a room.
- Update context information, e.g. send updates of temperature.
- Being notified when changes on context information take place (e.g. the temperature has changed) or with a given frequency (e.g. get the temperature each minute).
- Query context information. The Orion Context Broker stores context information updated from applications, so queries are resolved based on that information.

<sup>1</sup> <https://fiware.github.io/specifications/ngsiv2/stable/>

<sup>2</sup> [https://www.etsi.org/deliver/etsi\\_gs/CIM/001\\_099/009/01.01.01\\_60/gs\\_CIM009v010101p.pdf](https://www.etsi.org/deliver/etsi_gs/CIM/001_099/009/01.01.01_60/gs_CIM009v010101p.pdf)

<sup>3</sup> <https://github.com/telefonicaid/fiware-orion>

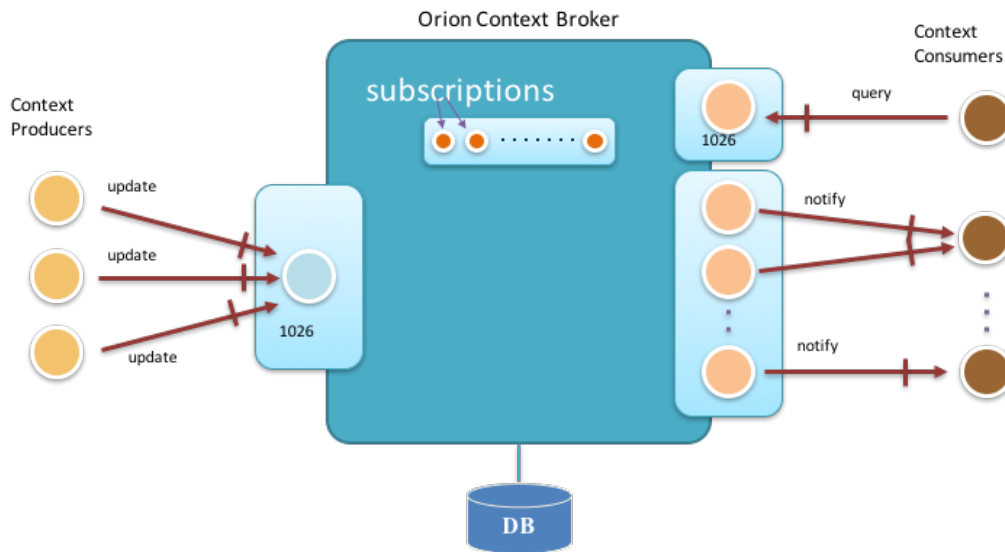


Figure 14: Orion Context Broker high level architecture.

### 4.3 Harmonized Data Models

The availability of shared, well-adopted information models is a key interoperability mechanism for enabling the System of Systems approach within a global market for IoT-enabled digital farming services. Such models provide an essential element in the common technical ground needed, by making replicability and portability of smart agri-food solutions practical.

In particular with regards to the System of Systems approach, it is needed to support and reuse open and standard data models and metadata by providing pre-built taxonomies to describe different assets. The final aim is to facilitate integrability of solutions across agri-food subdomains within common FMIS.

Thus, data models play a crucial role because they define the harmonised representation formats and semantics that will be used both by FMIS applications to consume data (through the northbound interfaces) and by data sources at the southbound (sensors, existing information systems or open databases) to publish data. Furthermore, data models are one of the key “minimal interoperability mechanisms”, enabling the participation in the System of Systems in agri-food. The implementation of the IoF2020 data models, together with NGSi interfaces ensure that each of the trials and use cases can deploy interoperable IoT-enabled digital farming services integrated within a System of Systems view.

More information about Data Modelling in NGSi and Data Models can be found at **D3.6**. They are largely based on the former work being done by the GSMA IoT Big Data Project. IoF2020 is actively contributing on the extension and enhancement of these data models to cover gaps, so that the needs of different trials and use cases are properly covered.

One of the main areas of enhancement developed, for the time being, has been around the Arable Domain. In this domain, and particularly in UC 1.4, there is an interest in offering a mapping between certain ADAPT concepts (or fragments of them) and NGSi-LD Entities. As a result, the communication of task execution from different machine vendor’s cloud and FMIS systems can be simplified and harmonized on a portable and multi-vendor configuration.

In addition, the Animal Data initiative developed by WP3 is intended to facilitate the participation of different stakeholders of the meat value chain within an IoF2020 enabled technology ecosystem. This initiative has benefited a lot from the collaboration of UC5.4 Decision Making Optimization in Beef Supply (Sharebeef), that has contributed with a Harmonized Data Model (see figure below) for representing the static and dynamic conditions of animals or cattle within an agri-food environment. It is noteworthy that this Data Model can also be reused by other trials; actually an interest has already been raised by the Dairy trial.



Data Model	Specification
Weather	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Weather-Observed.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Weather-Observed.md</a> <a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Weather-Forecast.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Weather-Forecast.md</a>
AgriParcel	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Parcel.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Parcel.md</a>
AgriParcelRecord	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Parcel-Record.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Parcel-Record.md</a>
AgriParcelOperation	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Parcel-Operation.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Parcel-Operation.md</a>
AgriGreenhouse	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Greenhouse.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Greenhouse.md</a>
AgriCrop	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Crop.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Crop.md</a>
AgriPest	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Pest.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Pest.md</a>
AgriSoil	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Soil.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Soil.md</a>
AgriProductType	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Product-Type.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Agri-Product-Type.md</a>
Device	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Device-Model.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Device-Model.md</a> <a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Device.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Device.md</a>
AgriFarm	<a href="https://github.com/FIWARE/dataModels/blob/master/specs/AgriFood/AgriFarm/schema.json">https://github.com/FIWARE/dataModels/blob/master/specs/AgriFood/AgriFarm/schema.json</a>
AgriApp	<a href="https://github.com/FIWARE/dataModels/blob/master/specs/AgriFood/AgriApp/schema.json">https://github.com/FIWARE/dataModels/blob/master/specs/AgriFood/AgriApp/schema.json</a>
Animal	<a href="https://github.com/FIWARE/dataModels/blob/master/specs/AgriFood/Animal/doc/spec.md">https://github.com/FIWARE/dataModels/blob/master/specs/AgriFood/Animal/doc/spec.md</a>

#### 4.4 Materialization of the SoS approach

The figure below shows a more detailed, closer to implementation, architecture diagram that will allow to materialize the System of Systems approach. The elements that take part in such an architecture are:

- **FMIS** which uses REST NGSI APIs to get data offered by the different integrated solutions, through the Context Broker (using the appropriate tokens for permissions, etc.).
- **Secured Context Broker**, an *Orion Context Broker* working together with a *Wilma* PEP proxy and a *KeyRock* instance for identity management. Such configuration has already been described by **D3.6**.
- **NGSI Agents**, which implement a dual interface: An NGSI interface (push or pull) to the Context Broker and the interface supported by the integrated Smart Agri-Food solution. The latter can be a proprietary interface, or it could already be an NGSI interface (following or not a harmonized representation of Entities). From the point of view of the Context Broker, NGSI Agents are considered as trusted elements in the architecture.



- **Smart Agri-Food solutions (AgriApps)**, these are the solutions that are integrated (the systems). These solutions might have been built using FIWARE or not.

Several Digital Farming Solutions (*AgriApp*) have to be integrated within an FMIS (see Figure 12, at the bottom). For each solution there should be an **NGSI Agent**, an adaptor capable of offering the relevant data of each solution through the NGSI interface, by following the corresponding harmonized Data Models adopted by IoF2020. For instance, if an Agri-Food application is devoted to managing the activity of cattle, all the information about cattle, subject to be managed by the FMIS, should be exposed through this interface, by using the NGSI API operations, and the **Animal** harmonized Data Model, presented before.

An NGSI Agent might work in two different modes (non-excluding between them):

- **Push mode.** In this mode, the NGSI Agent watches data changes in the origin *AgriApp* and propagate those data changes (once harmonized) to the Context Broker (using NGSI update operations). If the origin *AgriApp* supports the NGSI interface, this mode could be easily implemented through NGSI subscriptions. Otherwise, some kind of polling would be needed.
- **Pull mode.** In this mode, the NGSI Agent receives forwarded requests from the Context Broker concerning data managed by the corresponding *AgriApp*. The NGSI Agent shall translate such data request to the proper request to the *AgriApp* (including the proper security credentials). Once the data is got and harmonized, it is returned to the Context Broker and finally to the FMIS.

Each *AgriApp* should register in two different places:

- with the FMIS, indicating what kind of Entities is able to provide, and possibly the endpoint of its customized user interface, for instance, an HTTP endpoint for a Web application.
- with the Context Broker, indicating what is the NGSI Agent capable of offering the concerned Entities in the appropriate harmonized Data Model.

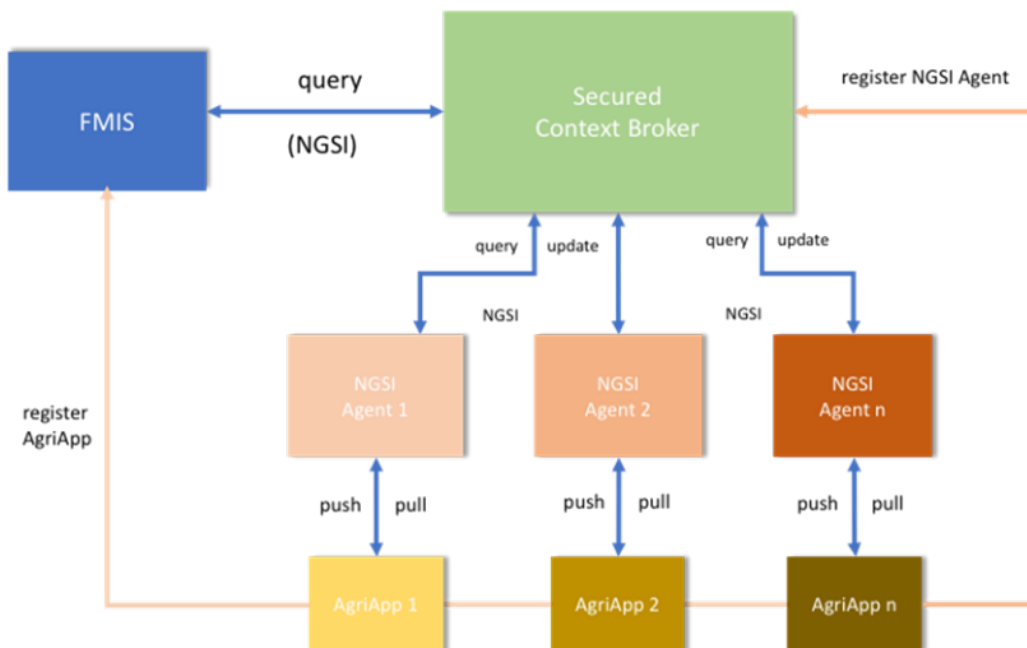


Figure 16: Materialization of the System of Systems approach using Context Broker.

The figures below show sequence diagrams of the interactions happening in both modes pull mode and push mode. The diagram also covers the interactions happening at configuration time (in a different colour).

In the pull mode (see below), NGSI Agents register with the Context Broker as providers of certain kind of Entities, probably in a certain geographical area covered by the corresponding *AgriApp*. Concerning data retrieval, FMIS initiates it through an NGSI query operation, the Context Broker resolves by forwarding the request against the corresponding NGSI Agent that knows how to resolve the query by gathering the data from the corresponding *AgriApp*.

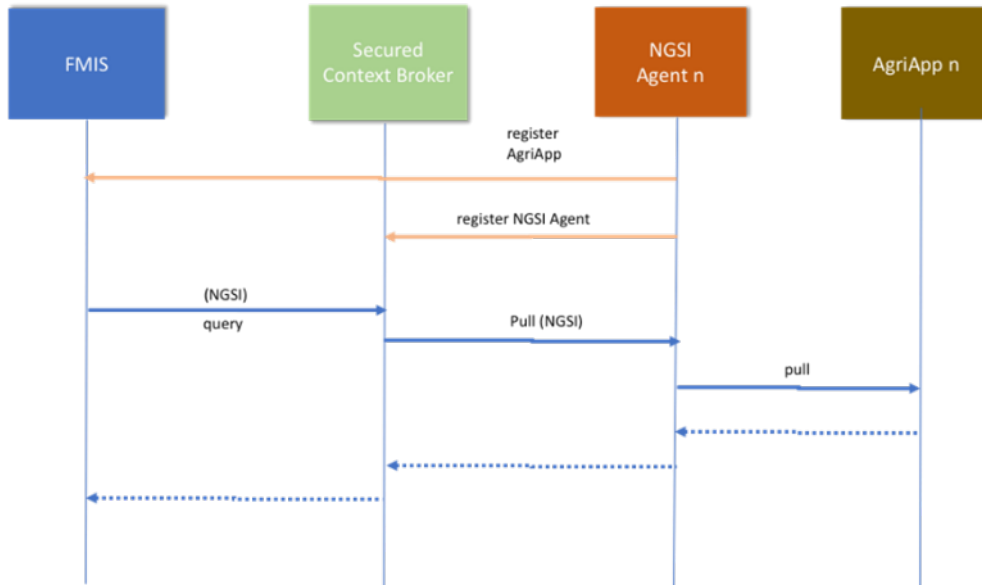


Figure 17: Materialization of the System of Systems approach (pull mode).

In the push mode (see below), NGSI Agents watch data changes happening in the origin AgriApp. Once they receive data changes from the AgriApp, those data changes are propagated to the Context Broker, which stores them in its local storage. As a result, when the FMIS initiates an NGSI query, the Context Broker directly resolves the query without contacting the NGSI Agent.

It is noteworthy that there can be deployments where both modes (push and pull) are used at the same time. Last but not least, readers should also consider the security implications in this architecture. Although, the data being accessed by the FMIS is properly secured, there should be mechanisms to protect such data depending on the user roles, so that users only get access to the data of the Agri applications they are concerned with.

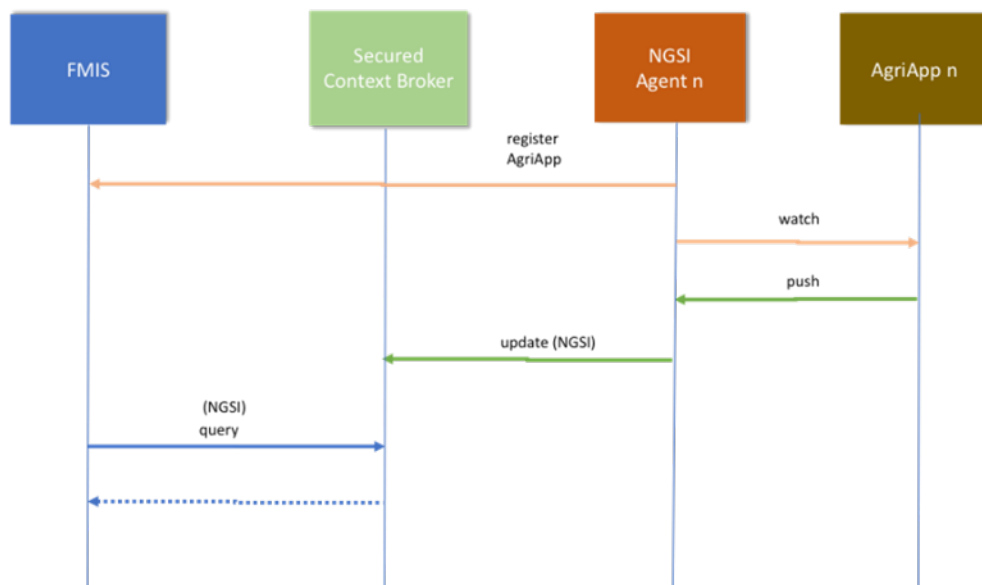


Figure 18: Materialization of the System of Systems approach (push mode).

#### 4.5 Node-Red enabler for the SoS approach

**Node-Red** is a visual programming tool that allows to create data flows for IoT-enabled applications. Node-Red is a really popular tool among the developer’s community, as it enables the rapid development of data applications that integrate different IoT protocols and data sources. The number of Node-Red extensions available is around 3000 and growing every day.



Node-Red is based on the concept of Nodes, which are data processing and interfacing elements that can be connected between them. A Node has zero or more inputs and generates zero or more outputs (to be consumed by other Nodes).

Node-Red provides some core Nodes and at the same time third parties can develop their own nodes to be integrated within the Node-Red ecosystem. A Node in Node-Red defines a user interface configuration part together with an execution part. Both parts are developed using Web technologies, HTML and Javascript. The UI part is executed on a Web Browser while the runtime execution part is run against a backend based on Node.js.

FIWARE Foundation, in the scope of the IoF2020 project, has developed a **Node-Red extension package**<sup>4</sup> that makes it easier for developers to integrate data from IoT and other data sources within the Orion Context Broker. That package is of particular interest to the System of Systems approach as it allows the rapid prototyping of the integration of Agri-food solutions within a System of Systems. The following sections describe the different Nodes available and their configuration. Finally, an example of Systems of Systems built using Node-Red is provided.

#### 4.5.1 Node-Red FIWARE Extension Overview

The Node-Red FIWARE Extension, described in detail in Appendix A: Node-Red FIWARE Extension Nodes, at the end of this document, includes the following Nodes (see figure below):

- *Context Broker*. A Configuration Node to set up the endpoint and security parameters of a FIWARE Context Broker (exporting NGSIv2 or NGSI-LD).
- *NGSI Entity*. Provides the JSON representation (both NGSIv2 and NGSI-LD) of an Entity queried by its identifier.
- *NGSI Dataset*. Allows to define an NGSI (LD, v2) dataset by providing the corresponding filtering and projection conditions.
- *NGSI Update*. Allows to update (in *upsert* or *update* mode) one or more NGSI (LD,v2) Entities.
- *NGSI Subscription*. Allows to subscribe to NGSI (LD, v2) Entities and Attributes enabling the propagation of NGSI (LD, v2) notification data to a Node-Red flow.
- *NGSIv2ToNGSI-LD*. A Node that allows to transform NGSIv2 representations to NGSI-LD.

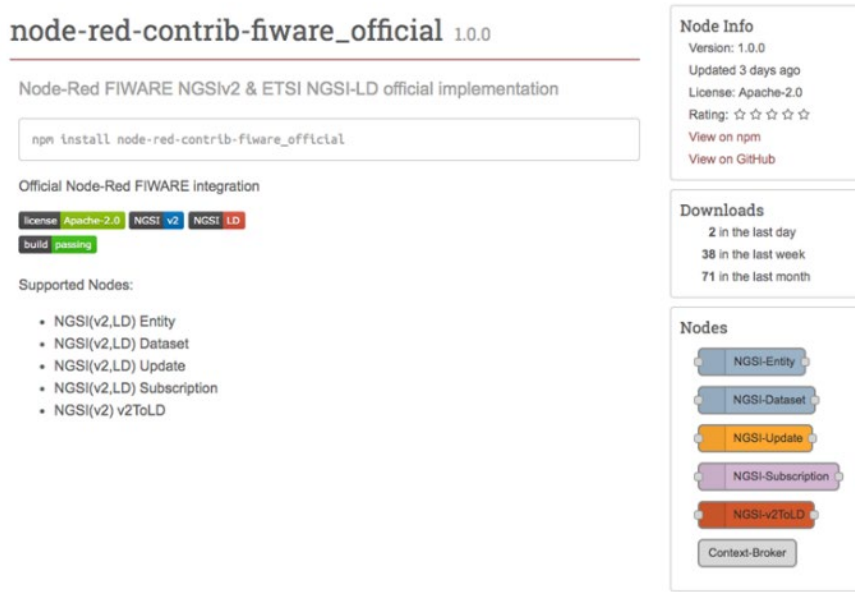


Figure 19: Node-Red FIWARE Extension Package Overview

<sup>4</sup> [https://flows.nodered.org/node/node-red-contrib-fiware\\_official](https://flows.nodered.org/node/node-red-contrib-fiware_official)

## 4.6 SoS case of study implemented using Node-Red

The following example shows how the System of Systems approach can be implemented by using a Node-Red visual programming flow that harnesses all the potential of the Node-Red package described in the previous section.

The assumptions of the case of study are the following:

- There are three Agri applications (AgriApp 1, AgriApp 2, AgriApp 3) to be integrated into a System of Systems and the data interfaces supported by each application are NGSI v2, MQTT and HTTP REST respectively.
- From the FMIS point of view only query operations are allowed, therefore data changes always occur at the level of the original AgriApp.

The figure below shows the case of study architecture, which is based on the *pull mode*. The NGSI Agents are in charge of harvesting all the data from the original Agri Apps and to consolidate it within a Context Broker that serves as unified data interface for the FMIS. During this process, typically, some data harmonization will happen, as the original applications might not be using the Harmonized Data Models, as expected by the FMIS queries.

The harvesting process can be done by issuing periodic queries to *Agri Apps*, as in the case of *Agri App 3*, which only supports a REST HTTP interface, or via a subscription-notification mechanism, as it happens with *Agri App 1* (NGSIv2) and *Agri App 2* (MQTT).

The NGSI Agents are implemented within Node-Red, as Node-Red Nodes connected through a flow.

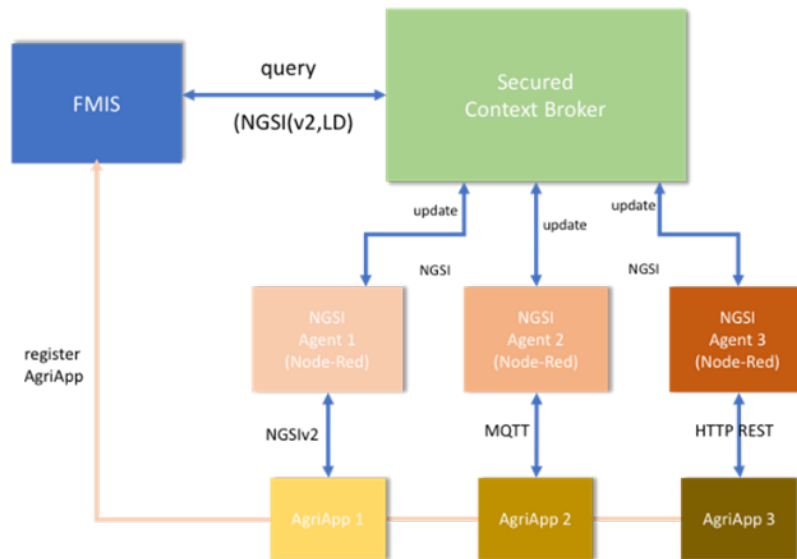


Figure 20: Case of Study for the SoS approach (push mode).

The figure below shows the Node-Red flow that allows to implement the **NGSI Agent 1**, which serves as an adaptor between *AgriApp 1* (based on NGSIv2) and the SoS. It is composed by the following Nodes:

- *Flow.start*. Inject Node. Allows to bootstrap the NGSI Agent but setting up a NGSIv2 Subscription against the corresponding Context Broker endpoint of *AgriApp 1*.
- *AgriApp 1 Subscription*. NGSI Subscription Node. Performs a Subscription that will encompass the Entities of interest from the point of view of the FMIS, i.e. it is not necessary to subscribe to the whole set of Entities published by *AgriApp 1*.
- *NGSI Agent 1 Notify*: HTTP input Node. Allows to receive NGSIv2 notifications from *AgriApp 1*.
- *Data Harmonization*. Function Node. It implements all the Data Harmonization needed before consolidating data into the FMIS Context Broker.
- *FMIS CB Update*. NGSI Update Node. It is in charge of persisting all the data provided by *AgriApp 1* to the FMIS Context Broker.

- *Debug*. Debug Node. An auxiliary Node to debug and trace the NGSI Agent 1 operations.

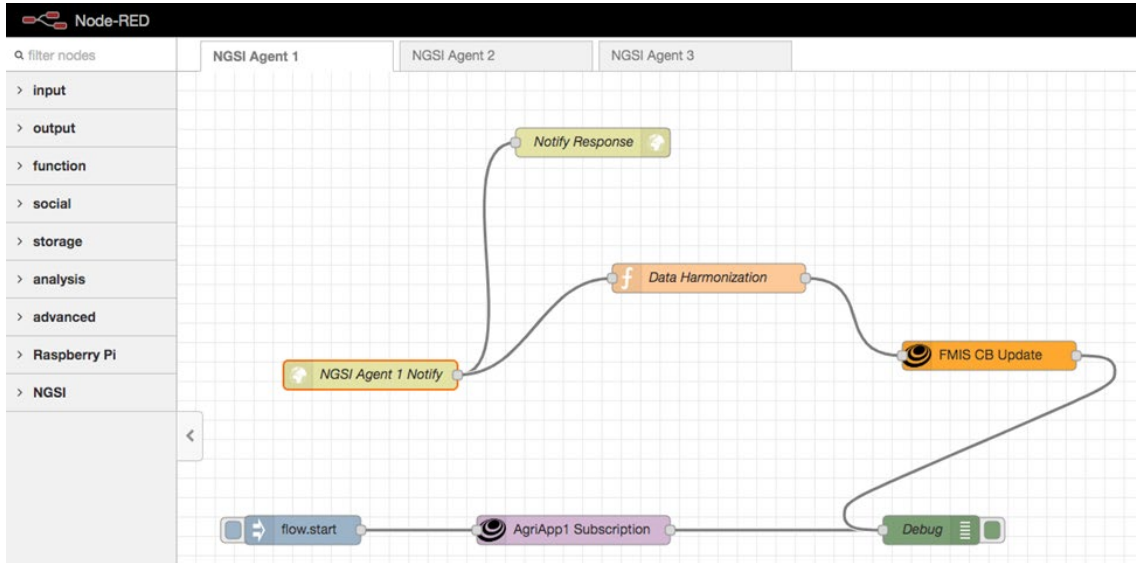


Figure 21: Node-Red flow that realizes NGSI Agent 1 (NGSIV2)

The figure below shows the Node-Red flow that allows to implement the **NGSI Agent 2**, which serves as an adaptor between *Agri App 2* (based on MQTT) and the SoS. It is composed by the following Nodes:

- *AgriApp2 MQTT Topic*. MQTT In Node. Subscribes to an MQTT topic that will provide the *Agri App 2* data of interest from the point of view of the FMIS.
- *Data Harmonization*. Function Node. It implements all the Data Harmonization needed before consolidating data into the FMIS Context Broker.
- *FMIS CB Update*. NGSI Update Node. It is in charge of persisting all the data provided by *AgriApp 2* to the FMIS Context Broker.
- *Debug*. Debug Node. An auxiliary Node to debug and trace the NGSI Agent 2.

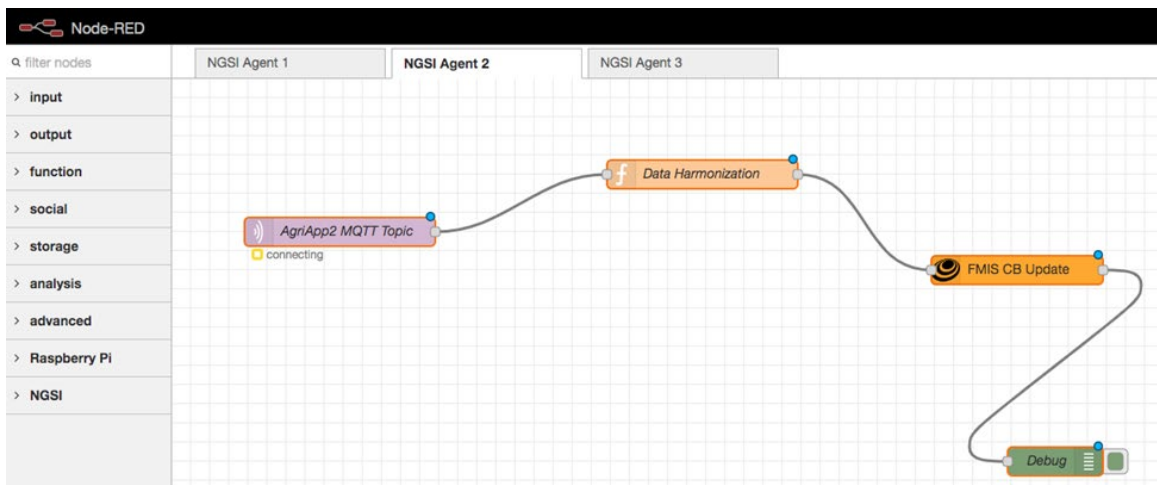


Figure 22: Node-Red flow that realizes NGSI Agent 2 (MQTT)

The figure below shows the Node-Red flow that allows to implement the **NGSI Agent 3**, which serves as an adaptor between *Agri App 3* (based on HTTP REST) and the SoS. It is composed by the following Nodes:

- *Periodic Harvesting*. Inject Node. It triggers the data harvesting process periodically.
- *HTTP REST Harvester*. HTTP Request Node. Performs the data harvesting by issuing HTTP REST requests against the endpoint of *AgriApp 3*.

- *Data Harmonization*. Function Node. It implements all the Data Harmonization needed before consolidating data into the FMIS Context Broker.
- *FMIS CB Update*. NGSi Update Node. It is in charge of persisting all the data provided by *AgriApp 3* to the FMIS Context Broker.
- *Debug*. Debug Node. An auxiliary Node to debug and trace the NGSi Agent 3.

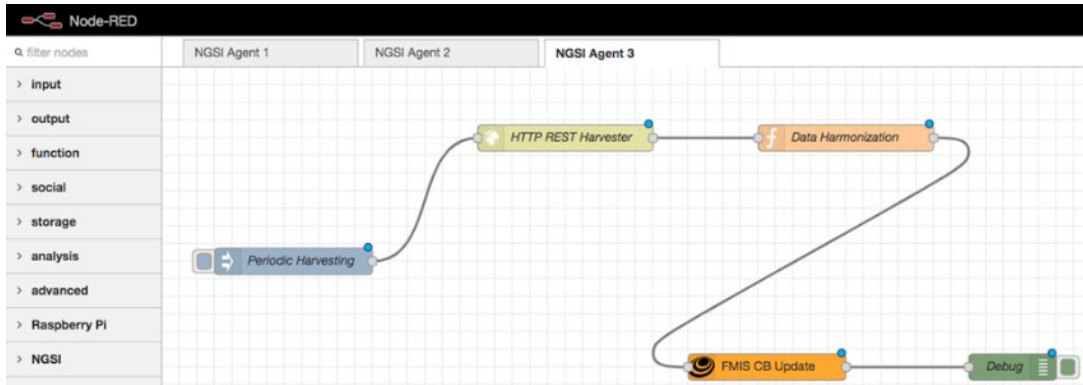


Figure 23: Node-Red flow that realizes NGSi Agent 3 (HTTP REST)

#### 4.7 Weather Station data integration in a SoS approach

Due to the ongoing discussion of the System of Systems approach as a powerful development mindset to “push and pull” new technologies and innovative solutions within the agro production, the purpose of this section is to provide an initial description of a FIWARE Domain Specific Enabler (DSE) for weather in the context of the FIWARE Agri-Food Reference Architecture<sup>5</sup>.

The main goal is that this DSE shall enable a standardized connection to weather sensors and agrometeorological services as part of core functionalities powered by a FMIS. This idea will be possible relying on Orion Context Broker<sup>6</sup> technology and the NGSi-LD<sup>7</sup> standard.

In order to give guidance on the importance of weather systems for the Agri-Food SoS, it can be considered, that the regional weather variations are a breakpoint in the agricultural production, more than in any other branch of production, for example temperature and precipitation acquire a restrictive and limiting dimension, in short words: the success of harvest and livestock depends on how the farmer reacts to the sudden weather changes and how he optimizes decisions by knowing weather trends of his region. The end user is subject to planning practically all of the production activities, according to the current climate information.

A good solution should provide a properly weather agronomic decision support service, which fulfils microclimate and macroclimate analytics. This is possible with the increase of the offer of weather stations based on IoT architectures, the optimization of satellite networks, more precise aerial images with drones or any other technique of image acquisition and more accurate proximal sensing. At this point, it seems appropriate to mention that the current work has some major interests:

- Providing a standardized interface to several producers that can be used by FMIS platforms and other Agri-Food related products
- Providing a standalone system to enable end users (farmers) to visualize data coming from different sensors
- Providing a product that can be used even by the government (local and central) to connect many different sensors and services in order to provide to optimized agronomic advice related to climate adaptation and disease prevention

<sup>5</sup> <https://www.fiware.org/community/smart-agrifood/>

<sup>6</sup> <https://fiware-orion.readthedocs.io/en/master/>

<sup>7</sup> [https://fiware-datamodels.readthedocs.io/en/latest/ngsi-ld\\_faq/index.html](https://fiware-datamodels.readthedocs.io/en/latest/ngsi-ld_faq/index.html)

### 4.7.1 Development of a weather system

All the evaluations about profitability and feasibility of a weather system will be irrelevant if those are not implemented in a solution. In this case, the capability to build a weather system presents an opportunity to grow business models, accessing to data sources that are dynamically linked through reusable and flexible components and to optimize the gathering of high quality weather data. In a first market analysis developed, and to the best of our knowledge, this kind of solution is missing and, moreover, is more than welcomed by public administrations and by Agri-Food actors to simplify the distribution and access to this kind of data.

Within the framework of the IoF2020 project, Agriculus and 365FarmNet are interested, in collaboration with WP3, to promote activities to demonstrate the implementation of an agrometeorological system, which could be integrated as part of these two regional FMISs. WP3 as the technological mediator of the IoF2020 project, together with FIWARE and ATB, aims to strengthen the use and adaptation of the resources and components, “Powered by FIWARE”, and exposed in this document, while Agriculus and 365FarmNet have recognized as a common benefit to run a joint effort to participate in this initiative using FIWARE components for this kind of development.

Table 4 summarizes the activities around the initiative and presents a proposal for the further tasks, in order to achieve exchange of open data and promote the data transformation within a sustainable Agri-Food SoS.

Table 4: Activities to develop a weather system.

Initiative	AgroWeatherGateway DSE
<b>Participants</b>	Agriculus (UC 5.4) and 365FarmNet (UC 1.4)
<b>Moderators</b>	ATB Bremen; FIWARE Foundation (WP3)
<b>Synergies</b>	<i>Aligned with the UC-specific goals and SoS approach:</i>
Weather – Interoperability and Harmonization	As partner of the UC1.4 365FarmNet is investigating the best way to achieve better models and vocabularies within the Agri-Food.
Weather Crop Production	Participants are FMIS and weather data providers, which can optimize climate related Alerts regarding diseases or pest-risks
<b>Activities</b>	<i>Aligned with SoS approach</i>
Discovery Synergies	Participants and mediators contribute in the discussion, how the weather affects his own domains and how the collaboration could take place to define a Proof of Concept for the initiative
Data Model Harmonization	Bringing together data models regarding weather station data, investigate the current FIWARE Data Models, extend the data models (if necessary) and transforming existing entities into one cohesive data set.
Weather-API	Investigate and define REST API for the services. Focus: feasibility of development based on NGSI-LD API  Technological Support: WP3
Context Broker implementation	As a fundamental piece of “Powered by FIWARE” solutions, the participants will explore the integration of the Context Broker in its own solutions

Initiative	AgroWeatherGateway DSE
Services Connection	PoC of Weather System.

#### 4.7.2 Architecture

Based on the general architecture for Agri-Food solutions provided by FIWARE, Agricolus and 365FarmNet will explore the development of a Weather-DSE considering the architecture proposed in Figure 24.

This Gateway is both a standalone application and a data gateway for agro-weather data:

- The **standalone application**: is provided to deploy a monitoring and data collection system, which for government, association of farmers and other organizations collects, visualizes and delivers weather data to third parties. With this DSE, each system will be able to setup an environment capable of implement several communication standards regarding weather data for several producers with a single point of access.
- The **data gateway**: each agent will be able to deliver, upon authorization of each entity, these dynamic data to third parties, such as AG platforms (FMIS) and mobile apps, to enable quickly and with a reduced cost new services and functionalities.

This DSE includes both the web client access for single users than the APIs interfaces to retrieved data.

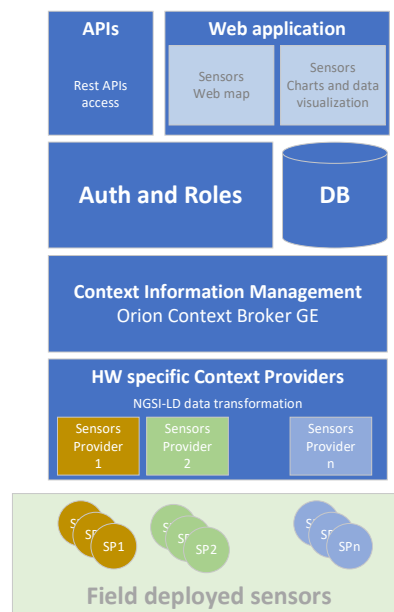


Figure 24: Proposed architecture of a Weather information System

**The DSE main development is dedicated to these areas:**

- **Context Providers**: will interact with different producers' ecosystems and cloud applications. Each producer may allow the access to a cloud repository or directly to the sensors.
  - The context provider will manage the communication layers, the authentication of the weather stations, sensors specific configurations and other hardware-related data (such as alarm notifications in case of hardware failure, if provided)
- **Authentication and Roles Management**: a simple authentication with an admin role for admin-users and access to specific group of sensors (each sensor will be included in a group of sensors)
- **API access for external applications**: upon specific authorizations third parties will be enabled to access real time data form the DSE with given credentials
- **Web interface**: including features such as sensors management and configuration, weather stations map, weather stations charts and data etc.



### 4.7.3 Harmonization of Weather Data

With an infinity of devices and sensors, which are constantly recording significant variables for the agriculture domain such as temperature, relative humidity, solar radiation, precipitation and wind speed among others, the current generation of weather stations is developed with special focus on the optimized energy consumption and more efficient communication protocols, mainly due to the increase connected devices as well as the complexity and amount of collected data.

The standardization initiatives regarding to the physical layers of a weather station and their connections with the data layer as well as the communication layers, are more advanced and haven a common acceptance, however there is still not consensus about how a solution shall be harmonized for a continuous delivery of value within a weather system, but most important how a weather solution may obtain feedback from the network and pro-vide the required accuracy in a DSS.

It is the goal of the SoS approach to facilitate consolidation and reuse of the different kind of data and information sources in a common framework. The information is provided from solutions, which may have only partial overlap of their data models. Possible conflicts may occur between solutions or even within the same domain of each solution if the description of data (meaning, provenance, etc) is ambiguous, changes over the time, or changes according to the responsible for the sourcing of information. Those reasons make evident our need of use of a common ground for the further development.

As described in section 4.3 and following the ideal steps to achieve a common framework, the first line of action is the harmonization of the weather data model. A significant part of the first approach should be addressed to the descriptions that FIWARE has documented in its Weather Data Model (<https://fiware-datamodels.readthedocs.io/en/latest/Weather/WeatherForecast/doc/spec/index.html>). These data models have been developed in cooperation with others organizations, but they are very close of the required basic entities in the Agri-food domain and are defined based on a common ontology. A more important point is that the model is open source and can be extended following the guidelines that are also presented in the documentation, which makes it even more attractive to the development community.

Table 5: Weather data models.

Weather Data Model	Specification	Comments
WeatherStation (POI)	<a href="https://github.com/FIWARE/dataModels/tree/master/specs/PointOfInterest">https://github.com/FIWARE/dataModels/tree/master/specs/PointOfInterest</a>	A weather station can be considered a Point of Interest. <i>WeatherStation</i> can be defined using the <i>Pol</i> Entity
WeatherObserved	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Weather-Observed.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Weather-Observed.md</a> Agri-Food Specific Data: <ul style="list-style-type: none"> <li>• weatherType</li> <li>• dewpoint</li> <li>• temperature</li> <li>• precipitation</li> <li>• relativeHumidity</li> <li>• solarRadiation</li> </ul> evapotranspiration	Weather Observations can be provided by a weather service local or regional.
WeatherForecast	<a href="https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Weather-Forecast.md">https://github.com/GSMADeveloper/NGSI-LD-Entities/blob/master/definitions/Weather-Forecast.md</a> Agri-Food Specific Data: <ul style="list-style-type: none"> <li>• weatherType</li> <li>• validFrom</li> <li>• validTo</li> </ul>	The weather forecast are regional and local. Each partner shall check the data licenses at the data source.

Weather Data Model	Specification	Comments
WeatherAlert	<p><a href="https://github.com/FIWARE/dataModels/blob/master/specs/Alert/doc/spec.md">https://github.com/FIWARE/dataModels/blob/master/specs/Alert/doc/spec.md</a></p> <p>Agri-Food Specific Data:</p> <ul style="list-style-type: none"> <li>• <b>category:</b> weather and agriculture</li> <li>• <b>subCategory:</b> (rain-fall, highTemperature, low-Temperature, heat-Wave, coldWave, ice, snow, wind, fog, tornado, tropicalCyclone, hurricane, snow/ice, thunderstorms, fireRisk, avalancheRisk, floodRisk) (for weather category)</li> <li>• <b>subCategory:</b> (noxiousWeed, snail, insect, rodent, bacteria, microbe, fungus, mite, virus, nematodes, irrigation, fertilisation) (for agriculture category)</li> <li>• <b>Severity:</b> low, medium, high, critical</li> </ul>	WeatherAlert are generated by a specific situation regarding to the <i>AgriCrop</i> and <i>AgriParcel</i> , depending on <i>WeatherObserved</i> and <i>WeatherForecast</i> .

An example of JSON-Schema for *WeatherObserved* on Agri-Food Domain is displayed below (Source: <https://github.com/FIWARE/dataModels/tree/master/specs/Agri-Food>):

```
[
  {
    "address": {
      "addressCountry": "ES",
      "addressLocality": "Barcelona"
    },
    "dataProvider": "FIWARE",
    "dateObserved": "2019-05-27T21:00:00.00Z",
    "id": "Spain-WeatherObserved-0201D-latest",
    "location": {
      "coordinates": [2.2, 41.3906],
      "type": "Point"
    },
    "precipitation": 0,
    "relativeHumidity": 90,
    "source": "http://www.aemet.es",
    "stationCode": "0201D",
    "stationName": "Barcelona",
    "temperature": 17,
    "type": "WeatherObserved",
    "windDirection": 105,
    "windSpeed": 3.5
  }
]
```



The following figure shows a diagram representing a first approach of a property graph, which could be used future implementations discussed below in this section. The graph was drawn following the example presented in annex C: clause C.2 Entity Representation of the ETSI-Specification Context Information Management (CIM); NGSI-LD API ([https://www.etsi.org/deliver/etsi\\_gs/CIM/001\\_099/009/01.01.01\\_60/gs\\_CIM009v010101p.pdf](https://www.etsi.org/deliver/etsi_gs/CIM/001_099/009/01.01.01_60/gs_CIM009v010101p.pdf)) and its intention is showing a possible model within a weather system that interacts with other defined Agri-Food Entities.

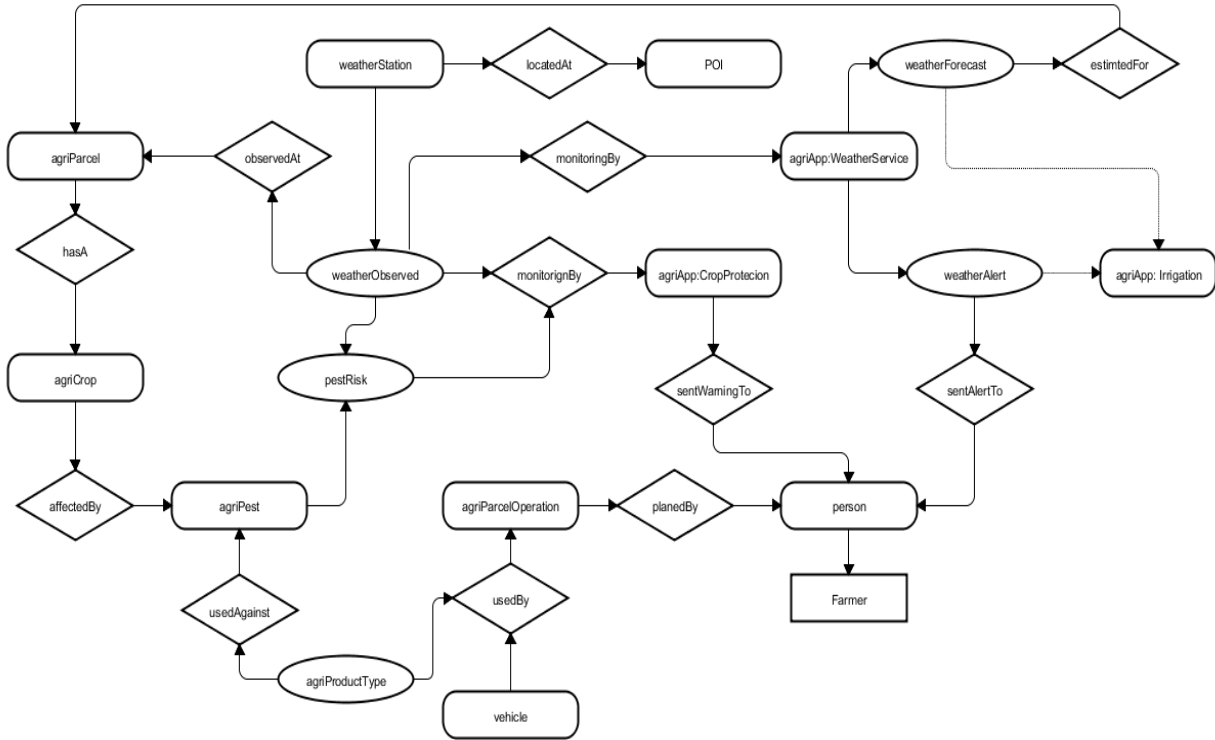


Figure 25: Example of Property Graph for Weather Stations in Agri-Food (Work in progress)

#### 4.7.4 Synergies of Weather Systems

Finally, in order to exploit the potential of weather system, the project participants need to partner with local and international providers of weather data. These providers must at least offer frequent and daily updates of agrometeorological variables, but if the partners would like to add value for their standalone products, the value proposition for end users can be improved for example by linking forecast directly to a FMIS, which can send alerts to react promptly to unexpected weather conditions. The FMIS can suggest the user to plan machines or to apply products, based on the weather observations and each integrated service within the system. This describes in a general way what is known as synergies between systems: separate solutions that when integrated can enhance their value as services in the Agri-Food domain.

The following Table 6 shows some of the identified synergies between a weather system and other systems in the SoS approach, as important topic for further development.

Table 6: Synergies between weather and other systems.

Partner System Issue	Weather System Solution	End User profit
<b>Machine-Equipment System</b> A. Risk of compaction B. Loss of soil structure C. Pesticide Application: Drift Risk D. Fertilizer Application: Groundwater Risk	Control of Temperature, Precipitation, wind, <ul style="list-style-type: none"> <li>Alerts and notifications</li> <li>Forecast microclimate</li> </ul>	Planning the output of products, just in time Reduce the over-application of fertilizers Saving Fuel, Saving Resources
<b>Seed Optimizing System</b> A. Market trends B. Prevention of diseases C. Biomass development	Control of temperature, pressure, relative humidity <ul style="list-style-type: none"> <li>Disease prognoses,</li> <li>Analysis of historical data</li> <li>Optimized the crop season</li> </ul>	Time-windows for seeding, optimal date. Reduce the risk of diseases, choosing another crop or crop variety Increase the crop-rotation Reduce Pesticides Inputs on the soil
<b>Irrigation System</b> A. Risk of drought B. Risk of freezing C. Groundwater control D. Soil nitrification	Control of Solar Radiation, Temperature, precipitation, dew point, relative Humidity <ul style="list-style-type: none"> <li>Alerts and notifications</li> <li>Forecast for the season and day-to-day</li> </ul>	What can I plant for the next season, which variety is more resistant to drought, flooding, etc.

## 4.8 Service Monetization – CoatRack

### 4.8.1 Key Features offered by CoatRack

Use cases and specifically the providers of IoT based solutions can potentially extend their business models towards third parties by offering their individual services also to a larger stakeholder audience. With an attempt to further monetize services they could provide own services, while also taking the advantage to consume services that are offered by third parties. The IoF2020 partner organisations ATB and CORIZON were joining forces to develop the so called “CoatRack” open component, enabling service monetization and more specifically a framework to

- Monetize services with the possibility to offer and consume software services in N-to-M relationships of service providers and consumers.
- Visualising the access to services to enable an understanding of users’ interest.
- Share ICT (infrastructure/ overhead) costs inside an organisation, while this is true for using internal as well as external services.

In general, use cases and any other external party can use CoatRack to offer software service APIs to developers via a trusted framework for service monetization and access control. As presented in the following schematic Figure 26, CoatRack facilitates the service provision and access control as well as taking care for service monetization. The typical users of CoatRack are software developers that are offering and/or consuming software services as basis to offer additional applications for specific end-users.

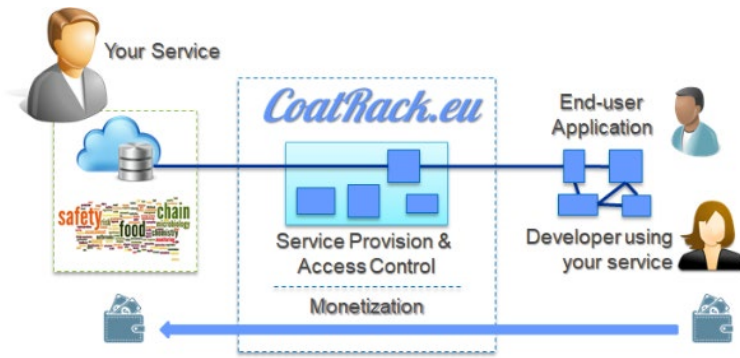


Figure 26: Schematic interaction of service providers and consumers, facilitated by CoatRack.

#### 4.8.2 CoatRack Hosting Environment

CoatRack is realised as a central service enabling and supporting the distributed offering and usage of software services. For that purpose, a central CoatRack instance is offered by ATB and CORIZON. This central instance is taking care for all administrative features that are facilitating access tracking, visualisation and monetization. To realise the N-to-M relationships of service providers and service consumer, the service provider is using CoatRack gateways that are generated for each individual service offering. This approach is also enabling CoatRack users to do both providing and consuming services at the same time.

As presented in the following Figure 27, the gateway is downloaded by the service provider and installed in the local server that is also hosting the service API that shall be exposed to third parties via the CoatRack gateway. By using the central instance of CoatRack, the service provider can register all required details of its service in the CoatRack front-end. As soon as this is done, the service provider can download a CoatRack gateway that is individually created for this specific service. Also changes in the service configuration can be done in the central CoatRack front-end that is also updating the configuration of the CoatRack gateway, already installed in the local environment.

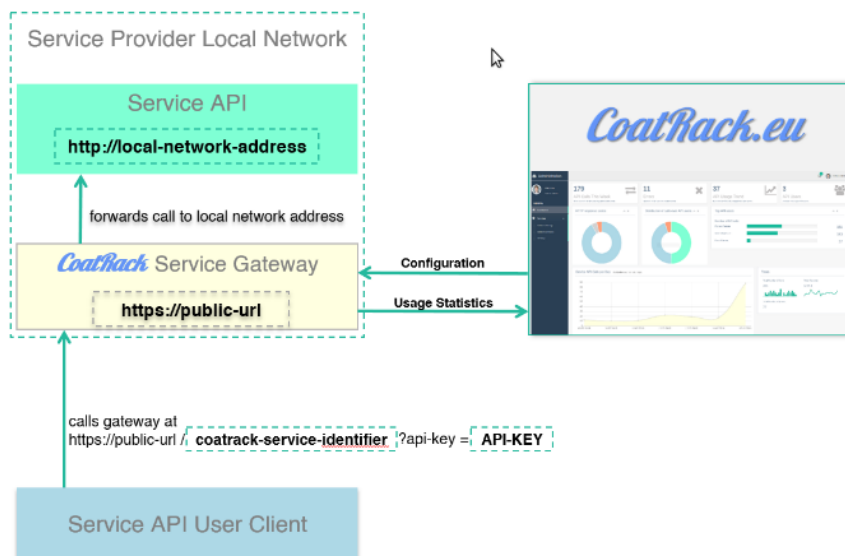


Figure 27: General architecture overview of CoatRack for Service Monetization.

Based on the architectural model as presented in Figure 27, it is possible to offer an open can reusable component that provides the following benefits:

- **P2P Interaction of Software Services:**  
Enabling a peer to peer interaction of software services. The communication payload is just exchanged between the provider and consumer of the service and not routed via CoatRack. This

avoids communication bottlenecks as well as assures that CoatRack will not track service content information that is of a confidential nature.

- **Proxy architecture for a high secure environment:**  
The proxy architecture facilitates an operation in a high secure environment. The offered service is only exposed via the gateway and usual security mechanisms can be deployed.
- **Access control with individual API keys:**  
The access control is realised with individual API keys that are centrally generated via the CoatRack frontend. Only by using the specific API key, a service consumer will be authorised for accessing the offered service.
- **Using different Gateways for multiple services:**  
Each service that is offered is using an individual gateway. Therefore, there will be different gateways for multiple services of a service provider. A proper listing of services and download of gateways is managed via the central CoatRack frontend.
- **Automatic reporting on software service usage:**  
CoatRack provides an automatic reporting of the software service usage. It provides detailed about the type of service usage and the frequency.
- **Different pricing policies for individual services:**  
Service providers can define different pricing policies for the way an individual service is used by the service consumer. At the same time, an additional gateway could be generated for the similar service backend for allowing deviating price policies e.g. for different type of partners/customers.

Throughout the runtime of IoF2020, CoatRack is provided by WP3 partners without costs to all IoF2020 use cases. On top of that, CoatRack is also provided as a freemium offering outside the IoF2020 consortium and after the project and use case completion. CoatRack is providing a not for profit usage with up to 10,000 service requests/month for free. The commercial usage asks for max. 10% of pay per access fees, while a minimum of 90% remains with the service provider. Flat rate usage can also be offered on an individual basis and with paid subscriptions. Individual offerings are provided for customised solutions and supporting infrastructures with multiple services and more complex partner structures.

Section 4.8.3 is explaining the usage of CoatRack, based on the current installation as provided to the IoF2020 use cases and other public stakeholders.

### 4.8.3 Example for the Usage of the current CoatRack Installation

This section is explaining the usage of CoatRack, based on the following example:

- The farmer Frank is owning and operating several weather stations in his fields. He is aggregating the measurements with a backend service that forwards the data to his local decision support system.
- His weather station provider was proposing that he can earn some Euro or just do interested parties a favour, if he would expose the data also for other potential users.
- The weather station provider was getting this idea, since Walter – operating a weather forecasting service – was calling him and asking if he has an idea on how to acquire weather related measurement data in higher detail, but without being urged to install and maintain numerous weather stations on his own.
- Frank was wondering about the cost-benefit ratio, as it might become an expensive exercise if he would need to pay software and interface development.
- At this point, his weather station provider explained him about CoatRack that is fully deployed and only asks for a minimum sharing of revenues, if a certain amount of calls is achieved. On top of that, Frank has already a running service aggregating the data from the weather stations that can be connected to the internet via a CoatRack gateway.
- Hence, the effort to remotely install this by his weather station provider that tried this already several times is less than an hour.

- After having registered the provided service as well as downloaded and installed the gateway, Frank's service can be used by Walter to acquire additional data for his weather forecasting service.
- The CoatRack frontend is managing the offering and tracking of the service usage. It is recommended to use some meaningful names for the offered service.
- Finally, Frank is providing the data from his weather stations that are located around his farm (i.e. 53.079296 8.801694). From that moment on, Walter can use the service offering as nowadays from many others and reimburse the usage as agreed to when doing the set-up.

You can directly access CoatRack via the following URL: <https://www.CoatRack.eu> and start offering your own services.

Appendix B: Step-by-Step Explanation of using CoatRack provides a detailed description of how to use this component.

#### 4.9 Data marketplace

FICODES enhances the System of Systems approach with two different but integrated FIWARE-based components: a data marketplace and a customizable dashboards system.

These components can be deployed as depicted in the figure below.

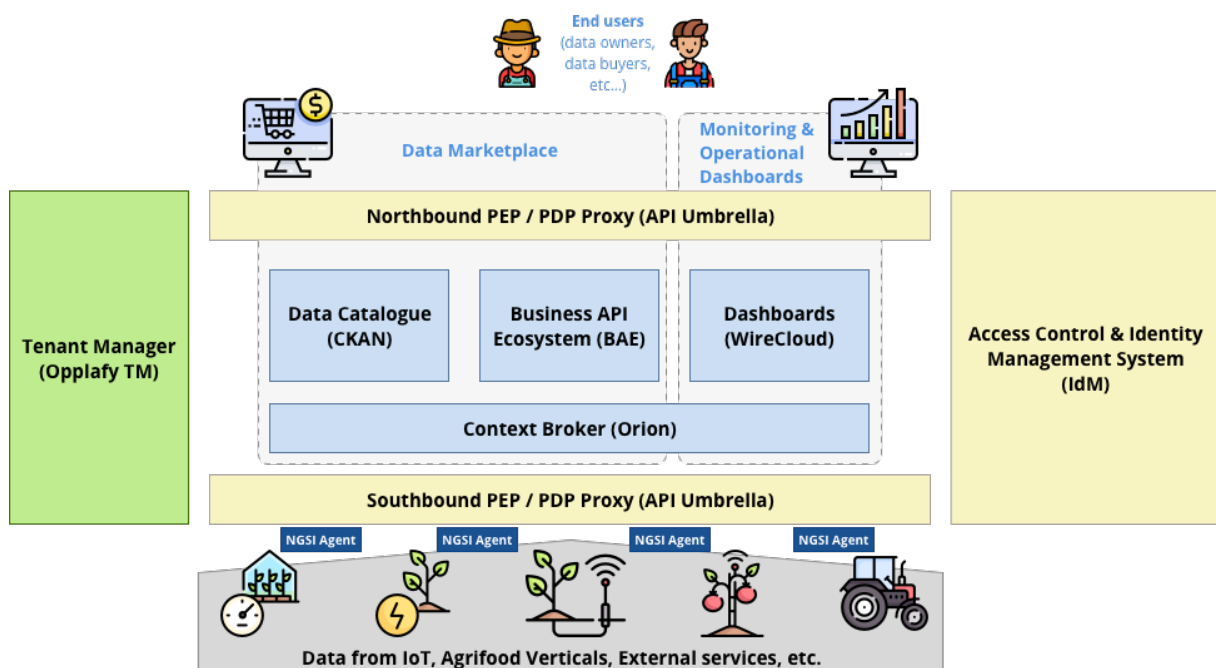


Figure 28: Overview of the System of systems architecture featuring Data Marketplace and Configurable Dashboards.

As depicted by the figure, the Data Marketplace component is made up of the Data Catalogue and Business API Ecosystem FIWARE Components, while the Configurable Dashboards System is based on FIWARE's *WireCloud*. All components are fully integrated and support the NGSI protocol exposed by the Context Broker. The following sections will delve into the details of this architecture.

Within the context of WP3, FICODES is providing a data marketplace made up as the integration of the FIWARE Generic Enablers offered in the API Management and Monetization chapter for the marketplace itself, as well as other required components from the Security chapter devoted to the enforcement of access rights.

The functionality of the Data Marketplace is offered as a combination of the *Business API Ecosystem*, element able to monetize digital assets, which are going to be NGSI-datasets in this case. Together with it,

a data publication platform as CKAN provides catalogue features such as publication and discovery of NGSi-data, and the FIWARE Security Framework, which provides identity features, definition of complex access policies, and data-access monitorization. The details of these components are provided below.

## FIWARE Security framework

For the creation of a data marketplace, the different publication and monetization components rely on the FIWARE security framework. These components provide the means for the management of users, roles, permissions and access policies, core for the monetization of data assets.

Within the current implementation of the data marketplace, two different components are used:

- *FIWARE IDM, Keyrock*: This component provides the means for the registration of users and roles as well as the authentication features using OAuth2.
- *PEP/PDP API Umbrella*: This component supports the definition of the access policies bound to the different system roles. Moreover, API Umbrella works as a proxy validating access tokens issued by Keyrock and providing the access authorization mechanism required by the system. Finally, this component accounts all the requests made to the secured data services generating an access log that can be used by other components for pay-per-use, terms and conditions validation, etc.

## Business API Ecosystem GE

The reference implementation of the FIWARE Business API Ecosystem Generic Enabler (BAE) is the marketplace used for monetizing NGSi-data generated in the context of IoF2020 by any IoT agent or data source, and uploaded to a Context Broker in the form of NGSi entities with their attributes. Indeed, any NGSi-data is being monetized, not only sensor-generated data, but curated information in the agents themselves, or produced by the application of business logic (through CEP, Big Data processing, etc.).

BAE is a marketplace able to monetize any digital asset, and extensible in the form of plug-ins for offering advanced functionality on certain types of assets. In the context of IoF2020, BAE is deployed with a special focus on the monetization of data in the form of NGSi. Nevertheless, basic BAE plugins for the publication of apps will be also installed.

BAE component provides functionality for a complete product management and asset monetization. In this regard, BAE component supports the creation of rich pricing models for the monetized products including single and recurring payments as well as pay-per-use. This price models can be combined and enriched with price alterations enabling the creation of complex models such as a pay-per-use model with an initial fee or a model with a discount when the usage is over a threshold.

In addition, BAE component supports the definition of the license, and terms and conditions that apply to a particular product in order to be accepted by data customers, and the definition of SLAs applying the data consumption.

The BAE component is highly integrated with the rest of the FIWARE ecosystem, using the security framework for the management of users and permissions and the API Umbrella monitorization feature as the source of the accounting information used by the pay-per-use pricing models. Moreover, for NGSi based data offerings it is integrated with FIWARE extended CKAN publication platform, enabling both the creation of offerings from a published dataset or the publication of a dataset from an existing offering.

## FIWARE CKAN Extensions

In addition to the BAE component, a CKAN based data publication portal is provided. This portal is enriched with a set of extensions developed as part of FIWARE and provide the means for the integration of CKAN with the FIWARE ecosystem.

In particular, the portal is fully integrated with the FIWARE security framework and it supports the publication of right-time NGSi data served by a FIWARE Context broker while enabling data providers to choose whether the dataset is open, private or accessible just by a selected set of users.

In addition, the portal is integrated with the customizable dashboards component (*WireCloud*) enabling the creation of rich visualizations for the published data in order to provide an attractive and usable view to final users of the system.



Finally, the FIWARE extended CKAN portal is integrated with the BAE component, enabling the automatic creation of products and offering of private datasets.

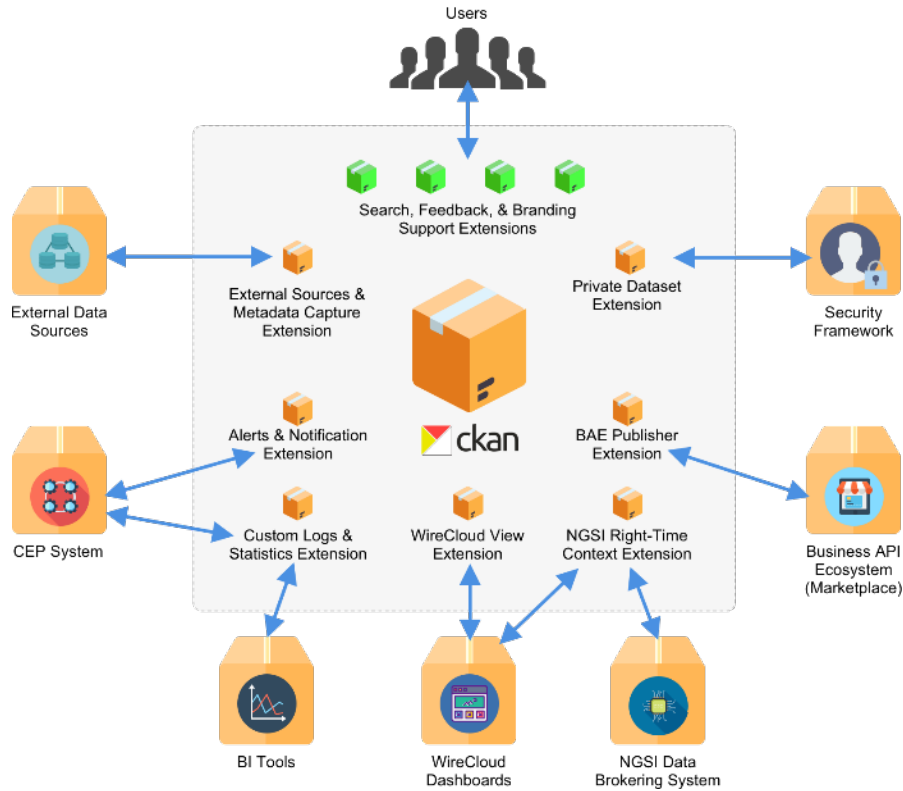


Figure 29: CKAN Extensions

### Tenant Manager

In order to simplify IoF2020 organizations to be integrated into the described data ecosystem, FICODES provides a Tenant Manager component which enables organizations to create an owned tenant in the Context Broker, providing the means to add and remove users.

This software supports organizations to assign roles and access policies to its members, without the need for organization owners to access to the different security framework components and isolating the management between organizations. In this way, this feature will enable each IoF2020 use case to have its own managed data infrastructure, enforcing data privacy and giving organization owners the control over its published data.

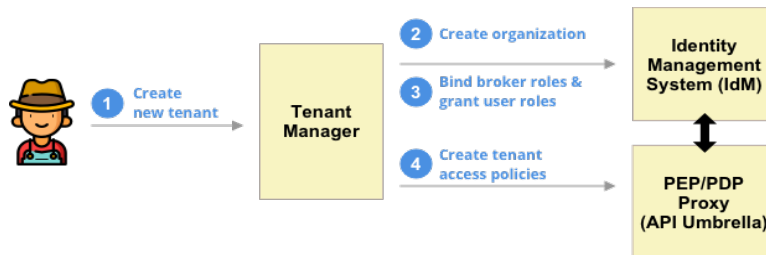


Figure 30: Creation of a new tenant.



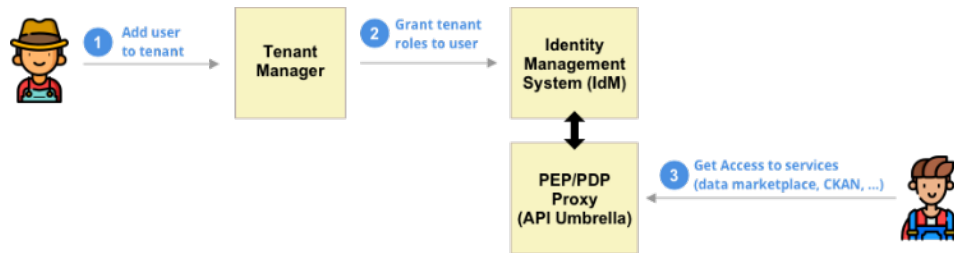


Figure 31: Adding users to tenants

## 4.10 Configurable Dashboards

As part of WP3, FICODES provides the Configurable Dashboard reusable component, realized by the FIWARE Application Mashup GE called *WireCloud*. *WireCloud* is a generic-purpose mashup application tool suitable for creating highly-customizable operational dashboards. *WireCloud* is a mashup-engine where users can create their own dashboards through a combination of components, which are interface-based ones (called widgets), able to show information or gathering user input; and component-oriented components suitable for accessing backend services (databases, context-broker, or any HTTP-based service), and for performing data adaptation and calculation.

The main strength of *WireCloud* is the ability of end-users to customize the interface (dashboard) or create a new one by instantiating widgets, operators and more interestingly, the communication that exists among them. This communication is carried out by asynchronous events sent from one component to others, as the user has established creating “pipes” among the components, in a process called *wiring*. Therefore, the final behaviour of an application (or dashboard) is defined by the chosen interface widgets (with a configurable layout in terms of sizes, deployment, overlay deployment and full-dragboard background), the chosen operators, and the wiring defined among them. Moreover, an administrator can define whether end users can modify/create their dashboards or they are given access to a frozen set of dashboards.

FICODES provides a multitude of *WireCloud* components for generic purposes. In this group is worth mentioning:

- Operators to access a Context Broker through NGSI (both for querying/subscription) and for sending commands to IoT devices through the Context Broker), databases, Quantum Leap (FIWARE component for historical data), etc.
- Data operation components including arithmetic, JSON filtering, selections, joins, intersections, etc; and multiple data manipulation operators focused on data preparation for visualization.
- Considering widgets, there are map components with layering capabilities and possibility to visualize customized POIs, generic data visualizations in panels or tables, and different widgets for visualizing data using graph libraries such as HighCharts, eCharts or Google Charts.

In this sense, it is straightforward to create a dashboard for gathering entities from the Context Broker, access their location and print them on a map (possibly together with multiple types of entities), and visualize data according to one specific entity or aggregated data on them. Operating with the entities is quite similar but usually requires a custom widget offering such functionality.

FIWARE Harmonized Data Models are extensively used in these generic components, since the usage of data modelled as one of the public Data Models makes it possible to use widgets and operators created specifically for FIWARE Data Models. An example of this would be the operator “FIWARE Data Model to POI” which creates Points of Interest of the different entity types already defined in such Data Models, using custom icons and information to be shown over them (the information shown for a streetlight and for a parking lot are not the same). These models promote the reuse of applications, and *WireCloud* components make extensive usage of that feature. FICODES is working on updating the operator to *fully* support current Smart Agri-Food data models listed in section 4.3.

Besides, *WireCloud* does also simplify the creation of domain-specific components. As an example, FICODES took ideas and data from UC 4.2 to showcase a demo in the IoF2020 event in Prague, showing how to create a monitoring dashboard able to mashup real-time data from greenhouses and lorries (logistic), following the System of Systems approach. FICODES will continue to create specific components and dashboards for the different IoF2020 use cases.

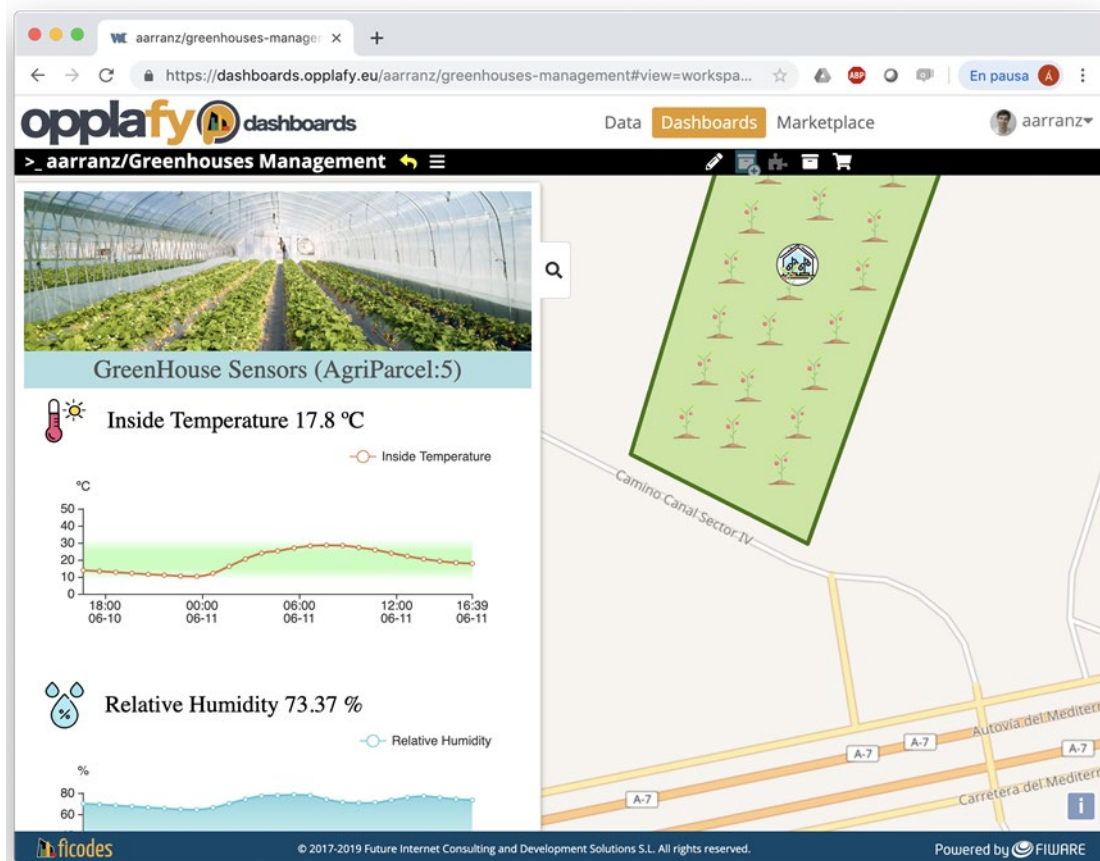


Figure 32: Greenhouse dashboard showcased in the Prague IoF2020 Event.

In addition, for the deployment of custom components devoted to a specific usage/corporation, a catalogue for WireCloud components has been developed, and a specific instance is to be installed for IoF2020. This catalogue of components can co-exist together with a WireCloud global instance to be maintained by the FIWARE Foundation (also with the support of FICODES) with the curated core of main components. The feature to make usage of more than one catalogue as repository makes it possible to use the global repository together with the IoF2020 deployment, where specific widgets, operators and mashups are to be served for distribution among any *WireCloud* instance. Development and deployment tools for components updates are also available to simplify the update/maintenance of current components and for the creation of future ones.

Last but not least, *WireCloud* provides out-of-the-box support for the FIWARE Security Framework. This means the *WireCloud* instance to be deployed into the IoF2020 Lab will be able to seamlessly use the same set of users defined in the System of Systems, with the same roles, permissions, etc.

## 4.11 Conclusions

This chapter has presented different open source components that can enable service monetization and data marketplaces (enhanced with dashboards) in a System of Systems environment for Agri-Food. As the System of Systems approach intends to provide an integrated view for Farms, such components can be extremely useful in harnessing data towards collaboration, portability, replicability and experimentation in Smart Agri-Food data sharing environments.

The final aim is to help use case stakeholders, product owners and developers to design, deploy and configure Digital Farming Systems of Systems around Open Platforms enhanced with Data Marketplaces, leveraging common technology enablers, software components, and related architectures that guarantee the creation of a sustainable ecosystem of portable data sharing solutions for the Farm and Food sector.

However, there are also additional challenges to be faced when deploying Smart Agri-Food solutions based on open platforms following a System of Systems approach. On one hand security, privacy and trust, and

on the other hand usability. The former is critical in a System of systems, where data from different applications has to be carefully protected in an environment that fully respects farmers privacy, while offering a trustable environment, where farmers can feel comfortable sharing their data. The latter is also of high importance, as the efficiency and effectiveness of the daily work of farmers with multiple applications will depend on the proper user interface design and usability of the different Smart Agri-Food integrated services offered. Next chapter provides a summary of security and usability guidelines that WP3 is recommending towards a proper realization of the System of Systems approach.

## 5 Guidelines

The system of systems approach presented in this document is not only supported by specific components, described in the previous sections, but also by a set of guidelines.

The successful adoption of IoT requires robust, secure and user-friendly solutions that can indeed solve problems and add value to the agri-food supply chain. While these issues are critical, they do not represent the main focus of the Use Cases, who concentrated in meeting functional requirements. Therefore, WP3 has been developing work to support Use Cases in two areas:

- Security, privacy and trust guidelines support use cases in analyzing potential threats in their solutions and then developing or selecting solutions to solve them.
- Usability guidelines to support developing teams in realizing usable and user-centred designs, maximizing potential for adoption by the diverse users in the value chain.

### 5.1 Security Privacy and Trust

The proliferation of “Connected Things” driven by the IoT has prompted the need for “Secure Things” that are critical in preventing the sophisticated system attacks. Given the security vulnerabilities of IoF2020, the UC owners are primarily concerned about the safest mechanisms to implement a secure data exchange among several systems.

#### 5.1.1 Security by Design Principle

Thus, to enable UCs to realize secure IoT architectures, we have proposed ‘Security by Design’ principle. Which, as shown in the below picture, is an iterative approach starting with a clear vision of having no security vulnerabilities in the envisaged IoT architecture followed by a Threat Modelling and Security Analysis (TMSA). TMSA in IoF2020 has been realized by means of STRIDE<sup>8</sup> methodology.



Figure 33: Security by design.

In a nutshell, the TMSA (STRIDE) performed by all the 19 existing UCs shows (see D3.7) that most of the UC architectures are inherently vulnerable to security exploitations. Thus, averaging around 50% of critical threats per each UC architecture. To increase the overall security of the Use Case, one must not only select the most secure technologies, it is equally important to design effective processes and train the people responsible for implementing these processes. Please refer to D3.7 for detailed guidelines for solution design with respect to process, people, and technologies.

<sup>8</sup> [https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878\(v=cs.20\)](https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20))

The SPT contents have also been shared with the IoT Catalogue team and would be available soon as a self-explanatory step-by-step procedure. Hence, allowing any interested group to kick start the TMSA for their UC straight away. The second iteration of STRIDE analysis has already been started. We anticipate that the percentage of critical threats would be reduced in comparison to the first iteration. We have also initiated the STRIDE analysis for the Open Call UCs by introducing the UCs to TMSA by means of webinars during their scheduled trial meeting.

### 5.1.2 SPT: WP3 Perspective

Having reviewed all the existing UC IoT architectures, as a part of STRIDE first iteration, and comparing the layered architecture perspective of the UCs (as depicted below), we see that SPT cuts through every aspect of the solution development. Starting from physical device, which can be made safe and secure with secure application microcontrollers<sup>9</sup>, secure elements<sup>10</sup>, to applications built on the information gathered on a backend, which could realize secure communication with users through the traditional HTTPS, FTPS protocols.

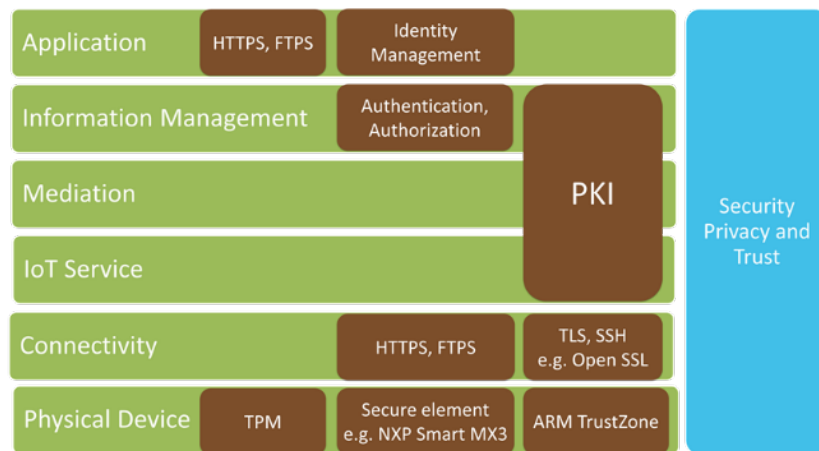


Figure 34: SPT from a WP3 perspective.

## 5.2 Usability

When we take time to test our systems, from a user experience point of view, the most common issue on software products is the gap on usability. Our developments and implementations offer an excellent functionality, but a successful product is in fact measured by the user acceptance quote in the market. Improvements in the user experience may take more effort than the programming itself, especially when the IT-teams did not have clear from the beginning the usability as part of their own development process, and the team does not prepare a strategy for addressing the topic as a potential breakpoint between users, stakeholders and systems.

In recent years, the agile mindset became more and more a popular way to satisfy users through ad hoc valuable software through continuous delivery, however the focus of the development is more functional than emotional. The human interactions are just described in terms of methods or functionalities, and the idea of good user experience is relegated to a secondary role under the technical development.

In the IoT context, usability can be even more crucial, because the interaction with the service (an app, a webservice, a dashboard) is even more important than the device itself. Specifically, a device can be connected and meet the required functionality, but if there is not a good enough user interface, the solution is simply not considered. This statement takes on a special complex dimension, when we extrapolate it to IoT-solutions for the agri-food domain.

<sup>9</sup> [https://www.nxp.com/products/processors-and-microcontrollers/arm-based-processors-and-mcus/i.mx-applications-processors/i.mx-6-processors:IMX6X\\_SERIES](https://www.nxp.com/products/processors-and-microcontrollers/arm-based-processors-and-mcus/i.mx-applications-processors/i.mx-6-processors:IMX6X_SERIES)

<sup>10</sup> [https://www.nxp.com/products/identification-security/authentication:MC\\_71548](https://www.nxp.com/products/identification-security/authentication:MC_71548)

The following sections provide an overview of activities towards integrating usability within the development process and a general approach how to test usability of solutions within the Agri-Domain. Remember: *Usability is not the goal, it is a medium to add value to a solution, which provides a good user-centered design.*

## 5.2.1 User-Centered Design

There are some ground criteria to assure usability on Software as a Service (SaaS), which can be used to build even a check list manifest for the IoT-Solutions. However, it might be useful to clarify a methodology and the role of the user experience before addressing the management of a continuous usability-delivery. Those topics should be known even before performing a usability-analysis or a usability-test.

The goal of usability is overall to offer the functionality of a solution in a comprehensible and intuitive way, centred on the needs and desires of the users, which is providing a good experience of the product based on the following milestones<sup>11</sup>:

- **Equitable Use:** useful and marketable to users with diverse abilities
- **Flexible:** wide range of user preference and abilities
- **Simple/Intuitive:** easy of understand, regardless of user's knowledge or skills, or current concentration level
- **Perceptible:** effective communication of information through user's sensory
- **Tolerance for error:** minimizing consequences of misusing or unintended actions
- **Low Effort:** interactions are efficient and comfortable with a minimum of memorization or concentration
- **Size/Space:** Appropriate size and space to allow manipulation and mobility

If we summarize the list, the objective of tracking usability, during the development of solutions, is to enable users to achieve goals effectively, efficiently and with satisfaction, taking account of the context of use, because usability is dependent on the specific domain, in which the solution is used.

At this point, it seems clear that a software solution focused on meet the user needs requires a methodology for achieving usability during the solution development process and ensure a quality level of the user experience. The usability engineering provides a couple of frameworks for this goal, but a well-known methodology for carrying out usability is called *User-Centered Design* (UCD). The UCD is a highly structured product development process, which declares six core principles in order to provide the structure for individual procedures during the design:

- Set business goals
- Understand users
- Design the total customer experience
- Evaluate designs
- Assess competitiveness
- Manage for user feedback

When the development team defines activities related with the usability of its solution, these actions might be broken down into four phases to have a iterative usability implementation process: Analysis, Design, Implementation and Deployment. These phases do not depart much from the agile framework and can therefore be integrated into it.

---

<sup>11</sup> The center for universal design (1997): The Principle of Universal Design, [http://www.ncsu.edu/ncsu/design/cud/pubs\\_p/docs/poster.pdf](http://www.ncsu.edu/ncsu/design/cud/pubs_p/docs/poster.pdf)



The aforementioned User eXperience (UX) is the goal and the core of usability. The term involves emotions, beliefs, preferences, perceptions, comfort, behaviors, and accomplishments that occur before, during and after use the solution. A quality UX is a consequence of the interactions between three categories of factors:

- from the solution or system: brand image, presentation, functionality, system performance, interactive behaviour
- from the user (internal and physical state): prior experiences, attitudes, skills, abilities and personality
- from the context of use.

The User Experience encompass what the users desire and require and the solution should be implemented functions that are desirable and useful.

### 5.2.2 User Experience on an Agile Project

Agile Mindset promotes the early and continuous delivery of software solutions. The principles and methodology of the Agile framework are not to be discussed in this document. However, it is remarkable that the integration of user-centered design and the usability as a quality parameter are barely covered within an agile project.

Since the IoT solutions have more and more a SaaS business models, this situation imposes special challenges on the development process specially because the developed IoT Services are focused on user-interactions and the IT-team should integrate UX/UI activities during the iterations at the beginning of each project

In the UCD-Methodology, the UX activities can be planned in phases: Analysis, Design, Implementation and Deployment. Just like the phases within agile iterations, the activities of the UX-team can be planned parallel and be part of the development process during the iteration or take place before it. The most important aspect is gathered requirements and desires from the users, modelling an “ideal solution”, which is not specified in detail, but it is built to share impressions and understandings of the solution. During development iterations, the team need to adopt UX-Artefacts, such as Personas and Scenarios, developers need to explore the user interfaces and they must allow the UX/UI experts to create low prototypes with integrated workflows and design to optimize the usability.

As guideline for the implementation of the UX/UI procedures, the development team can begin with a short description of the following usability topics (these questions will support a comprehensible user experience modelling in the development process):

Table 7: UX/UI Questions on an Agile Project

<b>Personas-Definition</b>
<ul style="list-style-type: none"> <li>• Who are the users, what are their knowledge, and what can they learn?</li> <li>• What do users want or need to do?</li> <li>• What is the general training of users?</li> <li>• What is the context in which the user is working?</li> <li>• What should be left to the solution? What to the user?</li> </ul>
<b>Stakeholders Feedback</b>
<ul style="list-style-type: none"> <li>• Is the logic of the solution clear?</li> <li>• Data models are clear enough?</li> <li>• Scenarios: Walkthrough the model and validate that it is able to support the workflow (scenario) of the use case.</li> </ul>
<b>Usage of Solution</b>
<ul style="list-style-type: none"> <li>• How many devices should one user install, configurate and manage?</li> <li>• Is the installation and the connection clear and easy?</li> </ul>



- Do users need help for the configuration of their devices?

The techniques to find answers to these and other questions are mainly user-focused. A good approach for the integration of UX on an Agile project is the adoption of the *Design Thinking* paradigm. This framework is a collection of processes to address the design and development of products together. A detailed description is out of the scope of this document, but the success of its adoption in many areas testified its potential in the Agri-Food domain. The core features of the Design Thinking are:

- Find Problems
- Generate solutions
- Sketch and Mock-up the solutions
- Model and prototype
- Test and Evaluation

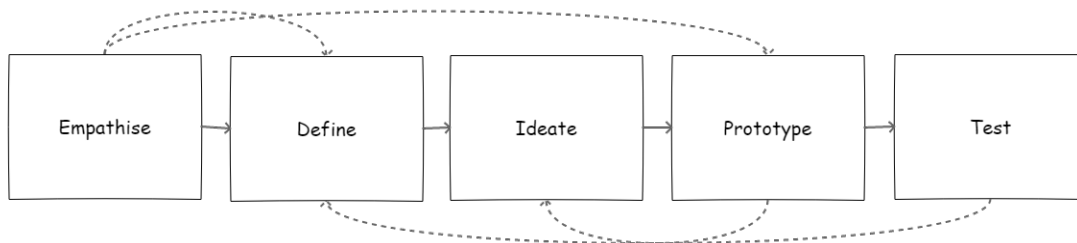


Figure 35: Design Thinking Processes

### 5.2.3 Usability: Analysis and Testing

Ensuring a solution with high usability and good performance can be done through user testing. The test can be divided into usability analysis, usage testing and acceptance testing. The distinction between the three types depends on the part of the solution to which they are focused.

Usage testing is a method by which users of a system are asked to perform certain tasks in an effort to measure the system’s easy-of-use, tasks time, and the user’s perception of the experience. This kind of test are performed with real users of the solution. Normally a sample of 15 system users can reveal more than the 70% of system usability deficiencies after a first release and after the test, the next sessions could be performed with a sample of just five users. However, in order to perform a good test and to be successful with the future improvements, the tasks, which must be completed during the test, must be well defined and the tester shall be impartial at the time of the test, i.e. the tester shall not interfere in any way with how the user performs the tasks. Even if the user asks for help it is assumed as an indication of usability failure of the solution.

Table 8: Usage Testing

<b>Usage Testing (How does the user use the solution?)</b>
<ul style="list-style-type: none"> <li>• Can users easily perform their planned tasks?               <ul style="list-style-type: none"> <li>○ can users perform planned tasks at the expected speed?</li> </ul> </li> <li>• How much preparation do users need?</li> <li>• What and how many mistakes do users make when interacting with the product?</li> <li>• Can the user recover from errors?               <ul style="list-style-type: none"> <li>○ What do users need to do to recover from errors?</li> <li>○ Does the product help users recover from errors?</li> <li>○ Does the software display informative, non-threatening error messages?</li> <li>○ Searching for Help?</li> </ul> </li> <li>• Have measures been taken to meet the special needs of users? (That is, has accessibility been taken into account?)</li> </ul>

Usability analysis is a more technical type than the previous ones and often represents a detailed follow-up of use cases and scenarios performed by an expert with help of stakeholders and users. The purpose of the test is to evaluate the usability of the solution using heuristic evaluations and statistical tools to find the issues to be improved. Currently, the following categories are evaluated in software solutions (regarding to the user interface and the user experience):

- visibility of system status
- user language/vocabulary (domain specific)
- control and freedom for the user
- consistency and standards
- back before cancel
- flexibility and efficiency of use
- dialogues and minimalist designs
- help and documentation

For each category, a specific list of questions exists, which helps to “score” the solution. Typically, these questions are presented as a table with a yes/no answer. Depending on the solution, there are aspects more relevant than others, but the main objective of the test is to check concrete aspects of the design and the workflow from an objective point of view.

Table 9: Usability Analysis (Example)

Category	Yes	No	NA
<b>Visibility</b>			
Titles and descriptions are clear?			
Icon selection is clear?			
UI-Menus are clear and have usual options?			
UI-Response time is appropriate? (<2s)			
If the UI-Response is more than 10s, the solution informs the user about it?			
<b>Language/Vocabulary</b>			
Is the language familiar for the users?			
Is there a usual sequence to select menus and option?			
Are the formats and conventions understandable for users?			
If input-data is required, is that clear for the user?			
Questions are clear?			
Domain vocabulary is appropriated for the user?			
<b>Control</b>			
Multiple Windows: Is easy for users to change through windows?			
Can users cancel actions?			
Can users move back and forward without problems?			

Category	Yes	No	NA
Does the UI ask the user for action confirmations?			
<b>Consistency</b>			
Icons have labels?			
Each window has a tittle?			
Scroll bars in the same place (each window)?			
<b>Back before Cancel</b>			
All required input-data is clear during a transaction?			
Are return actions used to avoid forgetting entries?			
Is color used to highlight inconsistencies in data entries?			
<b>Flexibility</b>			
The system offers shortcuts for the users			
The system offers “settings” for expert users			
Users can define their own “common tasks”			
<b>Dialogues/Concepts</b>			
Are the icons consequent with the concepts that they represent?			
Are tittles and names enough short to explain the UI?			
Popups explains shortly and briefly their messages?			
<b>Help</b>			
Instructions are clearly explained			
Procedures are easy to follow			
Information is easy to find			

Finally, the acceptance testing is more technical than the others. Those tests provide the status of the performance of the solution and they are normally automated. Automated tests are running often, at least daily. At the end of an iteration, the team will deploy its current development into a Quality Assurance testing environment, where the test is performed. The development process continues while the QA-environment reports issues in the current solution (n-iteration). These reports are treated like any other requirement. The team estimates the issues again, prioritizes it and puts it on the tasks list or backlog to be addressed at some point in the next iteration (n+1 iteration). The QA-expert uses test-tools like FIT (Framework for Integrated Test), Cucumber (Requirement specification), JUnit (Unit Testing). The objective of these tools is to provide a way to write routines for automated task, which are supposed to follow a specific workflow for multiple scenarios.

In 2018, WP4 offered an UX/UI-Webinar and a questionnaire called “Acceptance Analysis”, in order to support development and improvements on the MVPs for all the use cases. The documentation could be founded following the next links:

- Questionnaire : [https://wur.az1.qualtrics.com/jfe/form/SV\\_cFRo7vTWr4oGJTL](https://wur.az1.qualtrics.com/jfe/form/SV_cFRo7vTWr4oGJTL)
- Webinar: <https://3.basecamp.com/3618432/buckets/2138378/vaults/1374486302>

## 5.2.4 Usability Summary

In this document, usability relates to the outcome of interacting with a solution and it is considered as a quality parameter of the user experience and it can contribute to the product being usable and desirable in a particular context of use.

Usability is relevant to (according with the ISO13407):

- Enable users to achieve their goals effectively, efficiently and with satisfaction
  - Regular users: ongoing use
  - New users: learning process
  - Casual users: infrequent use, reuse.
- minimizing the risk and the undesirable consequences of use errors; and
- maintenance, in that it enables maintenance tasks to be completed effectively, efficiently and with satisfaction.

Usability is relevant when designing or evaluating interactions with solution for the purposes of:

- user focused development in order to perform better interactions and more satisfying experience of usage
- review or comparison in order to improve the solution (next versions, new releases)
- marketing and market research to achieve more users/implement new usage-trends/maintain the product alive.

## 6 Added Value for different Business Models

---

One of the main objectives of WP3 is to provide horizontal IoT support to all use cases in the project. The underlying aim is to minimise the effort for the development of non-value adding features as well as to facilitate the reuse of available software and hardware components. Those technologies are provided, configured and/or enhanced to fit or enhance their individual solutions. The WP3 offer, described extensively by **D3.6**, enables use cases to:

1. Easily access generic re-usable components, which are not in the core of their individual solutions but are necessary to enable them. This is specifically the case of usability, security, privacy and trust components.
2. Adopt the system of systems approach supporting the individual solutions in sharing their data, and using verticals or services from other solutions. This is supported by the harmonized data model fragments WP3 is developing and the architecture presented, centred around the FIWARE Context Broker.
3. Use components for monetizing their services and data, adding value to their overall solutions and potential benefits.

### 6.1 Added Value of Reusable Components

With regards to reusable components and from a benefit perspective, use cases have the potential to:

- Reduce costs in implementing interfaces to different systems needed for their individual solutions;
- Enhance their solution by incorporating additional features, components or services;
- Facilitate the realization of reliable and mature solutions by using previously tested and validated components; and
- Increase revenue by further monetizing services and data generated within the use case to external solutions or platforms.

Until the end of the project, WP3 will continue development of these reusable components and elaborate associated business plans (in collaboration with WP4), to enable not only sustainability of the components, but a suitable understanding and validation of their potential by use cases. This means we will aim at the elaboration of plans on two levels: what can use cases gain from using these components: and how can the WP3 developing team commercialise the components, or at least make them viable after the project ends.

In this work, WP3 will collaborate with WP4, and define revenue models, detailing estimations of costs and benefits of using the components being offered by WP3. It is of course difficult to define revenue models generic enough for several types of actors in the value chain, with quite different solutions and infrastructures. In any case, we will aim at provision of guidelines on how to estimate the possible revenue.

In addition, the partners responsible for each reusable component will elaborate a business plan aiming at the sustainability of their respective component after the lifetime of IoF2020.

### 6.2 Added Value of the System of Systems Approach

The main value added of the System of systems approach (or Platform of Platform as already identified by WP4) is that *Farm Management Information Systems* will be able to provide users an integrated view, encompassing information from different verticals and mashing-up the best of breed user interfaces. In the end, Context Information associated to a farm will be enriched with the contributions coming from different vertical solutions (System of Systems), all of them able to share data among each other and enabling a further optimization of processes, saving time, money and resources.

From a business point of view expected benefits of the System of Systems approach are numerous:

- Allow an open and competitive marketplace of compatible farm management systems and vertical smart farming solutions.

- Lower costs to achieve interoperability of vertical solutions or their integration with farm management systems.
- Allow a healthy public-private collaboration ecosystem around Digital Farming.
- Improve user friendliness: easing the management around a single stop shop offered by Farm Management Information Systems (FMIS).
- Allow modularity by adding platform components parallel to business needs.

Some of the conclusions of research on the initial 19 Use Cases ratify this view, as industry requires a data exchange solution to make data from different sources in a simple way available for innovative farm services. In general terms, the potential of integration of IoT platforms, cloud platforms (such as FIWARE or 365FarmNet) and FMIS, creating a System of Systems, improves decision-making, production, animal welfare, resource efficiency and satisfaction of consumer demands.

Several value proposition examples documented in WP4 mention potential benefits of a System of Systems approach for the different stakeholders listed below:

- Farmers
- Supermarkets and food processors
- Developers (who are searching for data to be used for applications)
- Government (who can provide and consume data concerning regulations on food, use of pesticides, animal welfare, etc.)
- Consumer of the product of the farms

The following subsections elaborate more on those benefits.

### 6.2.1 Added Value for Farmers

A clear trend in the distribution of smart farming services goes towards digital marketplaces for smart services on farm management information systems (FMIS) that already manage the master data and bureaucracy of the farm. For a long time, especially larger players, tried to come up with a full featured solution for the farmer and bound them in some kind of vendor lock-in to their hardware products or side services. This time is more or less over and service providers see the value of FMIS marketplace (a System of Systems) for a quick uptake and reach for new farm service.

According to studies made by WP4 with inputs from WP2 in UC 3.3 (Automated Olive Oil), the use of an FMIS integrated solution (which can be based on a System of Systems approach) and other connected IoT platform that manage post-harvest, the following benefits were estimated:

- 10% in increased yield
- 15% reduce total average costs on olive oil quality management
- 10% reduction of the processing time in milling
- 10% quality increase.

Furthermore, all the data in the value chain would be consolidated improving the Decision Support Systems (DSSs). Under this scenario integration and data consolidation increase in 50% compared to the pen and paper method that many traditional olive farmers follow. Additionally, irrigation system will be done automatically achieving 100% precision.

On the other hand, farmers producing multiple goods will be able to get access to different applications that will provide a consolidated view of the business and production processes in one place, gaining a holistic view of their business and the relationships with other stakeholders, including the government.

Last but not least, a Software as a Service model for FMIS, as the one introduced by the System of Systems approach is very interesting for farmers, as they do not need to invest on the operation and maintenance of on-premise computing platforms.

## 6.2.2 Added Value for Supermarkets

Concerning supermarkets, Intelligent Fruit Logistics use case (3.4), reported that the EPS Tray Radar will among other benefits be used as an anti-theft solution that sends alerts to clients or Europool. The value chain will be fully transparent achieving an increase of 25% on its efficiency, 15% in security for distribution and 50% improved traceability. Furthermore, the Temperature Monitor will allow 5% less food waste.

The former kind of solution can be integrated (in a System of Systems) with Grow Wise Control Systems, supporting demand driven production connecting the supply management system of a supermarket to schedule the next growth cycles in connection with the demand, including the demand of trays. And that will increase, indeed, the added value of the solution, thanks to a System of Systems approach.

Additionally, the above integrated solution (Fruit Logistics plus Grow Wise Control Systems) can work on a fully automated vertical farm setup that is steered by a smart control system. The idea is to setup these vertical farms near cities to provide in a very predictable way high quality and even enhance fresh leafy products to supermarkets with a minimal environmental impact if the electricity is sourced out of renewable power production (controlled by a subsystem within the System of Systems). Furthermore, it is possible to produce plants with enhanced nutrition (e.g. higher vitamin level) and all products are completely free of pesticides and external allergens.

Overall, the major advantage for supermarket and food processors is the all-season availability of certain food products and the risk-free on-demand production.

In the case of a Logistics Monitor integrated with a Poultry Chain Manager, the major end-customer of these two services is integrated poultry producers. On the second level, also food companies and supermarket chains can be a customer. They are interested to achieve a high level of transparency regarding quality and animal health towards the end-consumer and trustfully justify higher prices for quality meat.

## 6.2.3 Added Value for Developers

With a Systems of Systems architecture connecting external devices and machinery of all major farm machine manufacturers in Europe, data can be received and mashed-up in real-time and in a safe way from farm machines and other devices to send orders back to the machine (or other actuators) for automatization of production processes on the field. This can encompass seeding, fertilization, spraying or harvesting.

Software developers can gain access to new sorts of data directly from the machine that enables new management and forecasting algorithms. Based on the smart data, the software can create precise acting propositions for farm machines and send these orders directly to a machine on the field.

On the other hand, as already reported by **D4.9**, the time of farming app stores in the mobile field is ahead and Europe needs to find central platforms for certain sectors soon as infrastructure for a proper IoT uptake in farming. The opportunity for developers is huge, as the demand for this kind of IoT-enabled, innovative Agri-Food solutions is raising on a yearly basis.

## 6.2.4 Added Value for Governments

For Governments the potential added value can also be significant. On one hand a System of Systems approach can enable Public-Private collaborations at different levels (regional, national, trans-national). For instance, a regional government can host an FMIS Platform which can be shared by subscribed farmers, getting access to different smart farming applications. As a result, the costs of farm digitisation would be shared between public administrations and farmers, lowering barriers of entry, especially in less developed regions where farmers cannot afford big investments.

On the other hand, a System of Systems approach can enable collaborations between the public and private sector concerning data sharing. For instance, a model for Meat Information Transparency (MTIS) (in partnership with leading supermarket chains and high-quality food processors) a marginal price premium can be communicated to the end-consumer for independent and objective food quality transparency. This premium would be collected by retailers and forwarded to the MITS platform. Furthermore, part of this money could back governmental subsidies for investments by farmers or processors in sensing technology.

Last but not least, in the light of stricter environmental regulation to improve the sustainability of farming in the future, there is an increased need of smart farming services and especially FMIS to have access to



national, regional and local regulations for the application of fertilizer and pesticides. This will also facilitate the communication between the government and users of the value chain.

### **6.2.5 Added Value for Consumers**

All the benefits detailed in previous sections of this chapter will finally propagate to the final agri-food products and hence, consumers. In fact, the availability of Digital Farming solutions and, particularly those enabled by open platforms following a System of Systems approach, will also provide, indeed, an (indirect) benefit to consumers. Among those benefits, there will be an increase in transparency and an improved food quality (less pesticides, for instance) with better price, while following more sustainable and environmentally-friendly production schemes (using less water or optimizing logistics, for instance).

## 7 Conclusions

---

This report has facilitated an understanding by different stakeholders (particularly use case owners and developers) how Open Platforms and other accompanying components can be extended, configured and leveraged to implement a System of Systems approach in the domain of Agri-Food. From our analysis of use cases, it can be concluded that Open Platforms are increasingly adopted within IoF2020, and as a result, the concept of **System of Systems** can be introduced more easily. This novel approach for the development and deployment of integrated, multi-vendor Digital Farming Solutions, is intended to maximize scalability, interoperability, maintainability and sustainability aspects. To this aim, the *IoF2020 Lab* encompasses all the different assets that can ease the development, deployment and publication of Digital Farming Solutions.

There is an opportunity to **integrate innovative solutions coming from different parties**. It will be based on the integration of information generated by different solutions to build a holistic picture of what is going on at the farm. As a consequence, *Farm Management Information Systems* will be able to provide users an integrated, holistic view, encompassing information from different verticals and mashing-up the best of breed user interfaces. In the end, Context Information associated to a farm will be enriched with the contributions coming from different vertical solutions (**System of Systems**), all of them able to share data among each other and enabling further optimization of processes, saving time, money and resources. This document has described the different architectures to realize the System of Systems approach and has reported on a **Node-Red** extension, already contributed to FIWARE, that can be used by developers to prototype the integration of existing Agri-Food Apps into a System of Systems.

To that aim, the availability of **shared, well-adopted information models** is a key interoperability mechanism for enabling a global market for IoT-enabled Digital Farming. Such models provide an essential element in the common technical ground needed for standards-based innovation, by making replicability and portability of Smart Agri-Food solutions practical. This allows for sector-specific focus in a procurement or development process, while maintaining cross-domain consistency. WP3 is working on different Harmonized Data Models, based on previous work made by GSMA.

Besides, for realising the IoF2020 vision of System of Systems, it is necessary to count with integrated Open Platforms that can solve major development challenges, while assuring compliance to the overall architecture. One strategic element is **FIWARE** that represents an open source platform for implementing Smart Solutions, including but not limited to Digital Farming. FIWARE includes different components gravitating around the *Context Broker*, a “data broker” which offers standardized interfaces to publish, store, and consume (through NGSIv2 and NGSI-LD) data tagged with associated temporal and spatial metadata (context data).

WP3 has been developing a set of reusable components to provide horizontal IoT support to all use cases in the IoF2020 project. These components directly support the System of Systems approach, as described in this deliverable. WP3 is currently tightening the collaboration with use cases, establishing demonstration scenarios for each component, using data and solutions from the use cases. Our plan is to have fully functional reusable components in Spring 2020, demonstrated in the scope of use cases’ individual solutions.

The final aim of our work is to help use case stakeholders, product owners and developers to identify configure and extend a set of common technology enablers, software components, open platforms and related architectures that guarantee the **creation of a sustainable ecosystem** of portable and integrable solutions for the Farm and Food sector. Such solutions, within a System of Systems, are aimed at creating a rich marketplace and ecosystem for smart Food and Farming, representing an unprecedented opportunity to produce value and create business opportunities, by applying data-driven solutions. In addition, it can help stakeholders from other business domains to identify potential cross-sector implementation synergies. Ultimately this will allow building an IoT ecosystem around the IoF2020 project and around Smart Farming in Europe.

## 8 Appendix A: Node-Red FIWARE Extension Nodes

The *Context Broker* Node is a Configuration Node to set up the endpoint and security parameters of a FIWARE Context Broker (exporting NGSIv2 or NGSI-LD). The figure below shows the User Interface of this Node which allows to introduce a Context Broker endpoint and optionally all the needed security credentials. Using the logic provided by this Node, in conjunction with other Nodes of this package (see below), applications do not need to implement all the glue code to get access to or to update data stored by the Context Broker.

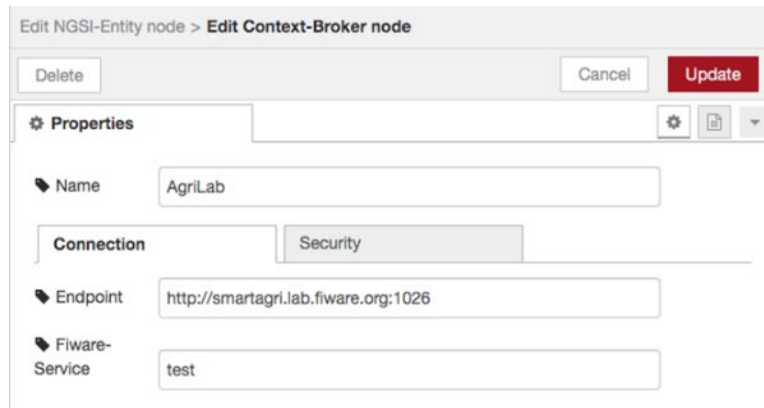


Figure 36: Context Broker Config Node UI.

The *NGSI Entity* Node user interface is shown below. It allows to configure an endpoint (defined by a Context Broker Config Node), protocol (NGSIv2 or NGSI-LD), LD @context, attributes, representation (normalized or keyValues) and MIME type required to retrieve an Entity. The Entity Identifier shall be provided through the Node's input payload. This Node enables an easy retrieval of the data of an Entity of interest to an application.

In the example below, the *location* information about a certain *AgriParcel* will be retrieved.

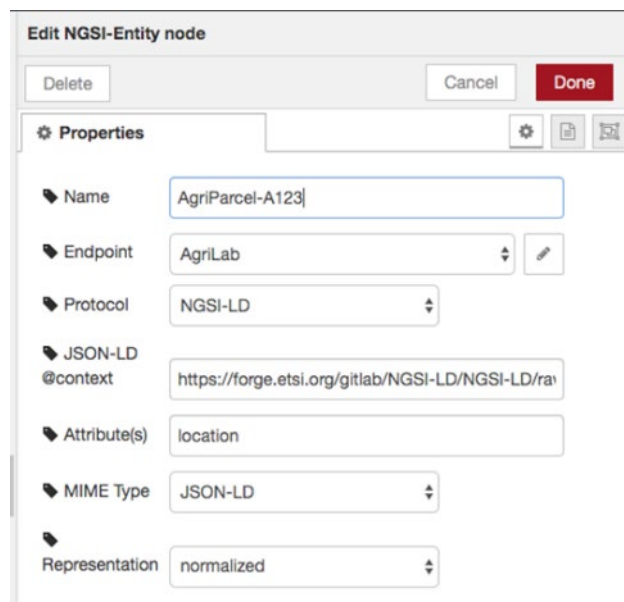


Figure 37: NGSI Entity Node UI.

The *NGSI Dataset* Node user interface is shown below. It allows to configure an endpoint (defined by a Context Broker Config Node), protocol (NGSIv2 or NGSI-LD), LD @context, attributes, Entity Type, filter, representation (normalized or keyValues) and MIME type required to retrieve a set of desired Entities. Some of the referred parameters can also be provided through the Node's input payload. This Node enables an easy retrieval of the data corresponding to a set of Entities of interest to an application.

In the example below all entities of type *AgriParcel* (*location* and *name* attributes) belonging to a certain *AgriFarm* will be retrieved.

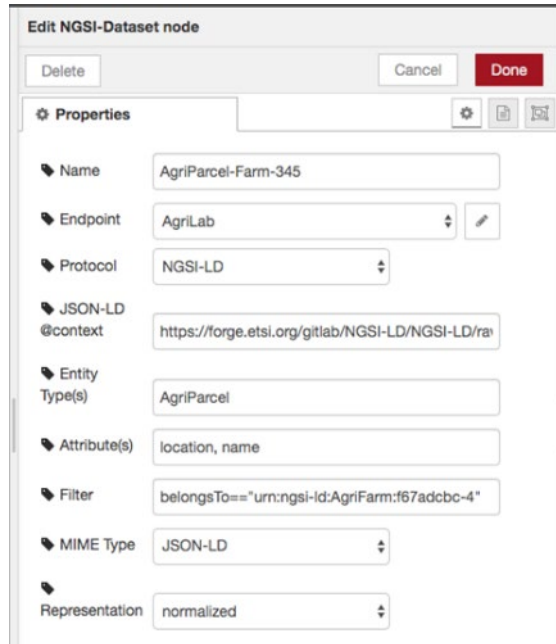


Figure 38: NGSi Dataset Node UI.

The *NGSI Subscription Node* user interface is shown below. It allows to configure an endpoint (defined by a Context Broker Config Node), protocol (NGSIv2 or NGSI-LD), LD @context, subscribed attributes, subscribed Entity Type, filter, representation (*normalized* or *keyValues*) and MIME type required to subscribe to a set of desired Entities. The Node allows also to set up the notification parameters, including the endpoint and attributes notified. Some of the referred parameters can also be provided through the Node’s input payload. This Node enables an easy subscription and notification when a change happens within a certain set of watched Entities of interest to an application.

In the example below a Notification will be sent whenever the air temperature of an *AgriParcel* overpasses a certain threshold.

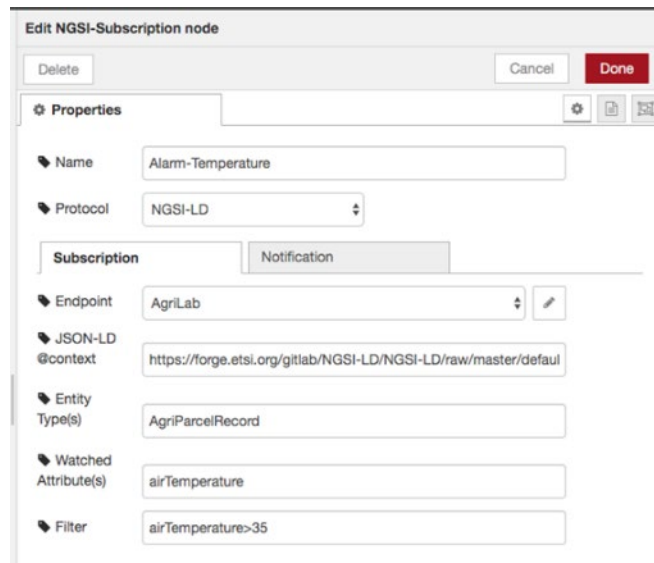


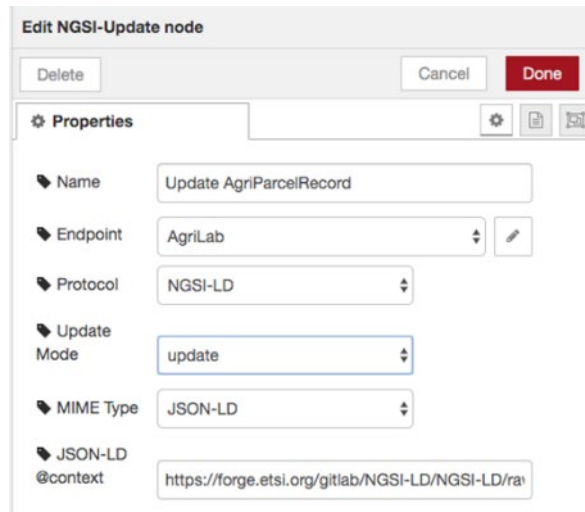
Figure 39: NGSi Subscription Node UI

The *NGSI Update Node* user interface is shown below. It allows to configure an endpoint (defined by a Context Broker Config Node), protocol (NGSIv2 or NGSI-LD), LD @context, and MIME type required to

update a set of desired Entities. The update can be performed under different modes offered by the NGSIv2 and NGSI-LD protocols (*upsert*, update with and without overwrite, etc.).

The Entity representation shall be provided through the Node's input payload. This Node enables an easy update of the data of a set of Entities of interest to an application.

In the example below a configuration, intended to update one or more entities of type *AgriParcelRecord* (for instance with data coming from IoT Devices) is shown.



The screenshot shows the 'Edit NGSI-Update node' configuration window. At the top, there are 'Delete', 'Cancel', and 'Done' buttons. Below is a 'Properties' section with the following fields:

- Name: Update AgriParcelRecord
- Endpoint: AgriLab
- Protocol: NGSI-LD
- Update Mode: update
- MIME Type: JSON-LD
- JSON-LD @context: https://forge.etsi.org/gitlab/NGSI-LD/NGSI-LD/ra/

Figure 40: NGSI Update Node UI

## 9 Appendix B: Step-by-Step Explanation of using CoatRack

To provide a service via Coatrack, the farmer Frank started at the CoatRack homepage as shown in the following Figure 41.

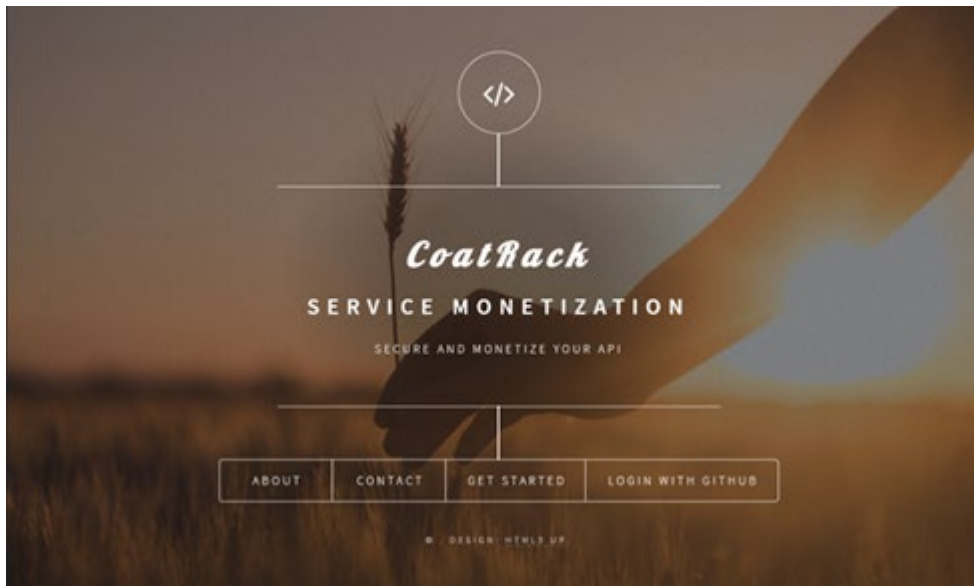


Figure 41: CoatRack Homepage.

The CoatRack homepage is offering information about its usage as well as the possibility to get in contact with ATB and CORIZON. For using CoatRack, a login via a GitHub account is offered. As presented in the following Figure 42, a user can sign in accordingly. In cases, where a user does not have a GitHub account, the registration procedure is offered via GitHub.

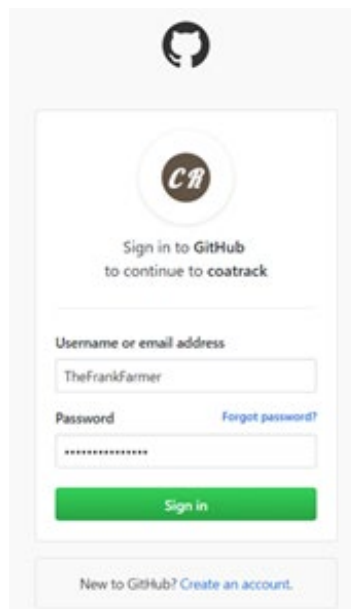


Figure 42: GitHub Account Login.

For being able to manage CoatRack, the user needs to authorise CoatRack to access personal data as presented in the following Figure 43. At the current moment, CoatRack is asking for access to the email address and the profile information.

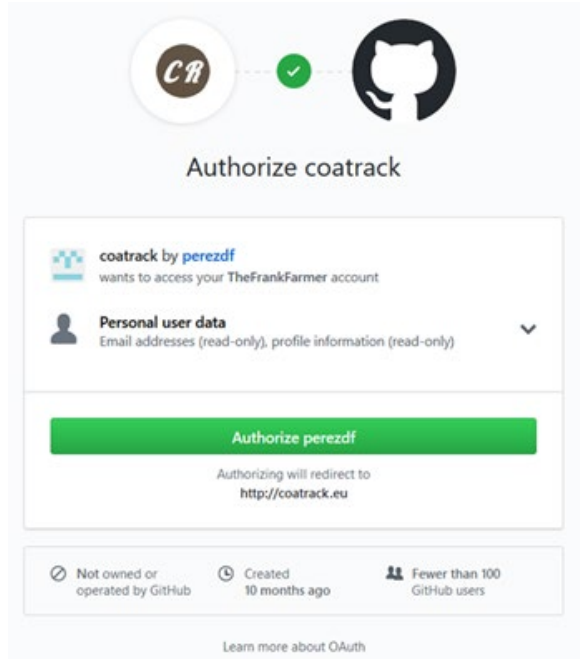


Figure 43: Permissions between GitHub and CoatRack.

After using the authentication procedure based on the GitHub service, the user needs to register in CoatRack itself. The following Figure 44 is presenting the registration form that is asking for the user ID, the full name, company name, and email address.

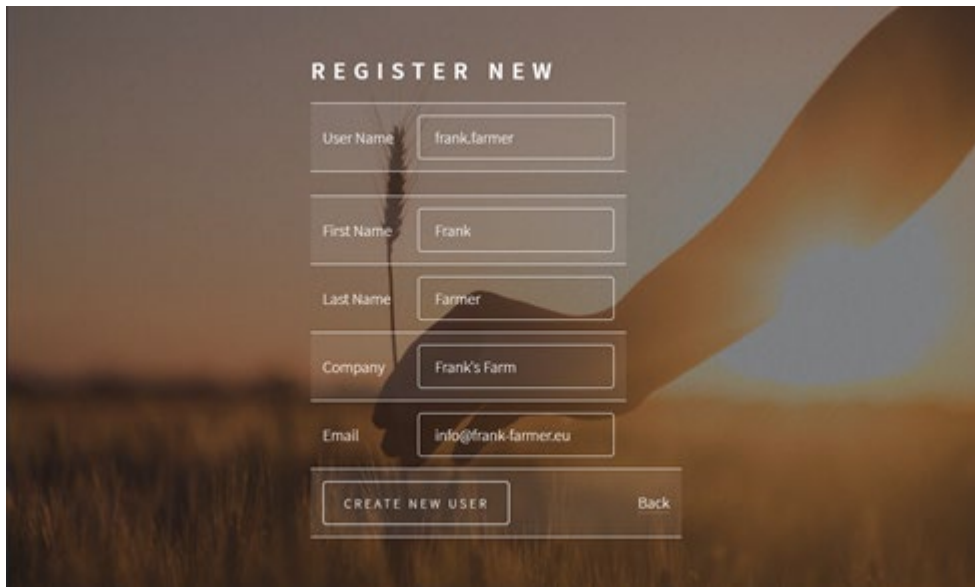


Figure 44: New User Registration.

As soon as the registration is accomplished, the user will see a getting started wizard as presented in the following Figure 45. The user will be guided through the creation and test of the first service. In the example below, Frank the farmer is offering the data collected from the weather stations in the location determined around the coordinates 53.079296 8.801694.



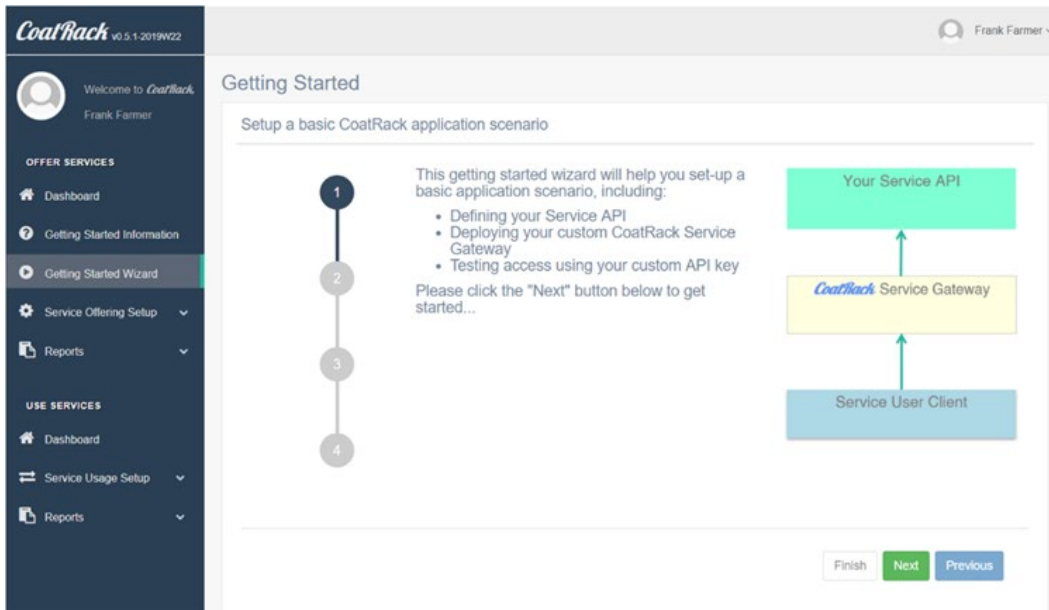


Figure 45: Getting Started Wizard.

The getting started wizard is used to insert the service name, service url and payment policy. The service name should clearly outline the purpose of the service, facilitating also its later administration as well as being logical for a service user like Walter that is searching for weather data at specific locations.

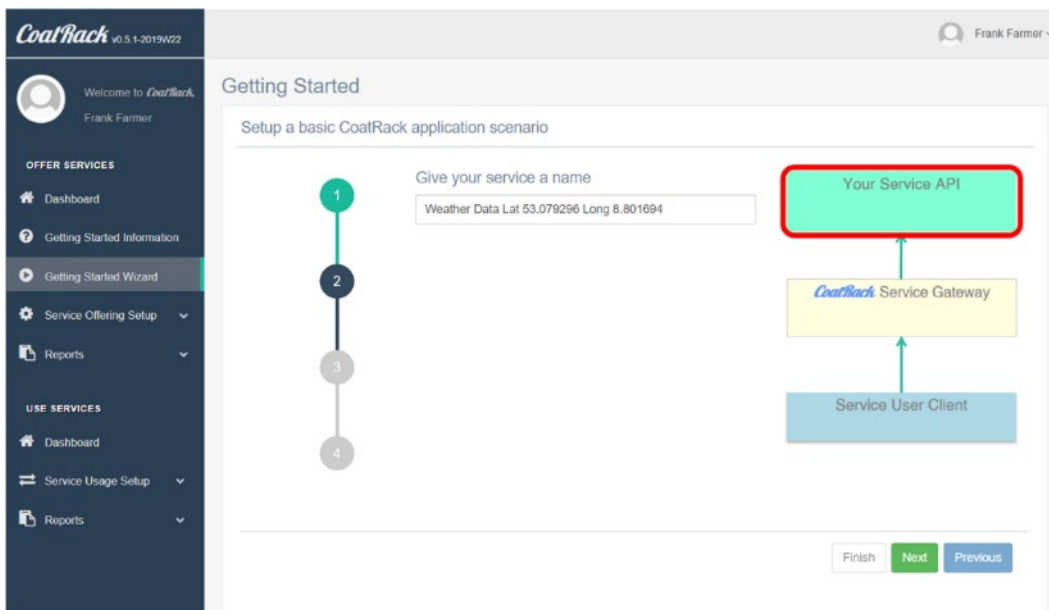


Figure 46: Getting Start Wizard (Service Name)

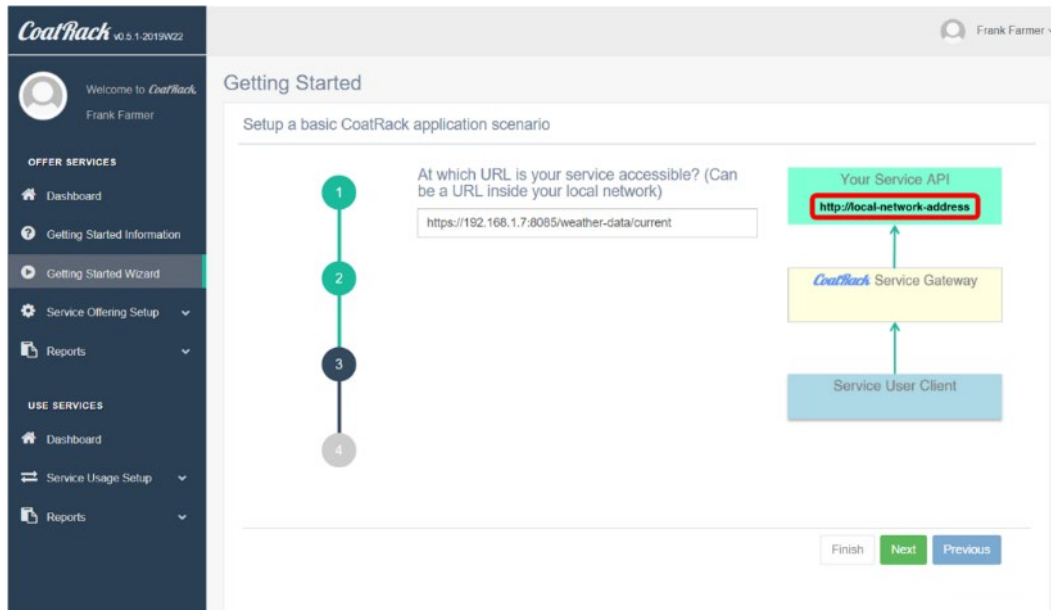


Figure 47: Getting Started Wizard (Service URL).

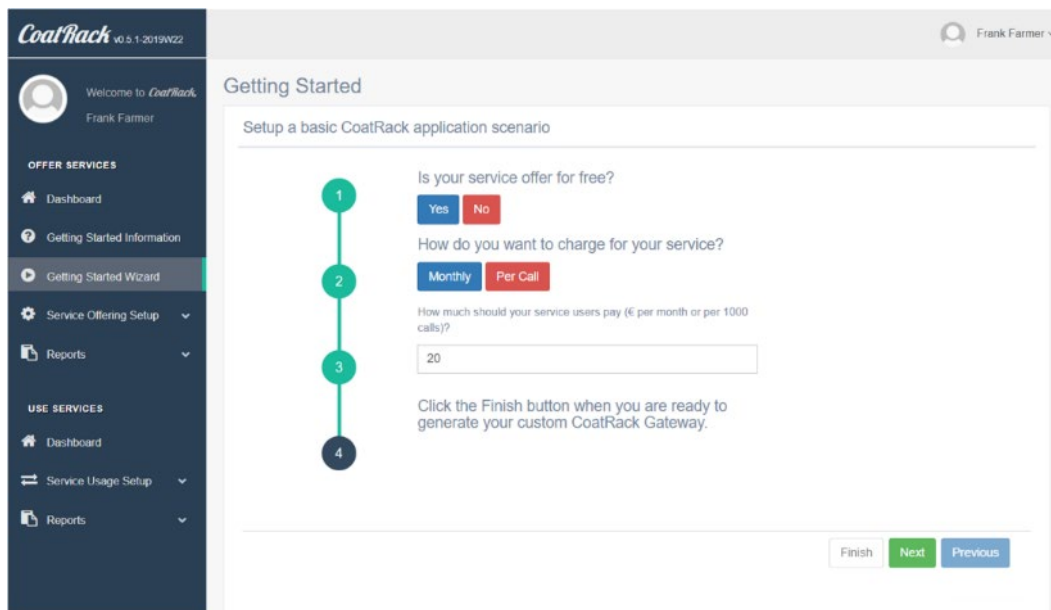


Figure 48: Getting Started Wizard (Payment Policy).

As soon as having registered the first service, CoatRack will show further instructions and an explanation to enable the service provider to test, if the service is working. The first step is to download the gateway, while the page is divided in two parts. The first step is to “Test it” by downloading and installing the Gateway. In the second step “Call it” – the user will get further instruction about how to access the service through the installed gateway. To access the service, an API key is automatically created and shown, as well as the link to be inserted.

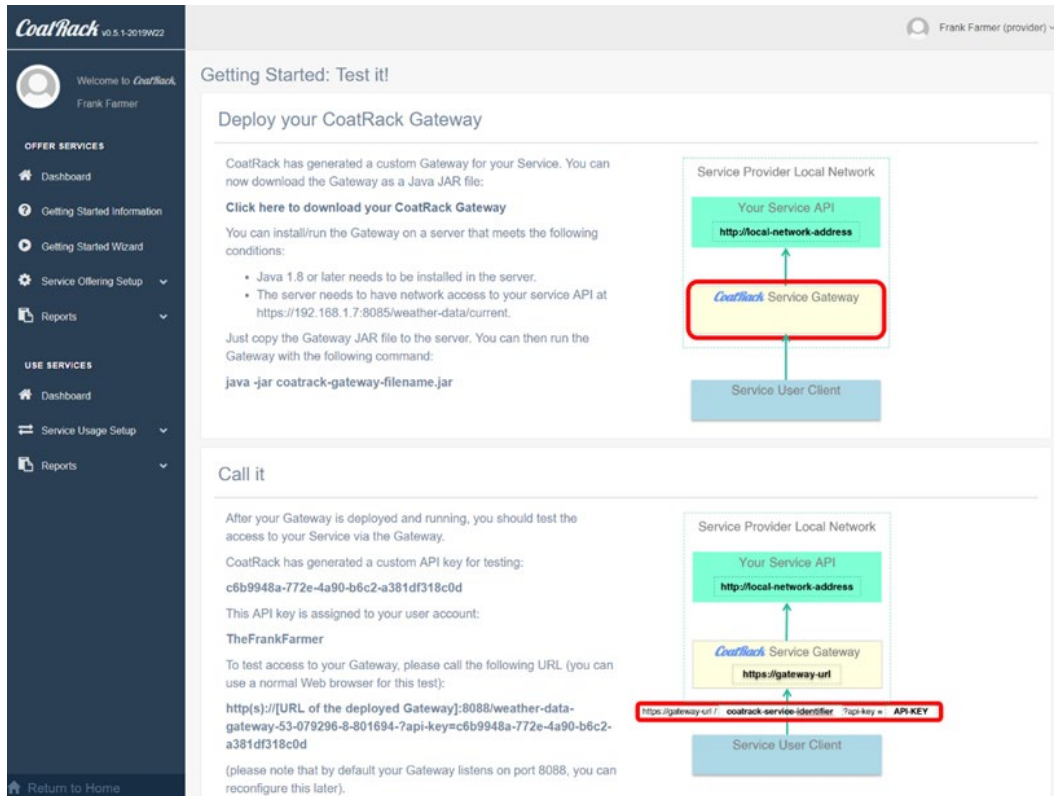


Figure 49: Download and test of the Service Gateway.

After having finalised the download, installation and test of the gateway, the dashboard as presented in the following Figure 50 is presented. The dashboard provides an overview like the number of API calls, users or errors.

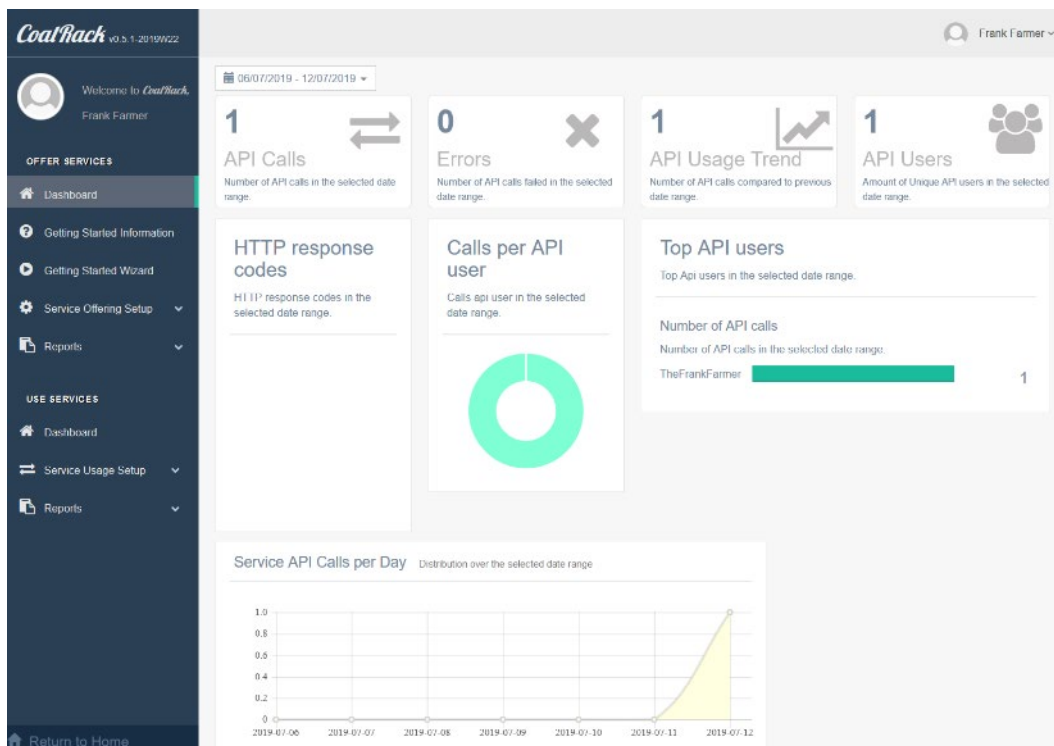


Figure 50: Coatrack Dashboard (Testing Own Service)

Finally, after the service is tested and running, the user needs to set it as a public service, so other interested users can find and start using it. To change the service to public, the user just needs to go to the service offering setup and edit the service. After that, the user just needs to change the setting from restricted to public as shown in Figure 51.

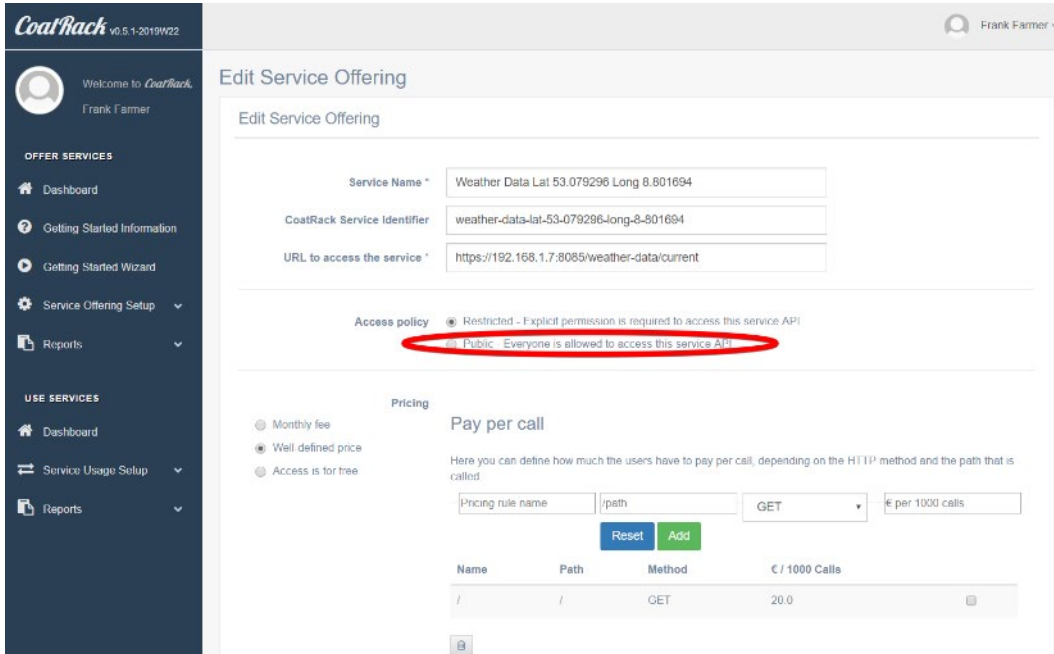


Figure 51: Service Access Policy Update.

After the service is set to public as shown in Figure 52, the service is up and running and ready to be accessed by interested CoatRack users.

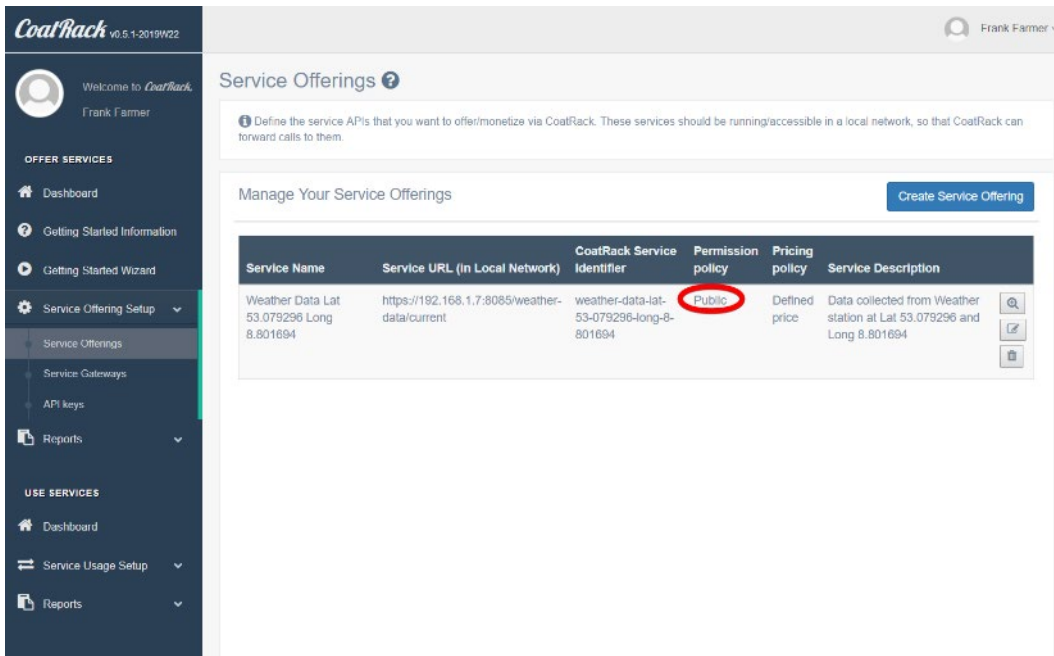


Figure 52: Service Set as Public.

Another user of CoatRack, which is a Weather Forecaster called Walter is interested in using the offered weather station data, especially the weather data provided by Frank. In order to access it, Walter needs to display the public list of services, shown in Figure 53.

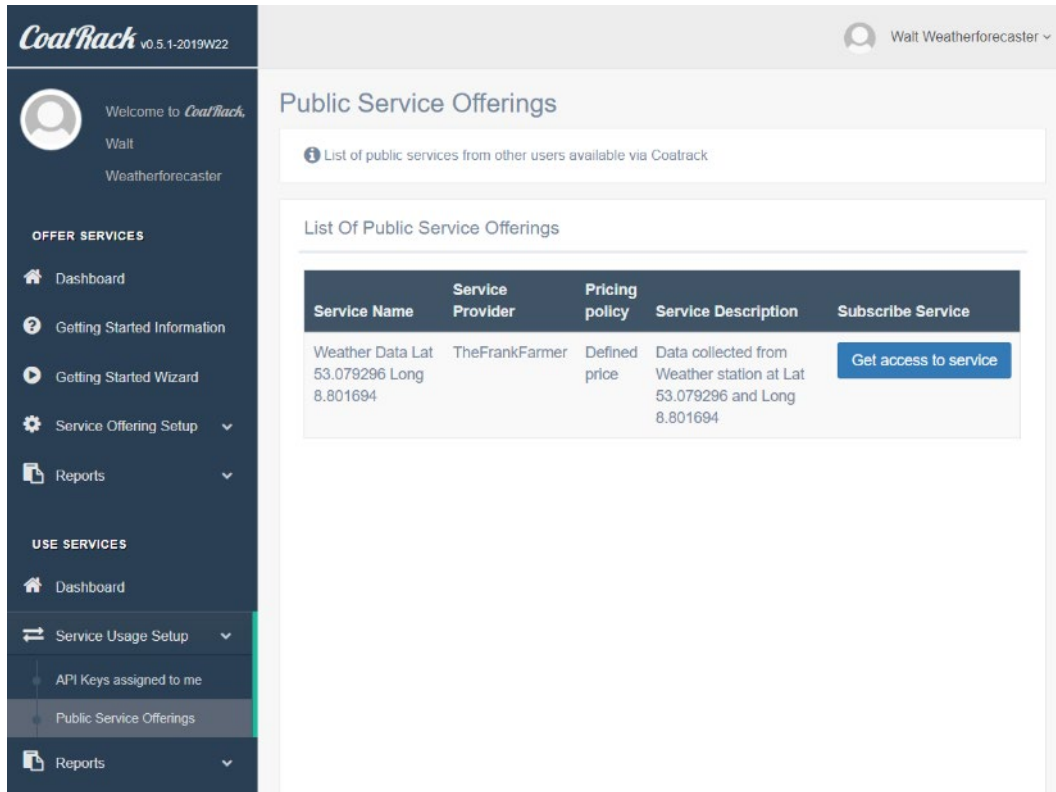


Figure 53: List of Public Services.

For being able to access the service, Walter needs to request it by using the button “Get access to service”. Once done, an alert will pop up, highlighting this as a paid service as presented in the following Figure 54.

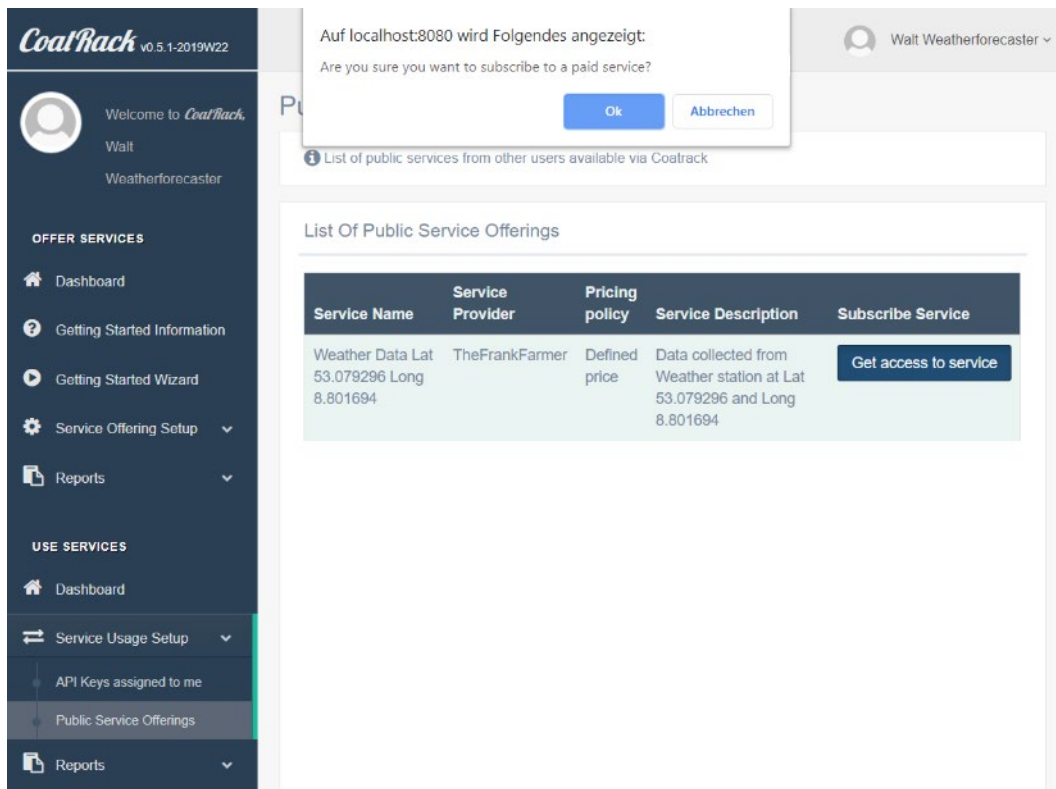


Figure 54: Get access to service button.

After acceptance of the payment policy, an API key was created and can be accessed via the list of public service offerings. As highlighted in Figure 55, also the current status of subscription is listed.



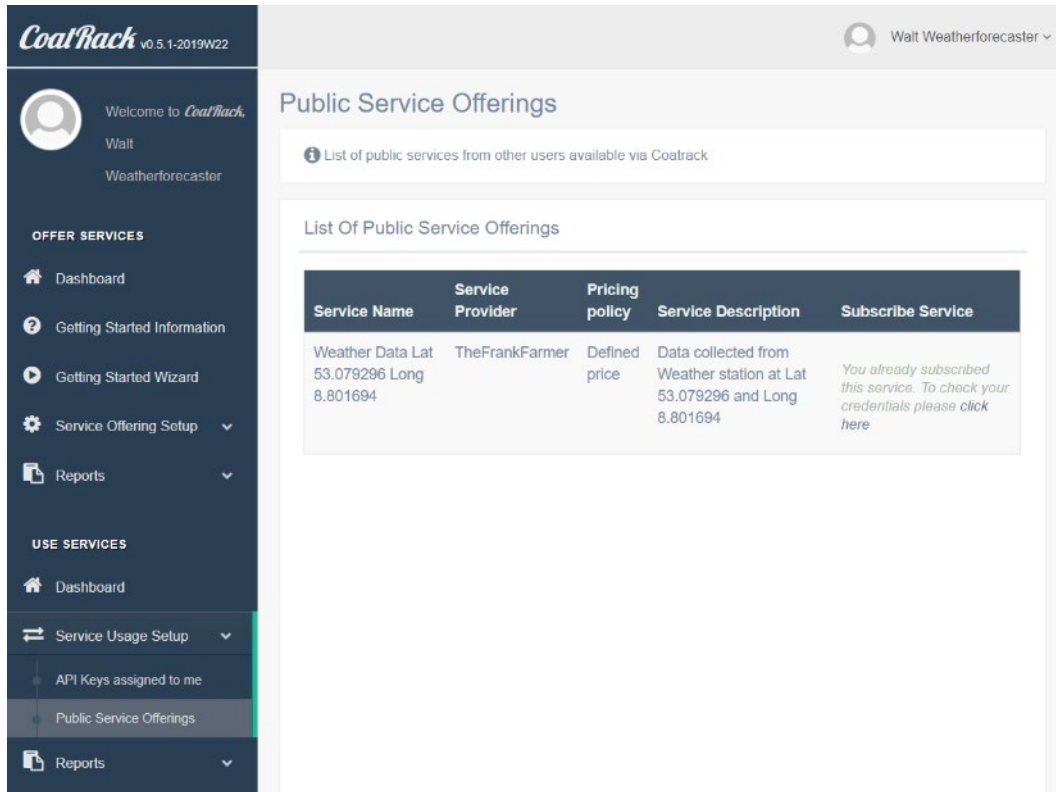


Figure 55: Subscription to Public Service.

When using the displayed link in the subscribed service column as shown in Figure 55, another list will open that is presenting “My API Keys” as shown in the following Figure 56.

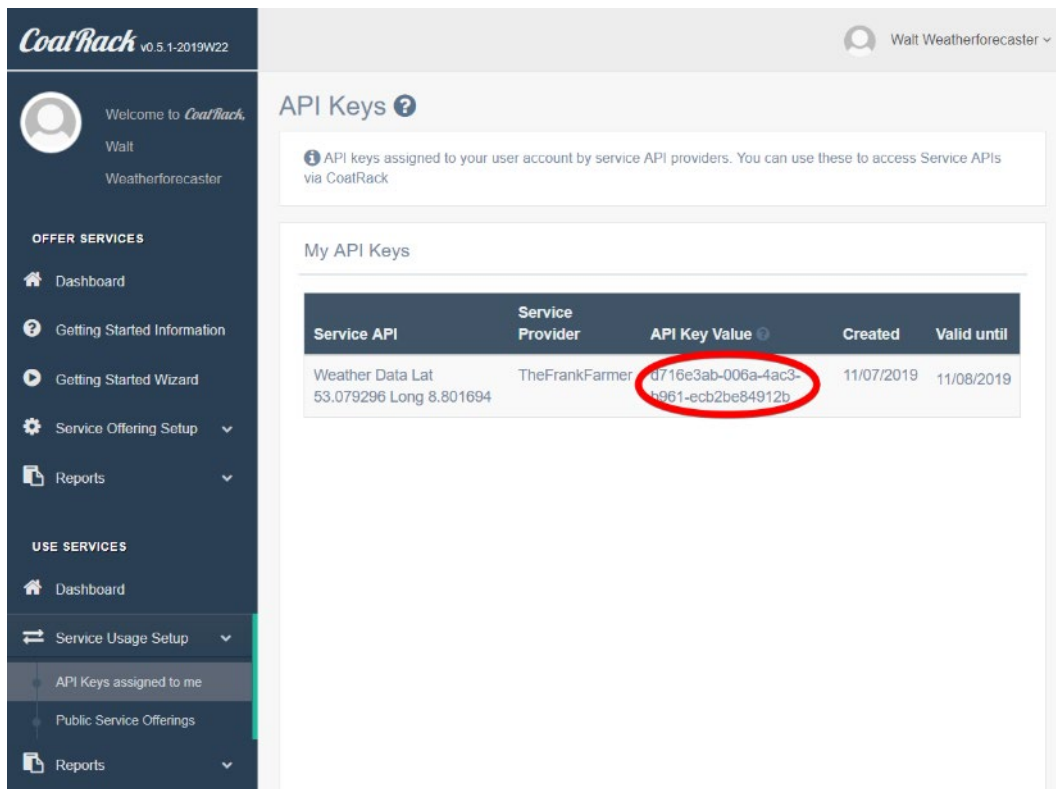


Figure 56: API Keys to access the Services.

Now that Walter has the API Key, he can start using the service provided by Frank. For accessing the service, he just needs to insert the API key and the service identifier in the url, as shown in the following example:

`http:// <public-url> / <coatrack-service-identifier> ?api-key= <apikey>`

The <public-url> is that UTL that was defined by Frank, the <coatrack-service-identifier> is the name of the service as defined in CoatRack and the <apikey> can be taken from the list of “My API Keys”.

The access of the service is tracked by CoatRack and will be immediately displayed in the Dashboard as presented in Figure 57.

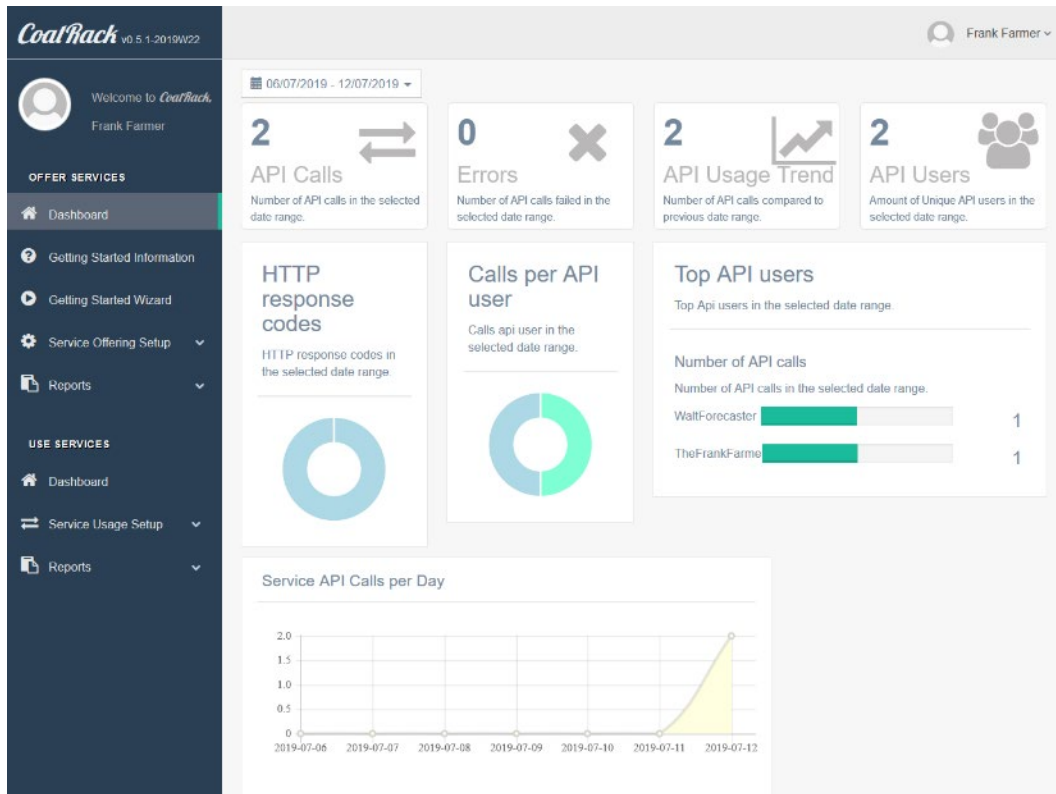


Figure 57: CoatRack Dashboard





INTERNET OF FOOD & FARM