

Project Number: 774571
Start Date of Project: 2017/11/01
Duration: 48 months

Type of document D6.2 – V1.0

SCADA System Integration and Field Validation

| | |
|---------------------|-----------------------|
| Dissemination level | PUBLIC |
| Submission Date | 2021-06-30 |
| Work Package | WP6 |
| Task | T6:2 |
| Type | OTHER |
| Version | 1.0 |
| Author | Emanuele Graziani |
| Approved by | Andrea Gasparri + PMC |

DISCLAIMER:

The sole responsibility for the content of this deliverable lies with the authors. It does not necessarily reflect the opinion of the European Union. Neither the REA nor the European Commission are responsible for any use that may be made of the information contained therein.

Executive Summary

This document aims at providing a description of the SCADA System Integration and Field Validation.

The idea behind the integration process is to compose the system starting from the components developed by the PANTHEON units, applying the bottom-up method. The operating process and data flow drive the integration phase.

In addition, the final section of the document presents the early results based on use of integrated system.

In particular, the document will cover the following topics:

- System Overview;
- System Integration;
- System Infrastructure Deploy;
- Field Validation.

Table of Content

| | | |
|-------|---|----|
| 1 | System Overview | 8 |
| 1.1 | Architecture Overview | 8 |
| 1.1.1 | Wireless Network Backbone | 9 |
| 1.1.2 | GPS-RTK Positioning System | 9 |
| 1.1.3 | RB-SHERPA Robots (UGV) | 10 |
| 1.1.4 | UAV | 11 |
| 1.1.5 | IoT Network | 13 |
| 1.1.6 | DB Architecture..... | 13 |
| 1.1.7 | Central Unit..... | 14 |
| 1.2 | Document Overview | 14 |
| 2 | System Integration..... | 15 |
| 2.1 | Development and integration strategy..... | 15 |
| 2.1.1 | Bottom-up approach..... | 15 |
| 2.1.2 | Publish subscribe middleware | 16 |
| 2.1.3 | Web API | 17 |
| 2.1.4 | Web Socket | 20 |
| 2.1.5 | Building blocks | 21 |
| 2.2 | System Integration Design | 22 |
| 2.2.1 | Integration schema overview | 22 |
| 2.2.2 | Integration schema description | 23 |
| 2.3 | Sub-System Design..... | 26 |
| 2.3.1 | UGV Campaign | 27 |
| 2.3.2 | UAV Campaign | 29 |
| 2.3.3 | Remote Sensing | 31 |
| 2.3.4 | IoT | 33 |
| 2.3.5 | Real-time..... | 35 |
| 2.3.6 | Alarms | 37 |
| 2.3.7 | Historical Data..... | 39 |

| | | |
|--------|---|----|
| 2.3.8 | Suggested Activities | 42 |
| 2.3.9 | Crop Yield Forecasting | 44 |
| 2.3.10 | Map Layer | 46 |
| 2.4 | Design validation..... | 48 |
| 3 | System Infrastructure Deployment | 50 |
| 3.1 | DSP (Data Storage and Processing) Server..... | 51 |
| 3.1.1 | Node-RED..... | 52 |
| 3.1.2 | User Application..... | 54 |
| 3.1.3 | Global Planner..... | 56 |
| 3.1.4 | Pruning Management System..... | 56 |
| 3.1.5 | Processing Chains..... | 57 |
| 3.2 | DCP (Data Collection and Pre-Processing) Server | 57 |
| 3.3 | UGV (Unmanned Ground Vehicle)..... | 60 |
| 3.4 | UAV (Unmanned Aerial Vehicle)..... | 61 |
| 3.4.1 | UAV Equipment..... | 62 |
| 3.4.2 | UAV Post-Processing..... | 63 |
| 4 | Field Validation | 65 |
| 4.1 | Infrastructure Validation..... | 65 |
| 4.1.1 | Wireless Network Backbone..... | 65 |
| 4.1.2 | Ground Robotic Platforms | 65 |
| 4.1.3 | Aerial Robotic Platforms | 66 |
| 4.1.4 | IoT Agrometeorological Network..... | 66 |
| 4.1.5 | Software Architecture..... | 71 |
| 4.2 | Benchmark Objective Validation | 77 |
| 5 | Traceability Matrix..... | 87 |
| 6 | References | 88 |

List of Figures

| | |
|---|----|
| Figure 1. Bottom-up strategy. | 16 |
| Figure 2. Publish/subscribe schema. | 17 |
| Figure 3. Web API architecture. | 19 |
| Figure 4. HTTP vs WebSocket communication comparison. | 20 |
| Figure 5. Integration schema overview. | 22 |
| Figure 6. Integration schemes legend. | 23 |
| Figure 7. UGV Campaign subsystem integration scheme. | 28 |
| Figure 8. UAV Campaign subsystem integration scheme. | 30 |
| Figure 9. The UAV Processing, integration scheme. | 32 |
| Figure 10. The UGV Processing, integration scheme. | 32 |
| Figure 11. IoT subsystem integration scheme. | 34 |
| Figure 12. Real-Time subsystem, integration scheme. | 36 |
| Figure 13. Alarm's subsystem, integration scheme. | 38 |
| Figure 14. Historical Data subsystem, integration scheme. | 40 |
| Figure 15. Suggested Activities subsystem, integration scheme. | 43 |
| Figure 16. Crop Yield Forecasting subsystem, integration scheme. | 45 |
| Figure 17. Map Layer subsystem, integration scheme. | 47 |
| Figure 18. System infrastructure integration overview. | 50 |
| Figure 19. Node-RED flow-based programming tools. | 52 |
| Figure 20. Node-RED development console. | 53 |
| Figure 21. PANTHEON integration flows example. | 54 |
| Figure 22. User application architecture. | 55 |
| Figure 23. A representative campaign folder. | 57 |
| Figure 24. DCP deployed infrastructure. | 58 |
| Figure 25. Raspberry board. | 59 |
| Figure 26. Raspberry Pi4 Model B board, ROS gateway. | 59 |
| Figure 27. SHERPA HL robotic platform R-A outer parts. | 60 |
| Figure 28. Components of the UAV subsystem. | 62 |
| Figure 29. Wireless connections in the UAV subsystem. | 63 |
| Figure 30. Schema of the connections between sensors, flight controller and portable device. | 63 |
| Figure 31. Flight data postprocessing. | 64 |
| Figure 32. "Save Measurements" Node-RED flow. | 66 |
| Figure 33. "Save weather measurements" Node-RED flow. | 67 |
| Figure 34. "Save soil measurements" Node-RED flow. | 67 |
| Figure 35. "rawMeasurements" MongoDB collection. | 68 |
| Figure 36. "measurements" MongoDB collection. | 69 |
| Figure 37. IoT real-time data on user interface. | 70 |
| Figure 38. "Soil sensors" historical data, Node-RED flow. | 71 |
| Figure 39. "historicalData" MongoDB collection. | 72 |
| Figure 40. User interface, historical air data. | 73 |
| Figure 41. User interface, historical rain data. | 73 |
| Figure 42. User interface, historical soil data. | 74 |
| Figure 43. Sub-zero pre alarm flow. | 75 |
| Figure 44. MongoDB "Alarms" collection. | 76 |
| Figure 45. User interface, alarm notification. | 76 |
| Figure 46. User interface, alarms table. | 76 |
| Figure 47. SLAM system recognizing a dynamic obstacle. | 77 |
| Figure 48. Comparison between old (left) and new (right) spray nozzles. | 78 |
| Figure 49. Planned flight for a coverage of 1 ha. | 79 |

| | |
|---|-----------|
| <i>Figure 50. Examples of detected true bugs. Confidence scores are reported. False positives are highlighted with red x-marks, while false negatives with red circle.....</i> | <i>81</i> |
| <i>Figure 51. Detection of a sucker with confidence level reported.</i> | <i>82</i> |
| <i>Figure 52. Sucker 3D mesh reconstruction. Surface area is equal to 0.8m².</i> | <i>83</i> |
| <i>Figure 53. Graphical representation for a symmetric type of tree for the automatic pruning system as outcome of the pruning algorithm (left) and as shown in the user interface deployed on a smart device (right).</i> | <i>84</i> |

List of Tables

| | |
|---|-----------|
| <i>Table 1. Web API HTTP verbs mapping.</i> | <i>18</i> |
| <i>Table 2. Summary of the subsystems involved in the integration process.</i> | <i>22</i> |
| <i>Table 3. Package’s description of the overview schema.</i> | <i>26</i> |
| <i>Table 4. Description of common Node-RED nodes used.</i> | <i>53</i> |
| <i>Table 5. Requirement’s traceability matrix.</i> | <i>87</i> |

Abbreviations and Acronyms

| | |
|---------|--|
| API | Application Programming Interface |
| CRUD | Create, Read, Update, Delete functions |
| CSV | Comma-Separated Values |
| DoA | Description of Actions |
| DCP | Data Collection and Pre-processing |
| DSM | Digital Surface Model |
| DSP | Data Storage and Processing |
| DB | Database |
| GeoJSON | Geographic JSON |
| GeoTIFF | Geospatial TIFF |
| GEXF | Graph Exchange XML Format |
| GNSS | Global Navigation Satellite System |
| GPS | Global Positioning System |
| IoT | Internet of Things |
| JSON | JavaScript Object Notation |
| LoRa | Long Range |
| LTS | Long Term Support |
| NDVI | Normalized Difference Vegetation Index |
| PLY | Polygon File Format |
| PTX | Plain Text data format |
| RGB | Red-Green-Blue color model |
| REST | Representational State Transfer |
| ROS | Robot Operating System |
| RTK | Real Time Kinematic |
| SCADA | Supervisory Control and Data Acquisition |
| SDK | Software Development Kit |
| SQL | Structured Query Language |
| TIFF | Tagged Image File Format |
| UAV | Unmanned Aerial Vehicle |
| UGV | Unmanned Ground Vehicle |
| UI | User interface |
| URL | Uniform Resource Locator |
| VPN | Virtual Private Network |
| WNB | Wireless Network Backbone |
| WP | Work Package |
| XML | eXtensible Markup Language |
| YOLO | You Only Look Once |
| RGBD | Red-Green-Blue-Depth model |

1 System Overview

The PANTHEON project [1] aims to develop a working prototype of a SCADA-like precision farming system able to operate in a real-world (1:1 scale) orchard. It is important to remark that, even though all orchards have several features in common, the specific farming operations to be performed and the way the sensory data should be collected and processed might differ from crop to crop.

We expect that the proposed SCADA system will be able to acquire information with the resolution of the single plant. This will allow to dramatically improve the detection of possible limiting factors for each individual plant, such as lack of water or presence of pests and diseases. This will of course then be reflected in more appropriate intervention measures.

More specifically, the proposed SCADA system is composed of the following main components which will be detailed in the following:

- Wireless Backbone Network;
- GPS-RTK Positioning System;
- UGV (RB-SHERPA Robots);
- UAV (DJI Matrice 600 Pro with RTK);
- IoT (Internet of Things) Network;
- DB Architecture;
- Central Unit.

1.1 Architecture Overview

The whole system integration process is based on the defined architecture. Each partner of the PANTHEON consortium, with a technological development role, has been assigned one or more subsystems identified in architectural design.

Once the development guidelines and the interconnection interfaces had been defined, each team was able to autonomously develop the modules of the PANTHEON SCADA System.

Follow, a summary is provided about the main components of the SCADA system, as reported in Deliverable D2.1 [2]

1.1.1 Wireless Network Backbone

The Wireless Network Backbone (WNB) is the infrastructure required to keep all the components of the SCADA system interconnected, from the central unit housing the Database (DB) to the single robots moving in the field. The selected WNB architecture is based on a set of mesh antennas and two long-distance antennas. The former is required to create a mesh network on the field, so that UGVs and UAVs can operate in the field itself. The two long-distance antennas are required to connect the central unit, stored in a remote warehouse, to the mesh network deployed in the field. For the sake of easy hardware and software integration, we decided to rely on commercial Wi-Fi solutions based on the 802.11 standard, at either 2.4GHz or 5.0GHz.

The WNB consists of eight antennas and one router, specifically:

- 2 AIRMAX antennas (LiteBeam AC GEN2);
- 6 UNIFI antennas (AC MESH PRO);
- 1 router (IR615-S-L3-WLAN).

The router is the main component, and it must be placed inside the control room, i.e., an area dedicated to the project to be defined within the facilities of the farm. The first AIRMAX antenna is wired connected, i.e., through an Ethernet cable, to the router and orientated to the second antenna placed in the field. This allows creating a connection from the control room to the mesh network deployed in the field.

Regarding the mesh network, the first UNIFI antenna is wired connected to the second AIRMAX antenna. Any other UNIFI antenna is wireless connected, i.e., through an uplink-downlink radio connection as specified by the UNIFI Protocol, to create a mesh network. In this way, the signal is re-broadcasted through a desired network topology pre-configured from a UNIFI Controller, permitting the connection between all devices in the field.

The AIRMAX LiteBeam AC GEN2 is an ultra-lightweight airMAX ac CPE device, that can provide 23 dBi of gain for long-distance connectivity and which uses a directional antenna pattern for improved noise immunity.

The UNIFI AC MESH PRO is an access point wireless dual-band 802.11AC 3x3 MIMO with two Gigabit Ethernet port and an integrated omni-directional antenna, ideal to cover ample external zones.

1.1.2 GPS-RTK Positioning System

The GPS-RTK positioning system provides accurate location information. Indeed, this capability is required to collect geo-referenced data, such as images or 3D laser cloud-points. Furthermore, it facilitates the development of safe and autonomous guidance algorithms for the unmanned vehicles. The selected GPS-

RTK positioning system relies on the availability of the WNB infrastructure for transmitting GPS corrections over the field.

The GPS-RTK positioning system consists of the following elements:

- 1 base station, Trimble BX982 GNSS receiver;
- 3 GPS antennas, GPS Zephyr (one for base and one for each robot);
- 2 GPS module Trimble MB-Two (one for each robot).

The base station should be connected to the WNB to transmits GPS L1/L2/L5 and GLONASS L1/L2/L3 signal corrections. After receiving these corrections, the GPS module mounted on-board each robot applies them to an initial localization provided by the GPS antenna, generating a final more accurate localization.

The GPS module provides faster dual frequency-based heading acquisition and an improved RTK/PPP positioning engine. In this way it permits the use of use a Global Navigation Satellite System (GNSS) configuration with dual antenna/frequency (GPS, QZSS, GLONASS, Beidou, Galileo). In addition, Ashtech's patented Z-Blade technology drives a powerful GNSS agnostic engine, allowing the GPS module to use any combination of GNSS system for positioning. The Trimble BX982 receiver enclosure is a multi-channel, multifrequency GNSS receiver, it is suited for applications that require precise heading and attitude information in addition to position. This combination of technologies, using L1 and L2 signals, allows reaching the following specifics:

- Accuracy (High-resolution mass spectrometry): < 8 mm + 1 ppm;
- Initialization time: < 1 min typical;
- Operating range: > 40 km.

1.1.3 RB-SHERPA Robots (UGV)

Two UGVs are required for the activities of the projects, namely robot A (R-A) and robot B (R-B). Both robots have the same sensorial equipment and share a common localization, safety, and navigation system.

In particular, the following onboard sensors are mounted on both robots: high accuracy gyro, laser scanners (SICK 300), GPS module (Trimble MB-Two), and 3D Lidar (Velodyne).

The main task of R-A is to collect sensorial data for tree geometry reconstruction, for the assessment of the phytosanitary status of the plants and to mark branches for pruning.

The following onboard sensors/devices are mounted only on R-A to facilitate activities in the field:

- 3D Laser scanner (Lidar);

- Sony a5100;
- RedEdge-M;
- DJI Ronin MX.

The 3D Lidar scanner provides 360°, 3D distance and calibrated reflectivity measurements allowing a graphical reconstruction of trees geometry at a later phase. The Sony a5100 provides High resolution RGB Images. The RedEdge-M sensor generates Chlorophyll maps, NDVI maps, Digital Surface Models (DSMs), and RGB Images. The Chlorophyll map is obtained also using the red edge spectral band, that works in conjunction with the other bands to provide a measure of plant vigour and health. The NDVI maps is obtained comparing the reflectance of the red band with that of the near-infrared band. A DSM is a product used to evaluating surface properties and water flow. The RGB Image is processed aligning all visible bands. The DJI Ronin MX is required to facilitate target aiming on scanning and marking phases.

The main task of R-B is to apply chemicals on suckers with the scope to remove them and any related features. For this purpose, an agricultural atomizer has been integrated with electrical driven pump. The sprayer is mounted on a Pan-Tilt mechanism to facilitate target shooting. This Pan-Tilt mechanism is based on Dynamixel motors.

Two different kinematics models were chosen for the two robots, according to the specific applications to be carried out:

- Omnidirectional, 4x Multidirectional wheels, on R-A;
- Ackerman, 2x Multidirectional wheels, on R-B.

In addition, each robot has a Wireless Emergency Stop for safety reasons, i.e., a button which can stop the robot's engines, in case of malfunction.

From a software standpoint, every main component of each robot is integrated and managed within the ROS environment, thus making it easier to manage, control, and save to the DB infrastructure all real-time data captured by robots or IoT sensors.

1.1.4 UAV

The selected aerial platform is the DJI M600 Pro with RTK which was customized to perform hazelnut remote sensing using high level sensors. The reasons for this choice are here listed:

- Affordability;
- Robustness of the platform;

- Widespread technology: in particular, the fact that this platform is becoming a state-of-the-art solution all around the world, which will potentially maximize the impact of the developments carried out in the PANTHEON project.

The robotic platform consists of five functional elements:

- DJI M600 PRO drone;
- DJI RTK system;
- DJI Manifold;
- DJI Ronin MX;
- The payload.

The Matrice 600 Pro is a six-rotor flying platform designed for professional aerial photography and industrial applications. This model is equipped with a DJI A3 Pro triple-modular redundancy system and advanced intelligent flight functions. The A3 Pro flight controller provides three GPS modules and IMUs which add triple modular redundancy to reduce the risk of system failure.

This system is complemented with an RTK module which, using a ground station, provides corrected GPS signals to improve its accuracy. To control the high-level behaviour of the drone, this will be augmented with a DJI Manifold added on the top of the aircraft.

The DJI Manifold is connected with all the components of the system and with a long-range high-speed communication radio that communicates with the rest of the SCADA system.

A gimbal Ronin MX has been installed to host, orient, and stabilize the payload. The gimbal Ronin MX has its own intelligent battery and compensation system. The performance of this gimbal can be also controlled either with the DJI radio controller or with the DJI manifold computer.

The payload consists of 3 different sensors attached to the gimbal, specifically:

- 1 multispectral camera Tetracam MCAW 6 + filter;
- 1 thermal camera ThermalCapture 2.0 640;
- 1 RGB camera Sony α 5100.

The functional specifications for the overall system are the following. The system must be capable of navigate autonomously on the orchard with a precision of less than 15 cm. The system must be capable of triggering all the camera of the payload, if needed in a synchronized way, and to record all the relevant telemetry data of the UAV and of the gimbal at the moment of the triggering. The system must be provided with a path

planner able to design the mission (trajectories to be followed and triggering point) and to execute it autonomously. The system must be compliant with Italian Laws and Rules on Unmanned Aerial System.

1.1.5 IoT Network

An IoT-based agro-meteorological monitoring network, based on the new LoRa communication protocol, is deployed in the field. It consists of the following modules:

- 1 station for the survey of meteorological data;
- 9 LoRa nodes to record humidity and temperature data of the soil;
- 1 LoRa/RoS Gateway of the network.

The modules of the network collect data from the sensors at a desired rate and send them to the Gateway, which is responsible for converting data into the ROS standard for storage in the primary DB server. The gateway interfaces the LoRa network with the WNB, creating a convenient decoupling between the IoT network and the ROS network infrastructure.

The meteorological station acquires several environmental variables: precipitation, wind direction and wind speed, air temperature, relative humidity, air pressure and solar radiation. The station is powered by a 12V - 7.2A battery and a 10W solar panel, which allows continuous data collection.

LoRa nodes represent peripheral units installed in the field for the acquisition of high-resolution soil moisture and temperature data, which will be collected at two depths, using capacitive SDI12 sensors. Nodes are based on a Teensy microcontroller that uses a 72 MHz Cortex-M4 processor and an RF transceiver module RFM95W that features an LoRaTM long range modem at 868 MHz. In particular, such modem provides ultra-long range spread spectrum communication and high interference immunity while minimizing current consumption. Nodes will have an estimated consumption of about 20-30 mA and will be powered by a 7.2V lithium battery at 6800 mA and a 5W solar panel. Thanks to high gain antennas and to the installation on special poles, the nodes will have a stable communication range of a few kilometres.

The Gateway is based on Raspberry Pi 3 microcontroller, a LoRa RFM95W module and a high gain antenna for receiving data sent from the nodes. A RoS node will be implemented on the Gateway to allow communication between the LoRa network and the Mesh Network of the WNB.

1.1.6 DB Architecture

DB Architecture is composed of 2 DB servers: a primary DB server and a secondary DB server. The primary DB server is placed on the control room near the field, and its main task is to store real-time data provided

by the IoT network, through a dedicated ROS node. In place to the server, on first stage, one or more gateways take care of the pre-processing and data transmission task.

The secondary DB server is positioned at the facilities of UNIROMA3, and its main task is to allow storage and computation capabilities for analyzing all data collected by the UAV/UAGs and by the IoT network. In particular, we decided not to rely on the ROS network for storing large data, such as 3D Lidar point clouds, coming from the sensors onboard the UAV/UGV. This decision has been made considering the following two factors: size of the data and the fact that the data will not be collected continuously over time (as per the IoT network). In first stage, the use of virtual machine is evaluated.

The PANTHEON project will rely on NoSQL database technology for the realization of the DB architecture.

1.1.7 Central Unit

The central unit is the workstation where the main software components run. This includes the ROS core, the primary DB server, and the support decision system. This workstation is installed on the control room, it runs a Linux-based system, and it is the main point of control of the SCADA architecture. The central unit is interconnected with the server at the UNIROMA3's Robotics Lab. This allows, for instance, the availability to the central unit located in the farm of the results of the computational-intensive tasks to be carried out in the main, more powerful workstation available at the UNIROMA3 facilities, where the secondary DB server is located.

1.2 Document Overview

The document consists of following sections:

- Section 1 provides an overview of the system architecture, and this document structure;
- Section 2 describes the system integration strategy and design;
- Section 3 describes the system infrastructure deployment;
- Section 4 provides field validation report;
- Section 5 provide the traceability matrix;
- Section 6 provides the document bibliography.

2 System Integration

The PANTHEON project involves the implementation of an equivalent SCADA system in the field of hazelnut cultivation. The process of controlling agronomic activities requires the ability to aggregate a series of activities ranging from data collection, processing, saving, transmission, and use.

Like any complex system, during development planning, the functionalities to be developed were assigned into limited thematic areas. In this way, subsystems were naturally created. Each subsystem is unique in terms of its role within the overall scenario and, moreover, it requires specific expertise to be developed.

On this basis, work packages were identified in order to be developed and assigned to each unit of the consortium according to their expertise in the field. From all this, it follows that the final step is system integration.

The System Integration allows to aggregate all the developed subsystems, as planned, allowing the correct exchange of information and recreating the flow of functionality expected for the original SCADA system.

In the next paragraphs, there will be presented the details of the integration activity, specifically:

- The strategies implemented for System development and integration;
- The analysis and design of the high-level integration process;
- The analysis and design of the detailed integration process relating to each subsystem.

2.1 Development and integration strategy

The following sections describe some fundamental approaches adopted during the planning and execution of the integration process.

2.1.1 Bottom-up approach

From its first conception, the PANTHEON project has brought with it a natural approach towards a bottom-up development methodology. In fact, the specific thematic areas of research and development are well identified from the beginning, such as the implementation of the UGV and the UAV, the processing of collected data, the database management, the user interface. In addition, the skills of each unit of the consortium were well defined from the start, to assign related tasks to each one.

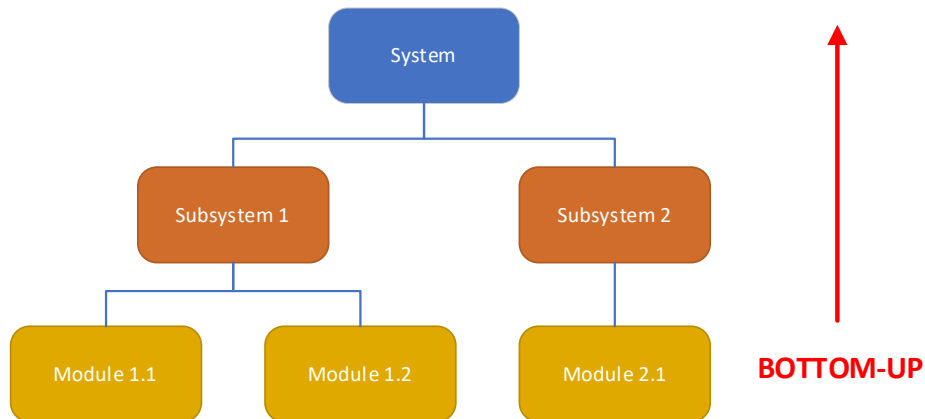


Figure 1. Bottom-up strategy.

The bottom-up approach, depicted in Figure 1, involves first completing the development of each of all the modules and subsystems, performing unit tests to verify correct operation and then proceeding with integration with the overlying system according to the interfaces defined in phase of analysis and design.

This design and development approach naturally affects the type of tests performed, as the subsystems tests are completed in the first phase and in the final phase group tests are performed to verify the operation of the entire system.

Among the advantages of this approach there is certainly the one that will integrate a fully developed and tested component from time to time, avoiding possible errors resulting from the use of many stubs as is the case with the top-down approach.

Another advantage is the reduction of risk as the development of critical modules is addressed in the early stages of the project rather than in the latter.

2.1.2 Publish subscribe middleware

Publish / subscribe is a design pattern used for asynchronous communication between processes or elements of an information system. Due to its nature, it can be considered a communication middleware.

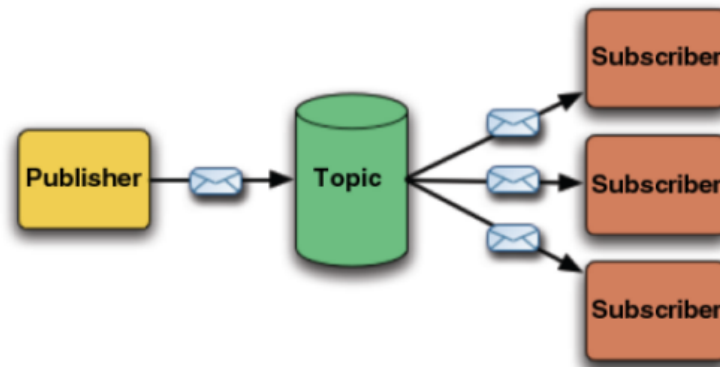


Figure 2. Publish/subscribe schema.

The communication scheme (exemplified in Figure 2) provides for the presence of a subject who manages the exchange of messages, called `dispatcher` or `broker`, while the sender and recipient are called `publisher` and `subscriber`. The mechanism provides for a strong decoupling between publication and receipt of messages, in fact, publishers do not know how many and which subscribers are and vice versa. This mechanism contributes to the scalability of the system.

In several components of the PANTHEON system, this pattern has been used as an integration strategy. For example, the UGV control system is based on the ROS environment, which uses a `Topics` structure for data communication and a `publish/subscribe` exchange protocol.

The use of this middleware allows to clearly define the data exchange interface that must respect the subsystems under development. Moreover, the middleware implies an independent operation of the various subsystems, so that the possibility of developing the various modules independently is guaranteed, as required in the bottom-up approach.

2.1.3 Web API

A Web API is a set of features offered by a Web service that are exposed to other products or services and that allow communication and access to resources without having to provide implementation details.

A Web API makes its functions available through endpoints accessed via the `HTTP` protocol and through which, systems exchange request-response messages with predefined specifications using the `JSON` or `XML` format for the exchange of such data, either in the request phase and in the response phase.

Inspired by the `REST` model, the Web APIs implemented for the project have the following characteristics:

- the available resources are identified and mapped on the exposed URLs;

- the operations on these resources have been associated with the verbs of the HTTP protocol (POST, GET, PUT, DELETE): with the GET method you will be asked to read data, with POST to create new ones, with PUT to modify them and with the DELETE to delete them.

Below, in the Table 1, the recommended return values of the primary HTTP methods in combination with the resource URIs are summarized.

| HTTP Verb | CRUD | Entire Collection (e.g., sensors) | Specific Item (e.g., /sensors/{id}) |
|-----------|-------------------|--|--|
| POST | Create | 201 (Created), 'Location' header with link to /sensors/{id} containing new ID. | 404 (Not Found), 409 (Conflict) if resource already exists. |
| GET | Read | 200 (OK), list of sensors. Use pagination, sorting, and filtering to navigate big lists. | 200 (OK), single sensor. 404 (Not Found), if ID not found or invalid. |
| PUT | Update Replace | 405 (Method Not Allowed), unless you want to update/replace every resource in the entire collection. | 200 (OK) or 204 (No Content). 404 (Not Found), if ID not found or invalid. |
| DELETE | Delete | 405 (Method Not Allowed), unless you want to delete the whole collection, not often desirable. | 200 (OK). 404 (Not Found), if ID not found or invalid. |

Table 1. Web API HTTP verbs mapping.

The URL and the body of the request specify the parameters that identify both the resource to be accessed and additional data to be exchanged with the server, in JSON format.

The greatest advantage of using the Web API is that they allow you to offer services to other products without specifying how they are implemented, thus simplifying application development and integration, as depicted in Figure 3.

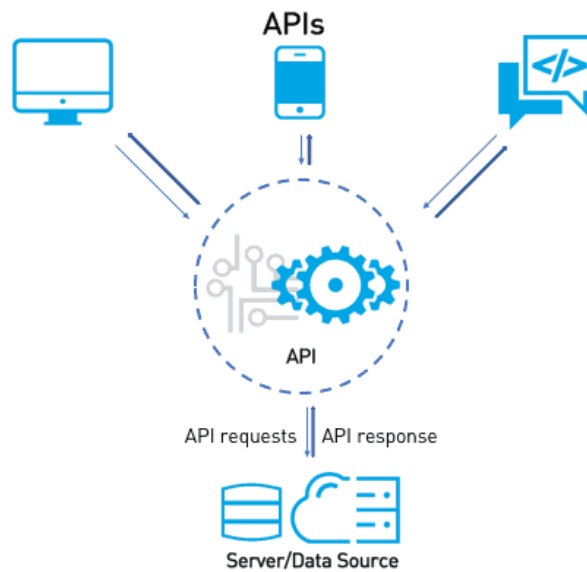


Figure 3. Web API architecture.

In the PANTHEON project, the Web APIs were developed in a back office created with the use of the MEAN stack, a free and open-source JavaScript solution stack, using the TypeScript language via `node.js` and the express library, thus creating a Web API model based on HTTP and JSON format.

The services offered by the API are mainly aimed at:

- interaction with the DSP back-end
 - notification of the arrival of new measurements;
 - notification of a new alarm.
- interaction with the front-end of the user application
 - user management;
 - extraction of data from the database, modification and insertion for various types of resources (measurement, geoObject, activity, comment, mission, chemical Products, damages, alarm, campaign, ...).

Furthermore, in the project, through the front-end of the user application, third-party Web APIs are used to exploit services such as:

- GoogleMaps API for map display and Geolocation API for user tracking;
- Yahoo Weather API for viewing weather conditions and forecasts.

2.1.4 Web Socket

For some types of interactions, the HTTP protocol is not optimal as it is designed to be unidirectional and therefore data transmission is only allowed in one direction at a time (request / response). For situations in which a continuous and real-time updating of data is required, other techniques must be used, in particular in this case, the WebSocket protocol has been chosen to be used.

The WebSocket protocol allows interaction between Client and Server with less overhead than HTTP, which should employ a polling mechanism to receive data. By opening a WebSocket, the server will be able to send data to the client without the latter sending continuous requests.

In a scenario in which a Client will need to receive data in real time, adopting this way, the server will immediately communicate if there is new useful information (see Figure 4).

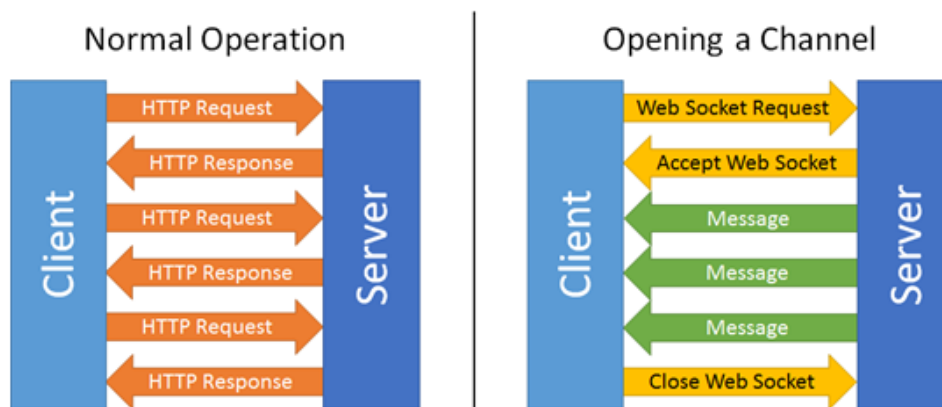


Figure 4. HTTP vs WebSocket communication comparison.

In the PANTHEON project, the WebSocket server was integrated into the back office and it was created using the TypeScript language via node.js and the Express and Express-ws libraries.

To develop the client in the front-end, Angular Rxjs library was used, employing in particular the WebSocket, WebSocketSubject and Observer classes.

Using WebSocket, it is possible to view the following information in real time in the interface:

- Notification of alarms;
- Campaign monitoring (UGV positioning);
- Visualization of real-time IoT data.

2.1.5 Building blocks

The Project partners are in charge of work packages that concern the development of subsystems. Table 2 describes the elements involved in the integration process, which we can treat as the building blocks of the SCADA system.

| Activity | Subsystem | Unit Leader | Subsystem Description |
|-----------------------|--|-------------|--|
| UGV Data Collection | UGV Campaign | UNIROMA3 | This subsystem concerns the development of the ground robot, with autonomous driving capabilities to reach the mission target trees and collect related data. An RGB camera, a multispectral camera and a laser scanner are hosted on the platform for data capture. |
| UAV Data Collection | UAV Campaign | ULB | This subsystem concerns the development of a drone to fly over the field and to acquire aerial data at low altitudes. The payload carried by the drone for data acquisition consists of 3 chambers: multispectral, thermal and RGB. |
| Data Processing | Remote Sensing | TRIER | This subsystem concerns the development of image processing and scanning algorithms. The data collected during UGV and UAV missions are processed. The processing generates synthetic processed data that can be used for the management of agronomic activities. |
| IoT Data Collection | IoT | UNIROMA3 | This subsystem concerns the monitoring sensors installed in the field. It is composed of the soil moisture sensors and the weather station. |
| Stream processing | Alarms Real-time | SIGMA | This subsystem deals with the real-time management of IoT data. |
| Data storing | Historical Data | UNIROMA3 | This subsystem concerns the definition and development of the system database and the procedures for storing and historicizing data. |
| Activities Management | Suggested Activities Crop Yield Forecasting | UNITUS | This subsystem concerns the development of methodologies applied to the agronomic activities of hazelnut groves. It allows you to train the system to suggest intervention activities following the data collected or provide future projections given the trend of historical data. |
| User Interface | User Application | SIGMA | This subsystem concerns the development of the user application. It is the interface that |

| | | | |
|--|--|--|--|
| | | | allows you to use the features of the PANTHEON SCADA system. |
|--|--|--|--|

Table 2. Summary of the subsystems involved in the integration process.

In the next section, we will describe the integration process that allows to aggregate all the developed blocks, showing the data flow, the interaction mechanisms, and the interfaces between the various subsystems.

2.2 System Integration Design

Once the strategy and integration methods were established, we moved on to the analysis and design phase of the solution. This phase created an integration scheme. The schematization allows to have a clear view of the system, of the subsystems which is composed, of the data flow and of the interfaces to be developed.

The following paragraphs illustrate the integration schemes at the system level and at the subsystem level.

2.2.1 Integration schema overview

Figure 5 shows the general scheme of PANTHEON SCADA system. This scheme allows to have a high-level overview, and it makes the description in detail of the subsystems involved easier.

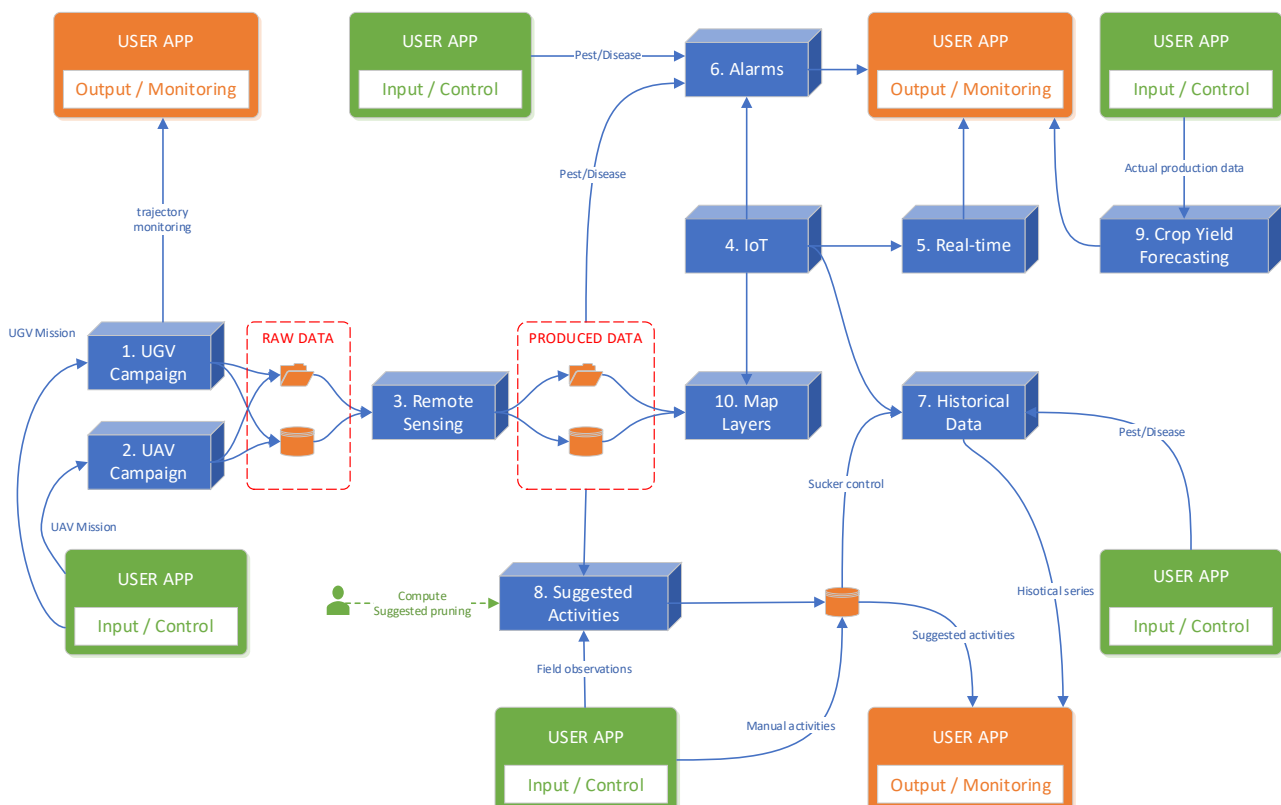


Figure 5. Integration schema overview.

The symbology does not follow a predefined standard to make everything simpler and more immediate. The integration detail schemes, for subsystems, also use the same symbology. Although it is easy to understand, Figure 6 shows a legend of the used symbols.

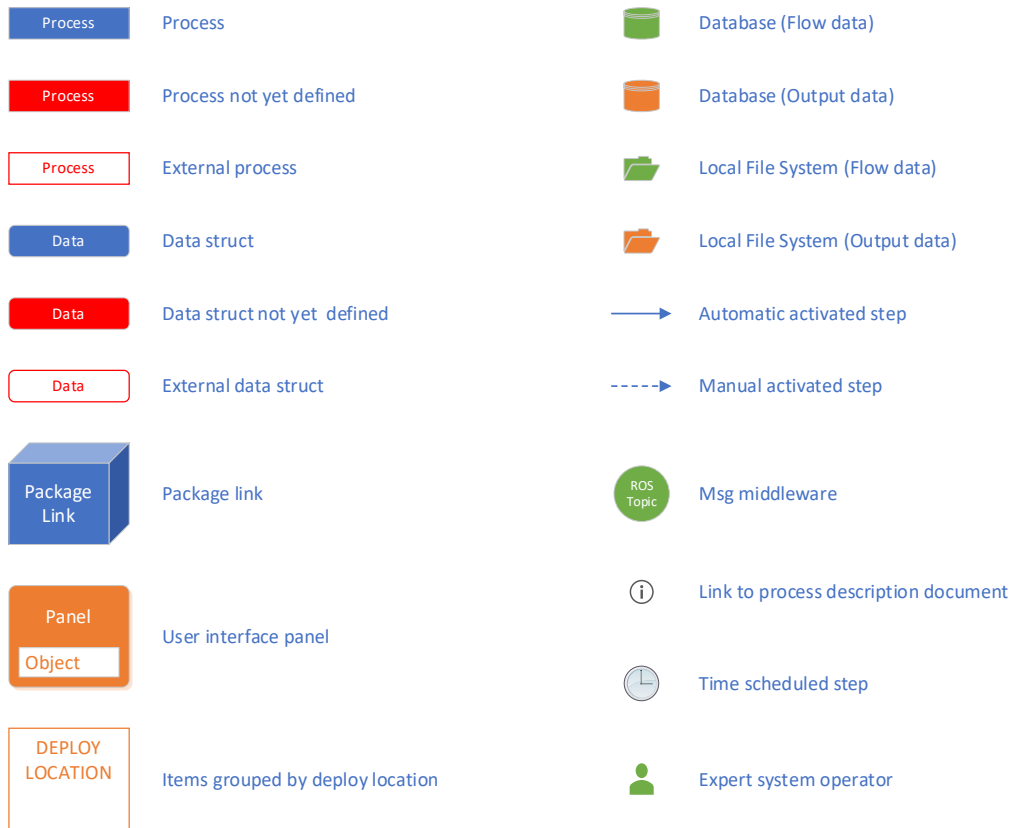
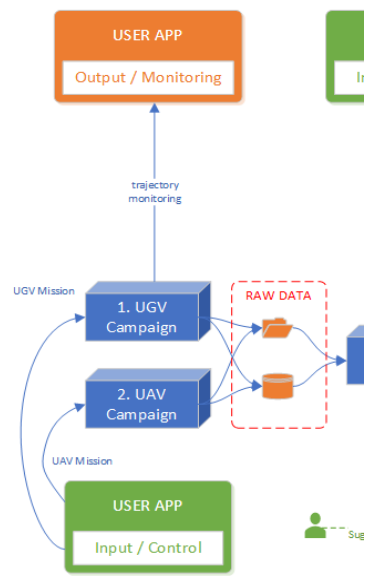
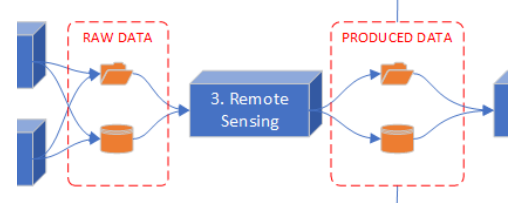
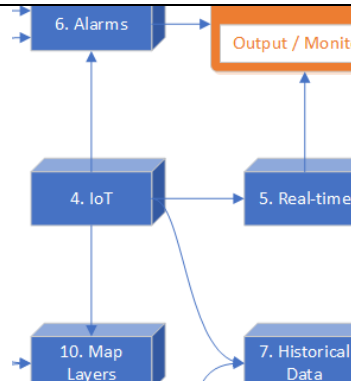
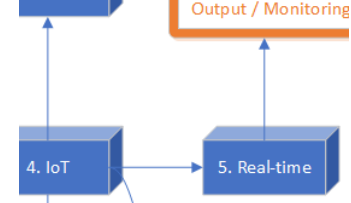


Figure 6. Integration schemes legend.

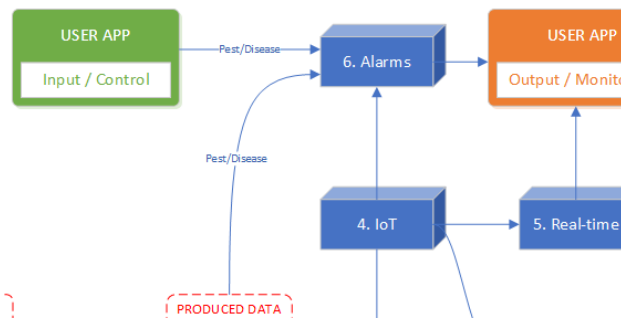
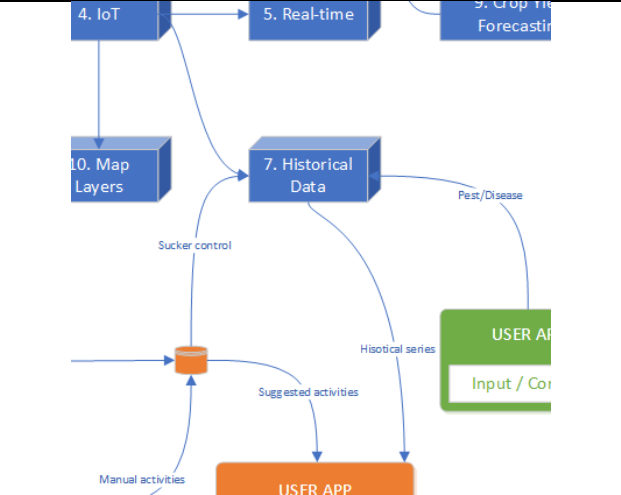
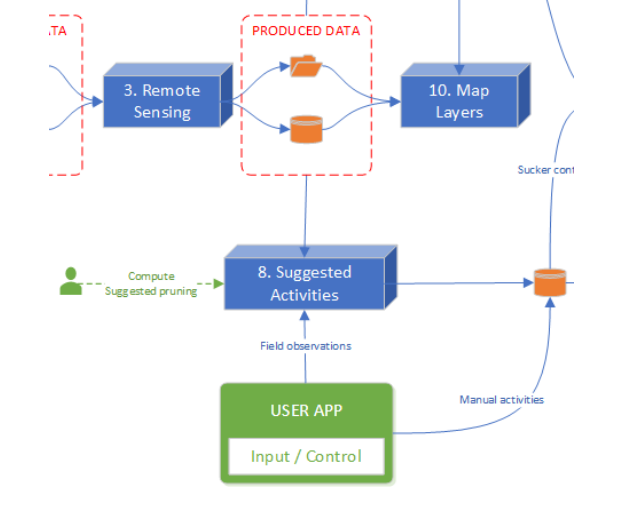
In the following paragraph the integration scheme in Figure 5 will be described, which will allow us to understand the thread that binds the various subsystems developed.

2.2.2 Integration schema description

Table 3 describes the 10 packages identified. Graphically represented as blue cubes, each of them represents a subsystem that will be described in detail in the next sections. Packages are logically interconnected to show the data processing path. In the diagram, there are green and orange windows that represent the user interface, that is points that allow the input (green) and the output (orange) between the operator and the SCADA system.

| | |
|---|--|
| <p>The first two packages, 1 and 2, are related to UAV and UGV campaigns. Through the user interface, the operator plans activities to be performed in the field, in particular the data acquisition missions. These missions are scheduled and act as input for UAV and UGV operators who collect data in the field. The user application allows you to monitor the UGV mission in real time through remote telemetry acquisition. The acquisition missions produce an output consisting of photographs, scans, and related metadata. The raw data produced is archived on centralized databases and file systems.</p> |  |
| <p>The collected data will be input of package 3, or "Remote Sensing", which includes data processing with the aim of extracting significant information for the management of agronomic activities. The result of the processing, that is the "produced data", are also archived on the database and file system.</p> |  |
| <p>Package 4, "IoT" (Internet of Things), is the block that completes the automated data acquisition in the field. In this case, meteorological and ground data are acquired using sensors installed in the field. In particular, the weather station and soil moisture sensors acquire monitoring data in continuous way.</p> |  |
| <p>At this point, the packages that use the acquired data to implement the system functions are involved. Package 5, "Real-time", allows you to perform the monitoring function and provides the operator with the option to consult the values detected by the IoT sensors remotely and in real time.</p> |  |

Precision Farming of Hazelnut Orchards (PANTHEON)

| | |
|---|--|
| <p>Package 6, “Alarms”, is responsible for monitoring the environmental parameters and the results of the robots' surveys to report any alerts from an agronomic point of view, which are useful for intervening in a timely manner. The alerts also take into account the surveys made by agronomists directly in the field and entered into the system through the application.</p> |  |
| <p>Package 7, “Historical Data”, is responsible for memorizing all the agronomic activities carried out, the environmental parameters measured over time and the agronomic observations that took place directly in the field. This allows you to build a knowledge base useful for data analysis purposes.</p> |  |
| <p>Package 8, “Suggested Activities”, collects advanced algorithms that derive from hazelnut management models, the subject of experimentation of the PANTHEON project. The algorithms involve a first tuning period carried out with the collaboration of agronomists. When fully operational, this subsystem will use the data produced by the “Remote Sensing” and the field observations to automatically suggest agronomic interventions to be performed in the current situation.</p> |  |

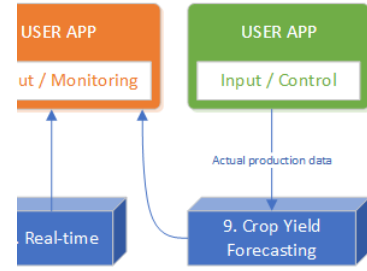
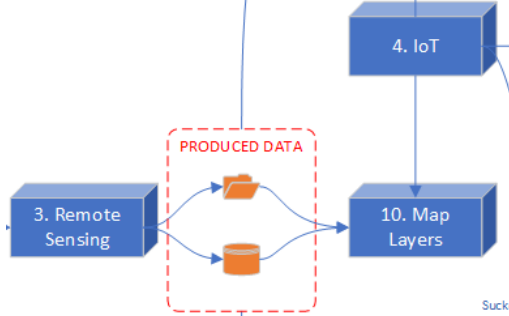
| | |
|---|--|
| <p>Package 9, “Crop Yield Forecasting”, attempts to estimate the quantity and quality of the season's harvest based on the history of previous years. This wait is particularly useful in managing cultivation.</p> |  |
| <p>Finally, package 10, "Map Layers", is useful for making the data produced by Remote sensing and IoT sensors usable in the user interface, in particular for viewing raster and vector georeferenced data on a map.</p> |  |

Table 3. Package’s description of the overview schema.

In the next section (2.3) each subsystem will be described in detail, highlighting the aspects of aggregation and integration.

2.3 Sub-System Design

The packages, represented as blue parallelepipeds in the diagram of Figure 5, represent the subsystems to be integrated. Such formalism was introduced to manage the complexity of the system, just as the development of a complex system requires the subdivision into work packages and even the schematization of the integration must follow the same path. The previous chapter (2.2) has shown a view of global integration at system level, while in the next paragraphs we will go into the details of the individual packages, that is:

- UGV Campaign;
- UAV Campaign;
- Remote Sensing;
- IoT;
- Real-Time;
- Alarms;
- Historical Data;

- Suggested Activities;
- Crop Yield Forecasting;
- Map Layers.

2.3.1 UGV Campaign

Scope, objective, and functionalities

The UGV Campaign involves two UGVs, namely robot A ($R-A$) and robot B ($R-B$). The two robots possess the same basic sensorial equipment and share a common localization, safety, and navigation system.

The main task of $R-A$ is to scan the orchard to collect sensorial data for assessing the water stress level of the plants, for detecting the presence of pest and/or diseases on the plants and for virtually reconstructing the tree. The information on water stress and pest/disease presence contribute to the overall assessment of the phytosanitary status of the plants as discussed in the respective Deliverables D5.1 (Water Management Control [3]) and D4.5 (Pest and Disease Detection [4]). The virtual tree reconstruction, whose functioning is detailed in Deliverable D4.2 [5], enables the fruit management and detection procedures, detailed in Deliverables D4.6 (Fruit Detection) and D5.5 (Fruit Development and Production Monitoring), and the pruning management procedures, detailed in Deliverable D5.3 (Pruning Management Protocol) [6].

The main task of $R-B$ is to manage the treatment of suckers which consists in detecting their presence and applying chemicals with the scope to remove them as explained in Deliverable D5.2 (Suckers' Management Control) [7].

From a software standpoint, every main component of each robot is integrated and managed within the ROS environment, thus making it easier to manage, control, and save to the DB infrastructure all real-time data captured by robots or IoT sensors.

Integration schema

Figure 7 shows the integration scheme of the UGV Campaign subsystem:

Precision Farming of Hazelnut Orchards (PANTHEON)

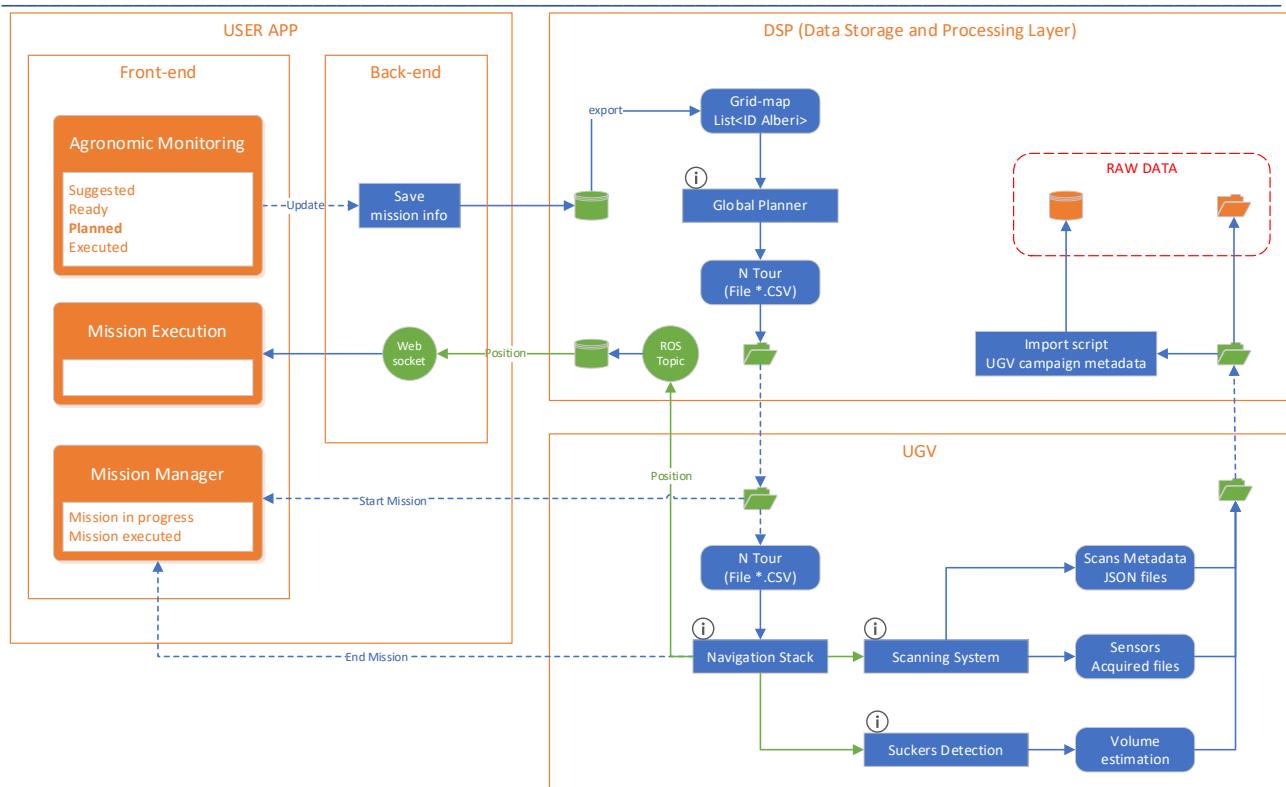


Figure 7. UGV Campaign subsystem integration scheme.

The data acquisition campaigns for UGV are planned through the user application, as well as the rest of the agronomic activities. The mission definition allows you to specify details such as the platform to be used, which sensors to use, the data collection targets and the day of execution of the activity.

The information, stored in the DSP database, is used by the “Global Planner” process to generate the paths that the UGV platform will have to automatically run. At this point, the file with the paths is transferred to the robot and the mission can be carried out in the field.

At runtime, the navigation system will allow the autonomous movement of the robot, the scanning system will collect data in specific points of the path based on the mission targets, while the sucker detection system will estimate the volume of the suckers.

During the execution of the mission, the robot will communicate its position in real time via the network with the DSP server, allowing remote monitoring through the user application. The scanning and detection processes, on the other hand, will save the collected data in local memory which will then be transferred to the server.

The different data transport strategy towards the server is motivated by two elements: the weight of the data to be transferred and the purpose of using the data. The real-time positioning of the robot is a relatively light figure, it takes up little bandwidth and it may be useful to remotely monitor the movement of the robot,

to follow its trajectory and the status of the activity. On the other hand, there is no need to have the collected data available in real time, as they will undergo post-mission batch processing. This motivation does not make it convenient to size a complex transmission system with sufficient bandwidth to transmit the huge amount of data collected.

When the data collected will be uploaded to the DSP server, the files will be placed in the predefined acquisition structure while the metadata will be imported into the database, both ready to be processed by the “Remote Sensing” module.

2.3.2 UAV Campaign

Scope, objective, and functionalities

The objective of this subsystem is to be able to obtain remote sensing measurements of the phytosanitary status of the orchard from different kind of sensors, used simultaneously or separately, while ensuring a high accuracy and quality of the data obtained.

The UAV subsystem has been developed to be able to integrate 3 different kind of sensors in a single flying platform: an RGB camera, a multispectral camera, and a thermal camera. These 3 sensors are controlled by an onboard computer which is also in constant communication with the flight controller of the UAV.

The system implemented computes an optimal mission based on the cameras used, relying on their field of view and the desired overlap, and triggers the selected cameras and the appropriate instants. The mission is computed offline and transferred to the UAV before the flight. This procedure has shown to be the best in terms of reliability and security during the sensing missions.

The UAV uses an RTK base station to correct the local position of the UAV during the flight and to provide an accurate path. To ensure the quality of the data obtained, the onboard computer also records the UAV data associated to each measurement providing a log file that can be used in the post processing of the data to obtain orthomosaics of high quality.

It must be noted that the UAV subsystem implemented is not directly connected to the rest of the SCADA system during the flight operation, and that the data is later transferred to the server. This decision responds to the current UAV regulations that are very strict regarding the human control of the UAV during every mission, and which make the use of completely autonomous missions not possible in future applications. Based on that, in the system developed the central unit informs of a remote sensing mission on the area and then receives the data to be analyzed. We believe that this procedure is more suitable for a near-future implementation of the system.

Integration schema

Figure 8 shows the integration scheme of the Campaign UAV subsystem:

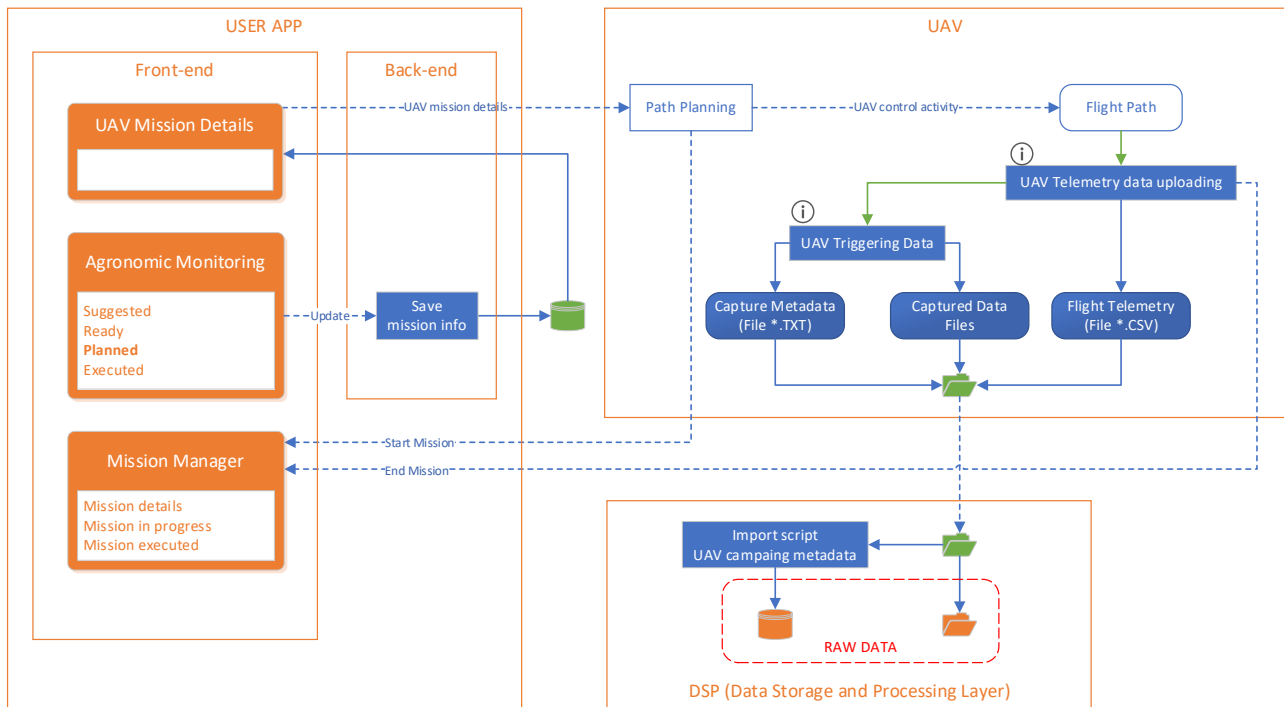


Figure 8. UAV Campaign subsystem integration scheme.

The data acquisition campaigns for UAVs are planned through the user application, as well as the rest of the agronomic activities and the UGV acquisition missions. The mission definition allows you to specify details such as the platform to be used, which sensors to use, the data collection targets and the day of execution of the activity.

The UAV operator can access the mission details and program the drone planner to fly over the requested acquisition area. Unlike missions with UGV, for safety and regulatory reasons the drone must have an autonomous control system supervised by the operator that cannot be integrated into the SCADA system. This does not allow you to follow the operations in real-time, but the operator can define the status of the mission (start and end) through the user application.

At runtime, the navigation system will allow the autonomous movement of the drone with operator supervision. The scanning system will carry out data collection by flying over the target mission area. The scanning and detection processes will save the collected data on local memory which will then be transferred to the server.

When the data collected will be uploaded to the DSP server, the files will be placed in the predefined acquisition structure while the metadata will be imported into the database, both ready to be processed by the “Remote Sensing” module.

2.3.3 Remote Sensing

Scope, objective, and functionalities

The main focus of the remote sensing subsystem is tree geometry reconstruction (Task 4.3), fruit detection (Task 4.8), water stress measurement (Task 4.6) and pest and disease detection (Task 4.7) [1]. The UAV based sensors are designed to monitor a) the state of pigments in the canopy by measuring leaf reflectance in the visible and near infrared range (Tetracam MCAW) and b) vegetation stoma activity and the resulting changes in evapotranspiration by measuring temperature differences (TeAx ThermalCapture). The raw output of a data acquisition campaign is a number of images for each respective sensor, which provide enough overlap to allow for the production of a seamless orthomosaic that covers the entire area. The orthomosaic is then used to calculate indices and model variables. These modelled variables can then be visualized as water stress maps in the SCADA system. Additionally, variable aggregates can be extracted for each tree and visualized with line diagrams. The UGV based sensors are designed to acquire point clouds of each individual tree and use these point clouds to reconstruct the tree geometry, which is then used to plan the pruning process of the tree. Additionally, the enriched point clouds are then used to estimate the plant yield by classifying the nut clusters.

Integration schema

Figure 9 shows the integration scheme of the UAV “Remote Sensing” subsystem:

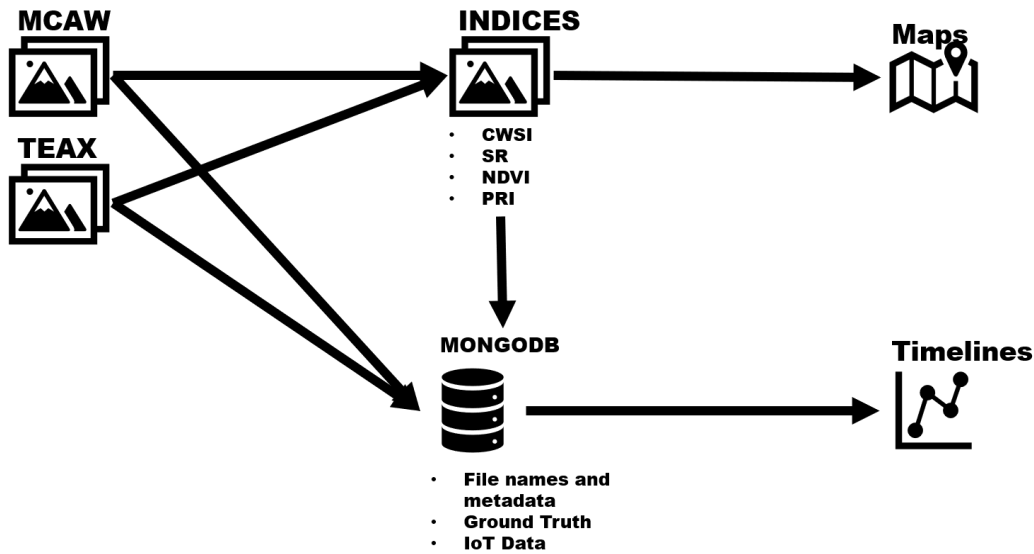


Figure 9. The UAV Processing, integration scheme.

At the beginning of the UAV remote sensing integration, raw files from the sensors are uploaded to the server. This triggers the python-based processing chains, which add the files and metadata to the database, and process the raw images with raster files for the indices as a final output. These raster files are then used to visualize risk maps with the SCADA system. Additionally, values for each tree are extracted based on its location, which are then added to the database. These values can then be visualized in a timeline together with the ground truth data.

Figure 10 shows the integration scheme of the UGV “Remote Sensing” subsystem:

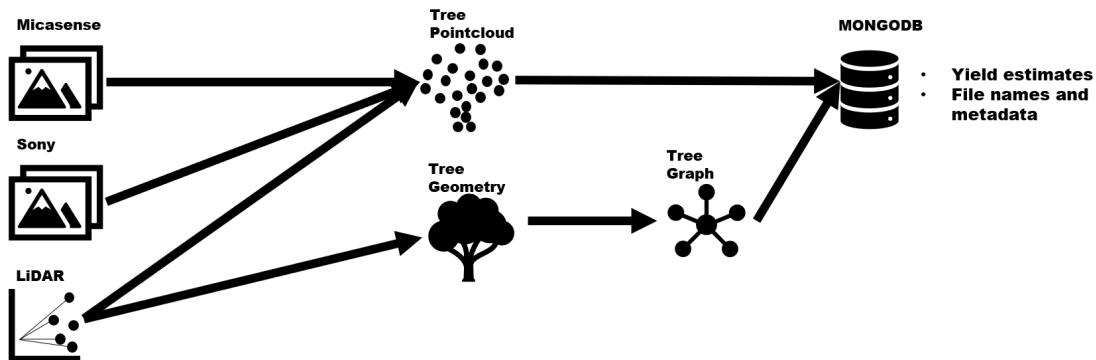


Figure 10. The UGV Processing, integration scheme

The raw files, namely the Sony and Micasense images as well as the Faro laser scans are uploaded to the server. Then the Python scripts to convert file formats, pre-process the laser scans and co-register the laser scans are triggered. Additionally, the image color values are projected onto the point clouds in order to produce the enriched point clouds used in the fruit detection. Details can be found in “D4.1 - Multispectral

LiDAR Point Clouds” [8]. Finally, the aligned point clouds representing the trees are extracted and tree geometries are reconstructed.

2.3.4 IoT

Scope, objective, and functionalities

In the PANTHEON project, the IoT sensor network is used for monitoring climatic / environmental conditions and for evaluating possible risk factors that could affect production.

The IoT infrastructure provides to the DCP server all the necessary data for the agronomical assessment, including among other data for fruit detection, tree geometry reconstruction, water stress analysis, pruning plant policies, and phytosanitary evaluation.

The IoT infrastructure is composed by the Wireless Network Backbone (WNB) and the LoRa Network, as described in Deliverable D3.3 (Communication Infrastructure [9]).

The WNB is the infrastructure responsible for interconnecting the elements of the SCADA system. All the components, from the DCP server to the single robots moving in the field, are interconnected through the WNB. The LoRa Network is a digital wireless data communication technology which is responsible for sensing, management, and monitoring of weather conditions.

Below, the list of devices installed in the field:

- WNB
 - AIRMAX antennas (LiteBeam AC GEN2);
 - UNIFI antennas (AC MESH PRO);
 - 1 router (IR615-S-L3-WLAN).
- Lora Network:
 - 9 Sensors (or nodes) which are scattered in the field;
 - 1 custom weather station;
 - 1 Raspberry Pi 3B+.

Through the visualization and analysis of these data it is possible to verify the current conditions of the field, evaluate how the climatic conditions have affected the production over time and evaluate important aspects such as the management of water requirements and the management of events that could cause damage to the orchard.

Integration schema

Figure 11 shows the integration scheme of the IoT subsystem:

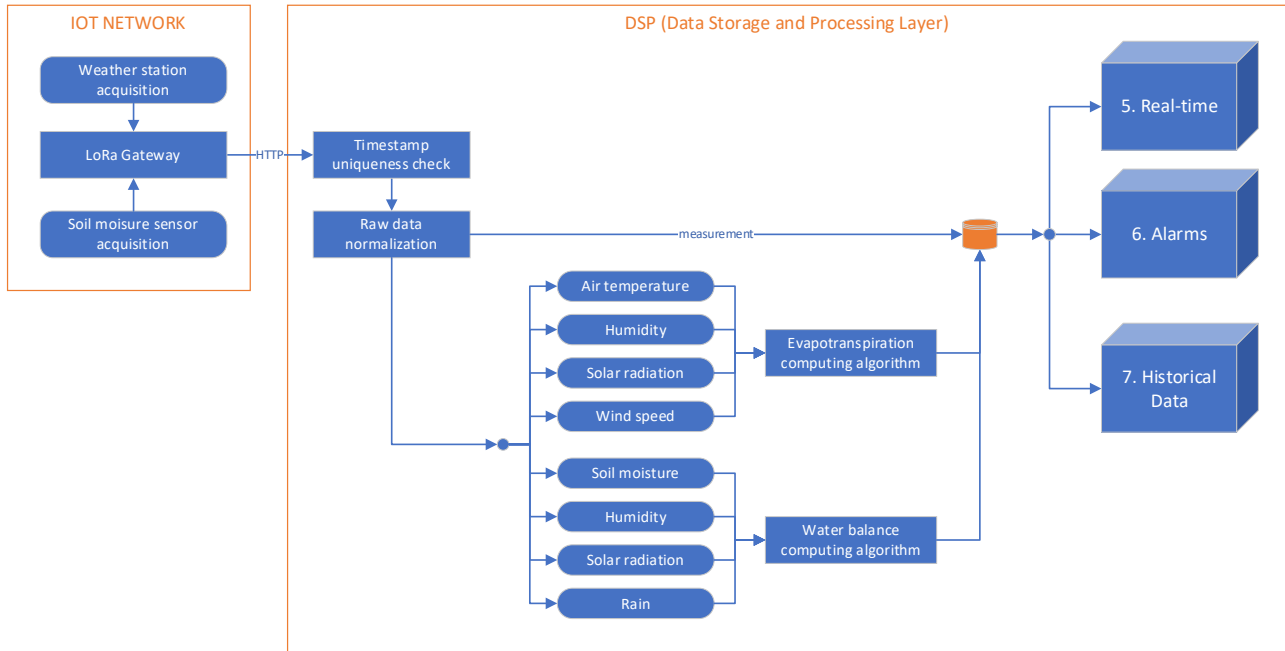


Figure 11. IoT subsystem integration scheme.

The "IoT" subsystem concerns the collection, transmission, and storage of environmental measurements (meteorological and soil ones) acquired through sensors installed in the field. In this case, the integration work is focused on realizing the data exchange that takes place between the IoT network and the DSP server and on storing the received data on the database.

The data detected by the sensors, both weather station and of the nodes on the ground, are collected by a Lora gateway which takes care of packaging and sending them through an HTTP call to the PANTHEON Web API.

The environmental status is monitored every 5 minutes and the gateway carries out a first pre-processing of the data. The minimum, average and maximum values are calculated in an interval of 15 minutes and sent to the DSP server.

On the acquired data, on the server side, checks are carried out to verify the correctness of the format, that is, that it complies with the defined specifications, and that the data has not already been received previously. The gateway does not have internal storage capacity, so in some situations it is unable to know which data has already been transmitted: so, on the server side it is verified the uniqueness of the data based on the acquisition time mark (timestamp).

Subsequently, the package containing all the measurements is normalized and the data of the individual sensors are extracted and stored separately on the database. At the same time, the acquired measurements are used for the calculation of derived values, such as evapotranspiration and water balance.

Once acquired, the environmental measurements will be used for various purposes:

- Real-time visualization of the climatic conditions of the field;
- Generation of alarms regarding critical situations for production;
- Construction of the climatic data history.

2.3.5 Real-time

Scope, objective, and functionalities

The collection and display of real-time data allow, through the user interface, to monitor the current weather conditions even remotely.

This is especially important as it allows you to make decisions and take immediate actions based on the needs of the plantation and individual trees, for example by regulating irrigation based on the soil moisture.

Integration schema

Figure 12 shows the integration scheme of the "Real-Time" subsystem:

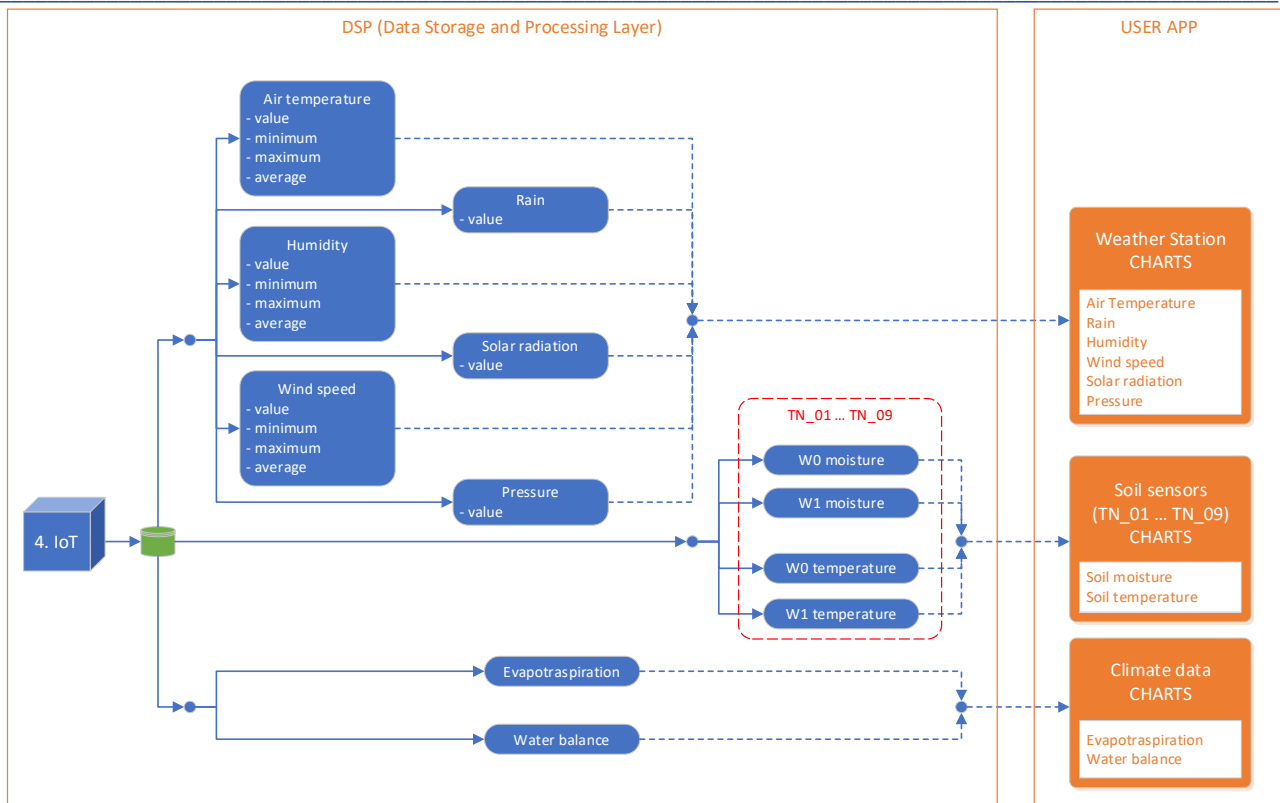


Figure 12. Real-Time subsystem, integration scheme.

The "Real-time" subsystem allows operators to remotely monitor IoT environmental parameters via user application. The integration work, in this case, consists in providing an access point for the user interface in order to receive the data coming from the IoT subsystem.

When a user views the page with the real-time IoT dashboards, the various graphs regarding each sensor are shown.

A first group of graphs concerns the data acquired through sensors of the weather station:

- Air temperature: Line chart type with series min (°C), max (°C) and average (°C);
- Rain: Bar chart type with only the series sum (mm);
- Humidity: Line chart type with series min (%), max (%), and average (%);
- Wind speed: Line chart type with series min (m/s), max (m/s) and average (m/s);
- Solar radiation: Line chart type with only the series solar radiation value (W/m2);
- Pressure: Line chart type with only the series acquired value (mbar).

A second group concerns the data acquired through sensors of the ground nodes, for each node it is shown:

- **Soil moisture:** Line chart type with the following series $w_0=15\text{cm (m}^3/\text{m}^3)$ and $w_1=40\text{cm (m}^3/\text{m}^3)$;
- **Soil temperature:** Line chart type with the following series $w_0=15\text{cm (}^\circ\text{C)}$ and $w_1=40\text{cm (}^\circ\text{C)}$.

A final group concerns the calculated data:

- **Evapotranspiration:** Bar chart type with the calculated data (mm) series daily;
- **Water balance:** Bar chart type with the calculated data (mm) series daily.

2.3.6 Alarms

Scope, objective, and functionalities

The presence of alarm events in an integrated system that allows the management of the field helps to pay attention to critical situations that could cause damage to the crop and a lower yield in production.

The main causes of these damages can be climatic, such as frosts, gusts of wind, hail, or heavy rain, or caused by diseases and insects.

Notifying the operator of a danger of this type in a timely manner allows agronomists or field managers to take sudden actions to avoid or limit damage.

Integration schema

Figure 13 shows the integration scheme of the "Alarms" subsystem:

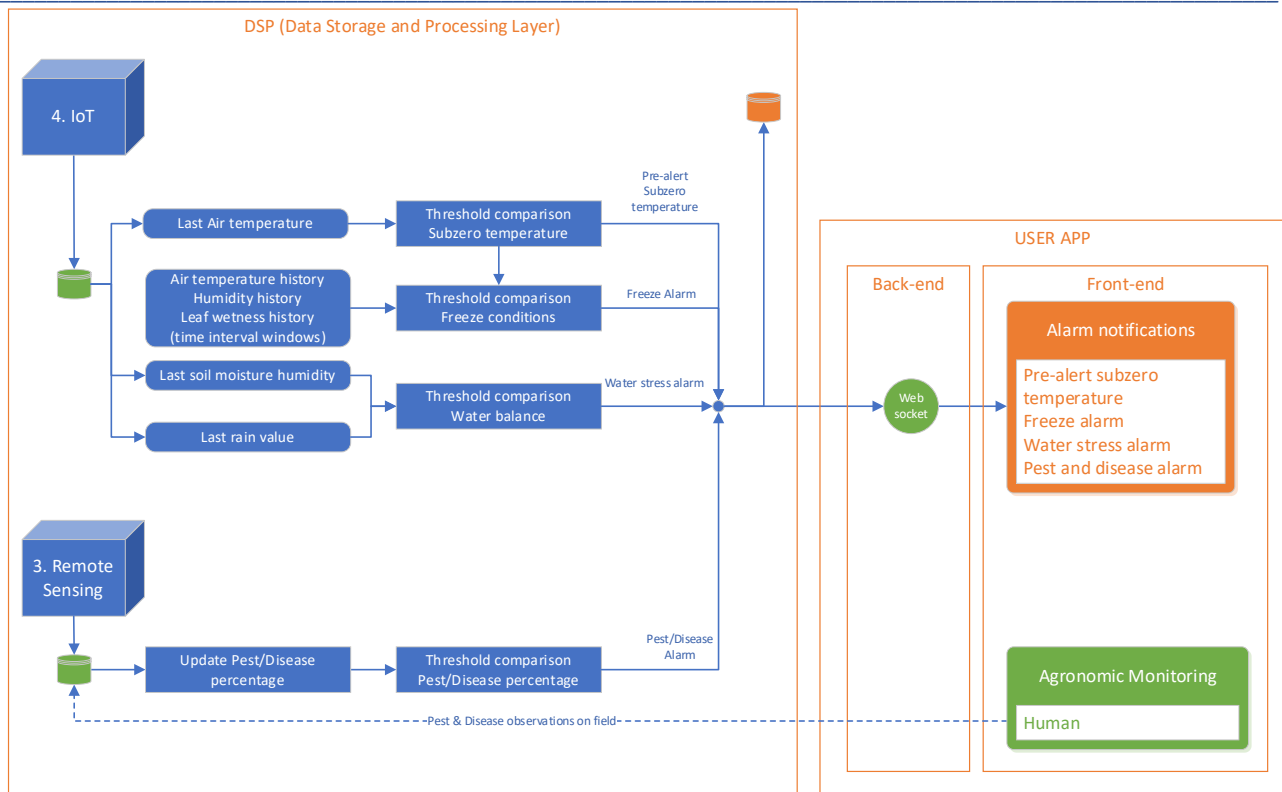


Figure 13. Alarm's subsystem, integration scheme.

The “Alarms” subsystem processes, as soon as available, the data produced by the “IoT” and “Remote Sensing” subsystems. Rules derived from agronomic models are applied to these inputs, which make it possible to detect anomalies and report them to operators via the user interface, as well as log the report. The integration work, in this case, mainly concerns the application of agronomic models on the input data and preparing the notification system to the user application.

The alarms that are managed in the system can depend both on unique values read by the sensors and on aggregated data considering a predefined time window. Some events are monitored throughout the year, while others depend on the phenological age of the plants.

In the project, the following alarms are monitored:

- Pre-alert temperature: the alert is triggered when the single temperature value drops below 0°C. This alert is monitored for 12 months a year;
- Frost Damage Alert
 - during the flowering period (15 December - 15 February) only the temperature is checked. The alert is activated when the minimum temperatures drop below -5°C for at least one hour continuously;

- in the period of vegetative restart (February 16 - April 30) the alert is triggered when the temperature drops below -2°C , the relative humidity of the atmosphere is higher than 50% and the wind is lower than 15 knots for at least one hour continuously.
- Water balance: during the hazelnut growing season, from May 1st to September 30th, the alert is triggered when the soil moisture content (m^3/m^3) recorded by any of the soil moisture sensors drops below 0.12 (m^3/m^3), according to the daily or weekly entities of the reference evapotranspiration (ET0) and effective rainfall (R) to be related to the proper crop coefficients (Kc).
- Pest and disease: For the gall mite *Phytoptus avellanae*, the migration phase starts usually in April when the mean temperature is $10\text{-}11^{\circ}\text{C}$. Treatment against this pest should be applied in this period. For the *Anthraco* disease too few data about the pathogen epidemiology are known which makes this alarm not applicable.

In the application interface, the notification of alarms occurs in two ways:

- Alarms in evidence: there is a widget that shows the alarms in progress, when the alarm falls within the allowed threshold, it is automatically deactivated by the system, and it disappears from those in evidence;
- Alarm history: there is a page dedicated to the list of alarms that have occurred over time and where there are all listed, both the active and completed ones.

2.3.7 Historical Data

Scope, objective, and functionalities

In a crop management system, historical data analysis represents an important resource for analyzing and comparing data, both meteorological and agronomic ones, between the different years.

Historical analyzes have been included in the system for:

- Meteorological data acquired through the weather station;
- Data acquired through sensor nodes in the soil;
- Aggregation of calculated data;
- Aggregation of data on the growth of suckers;
- Aggregate historical data on damage and plagues.

Integration schema

Figure 14 shows the integration scheme of the "Historical Data" subsystem:

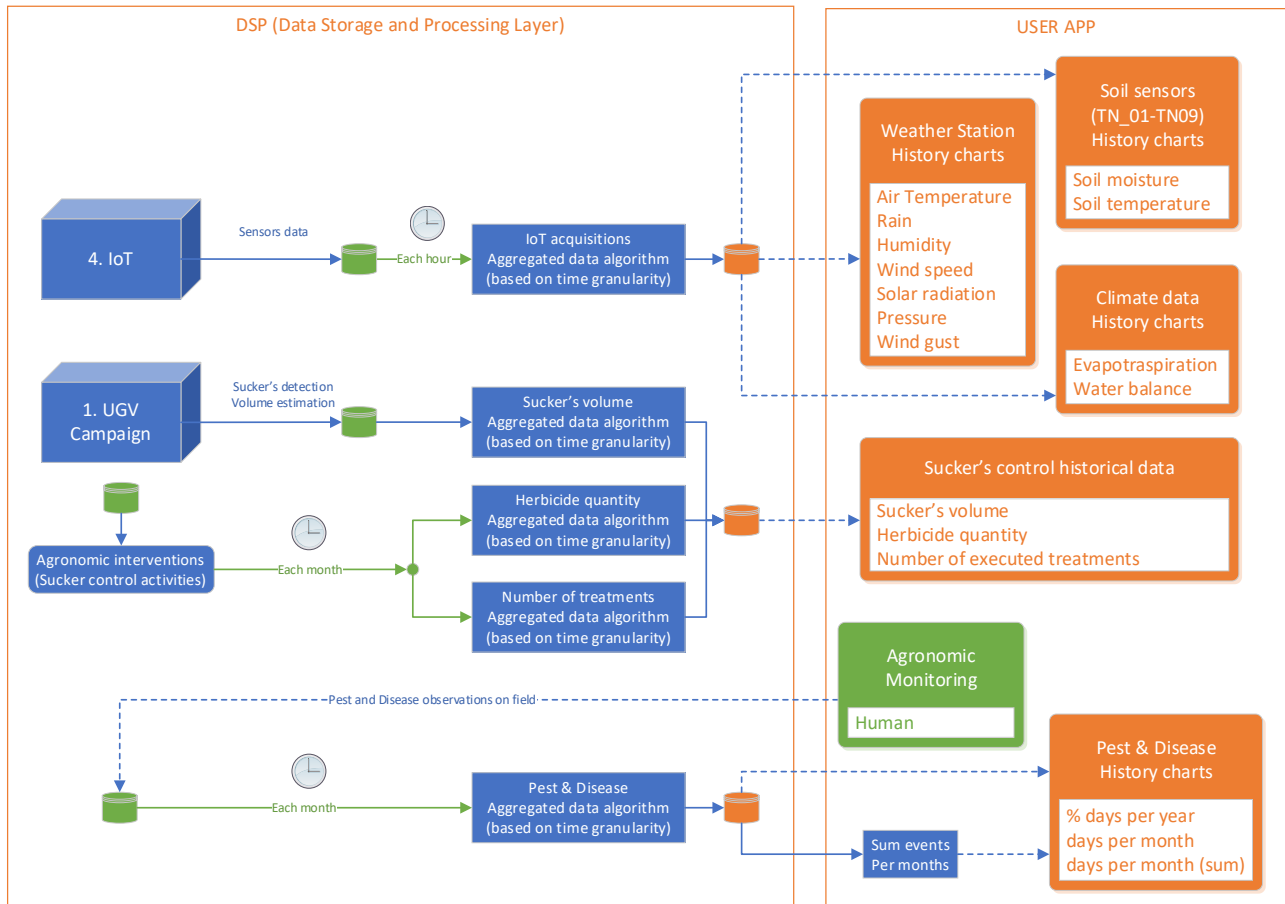


Figure 14. Historical Data subsystem, integration scheme.

The "Historical Data" subsystem concerns the historicization of data aggregations performed at predefined time intervals (hourly, daily, weekly, monthly, and yearly). These indexes are stored on databases for analysis purposes and reported to the operator via the user interface. The integration work, in this case, focuses on the implementation of the aggregation algorithms that are automatically scheduled in certain time periods. Starting from the collected data from IoT sensors and from field scans, the data is aggregated, and historical values are calculated for each individual piece of information on an hourly basis. On the historical data view page, you can select the granularity with which to analyze the data.

A first group of graphs concerns the meteorological data:

- Air temperature: Line chart type with the min (°C), max (°C) and average (°C) series of the arithmetic mean of the acquired values;
- Rain: Bar chart type with only the series: sum (mm) of the sum of the acquired values;

- Humidity: Line chart type with the series min(%), max(%) and average(%) of the arithmetic mean of the acquired values;
- Wind speed: Line chart type with the min(m/s), max(m/s) and average(m/s) series of the arithmetic mean of the acquired values;
- Solar radiation: Line chart type with only the solar radiation value(W/m²) series of the arithmetic mean of the acquired values;
- Pressure: Line chart type with only the acquired values(mbar) series of the arithmetic mean of the acquired values;
- Wind gusts: Line chart type with only the max(m/s) series with the maximum values acquired;
- Soil moisture w0: Line chart type with a series representing the average value on each sensor plus an additional series representing the average of all sensors, represented in m³/m³;
- Soil moisture w1: Line chart type with a series representing the average value on each sensor plus an additional series representing the average of all sensors, represented in m³/m³;
- Soil temperature w0: Line chart type with a series representing the average value on each sensor plus an additional series representing the average of all sensors, represented in °C;
- Soil temperature w1: Line chart type with a series representing the average value on each sensor plus an additional series representing the average of all sensors, represented in °C;
- Evapotranspiration: Bar chart type with the calculated values(mm) series of the sum of the acquired values;
- Water balance: Bar chart type with the calculated values(mm) series of the sum of the acquired values.

A second group concerns the data on the management of suckers:

- Suckers volume;
- Amount of herbicide;
- Number of treatments.

A final group concerns pest and disease, and it includes the following graphs:

- Number of days per year;
- Days per month;

- Sum of days per month.

2.3.8 Suggested Activities

Scope, objective, and functionalities

In the project, particularly important are the algorithms that, by analyzing the surveys carried out, both automatically via UAV and UGV and manually by operators in the field, detect situations in which to intervene and those in which to support agronomists in choosing the agronomic activities to be performed to improve the state of health of each individual plant.

The types of activities that can be suggested by the System concern:

- Pruning management;
- Sucker control;
- Water management;
- Pest and disease.

For each activity to be undertaken, details such as, i.e., type and quantity of pesticides to be administered, branches to be cut, quantity of water to be administered are also suggested.

In particular, the pruning management protocol analyses the virtually reconstructed tree to individuate branches of the tree that do not meet certain agronomical criteria in terms of shape, orientation, or spatial location. If such branches are found, then the management protocol provides a visual representation of the tree suggesting which branches are to be removed to properly fit the agronomical criteria.

In addition, through the interface, users will be able to enter additional agronomic activities carried out on the trees to monitor all treatments performed on individual trees and to reconstruct the entire growth history of each tree in the field.

Integration schema

Figure 15 shows the integration scheme of the "Suggested Activities" subsystem:

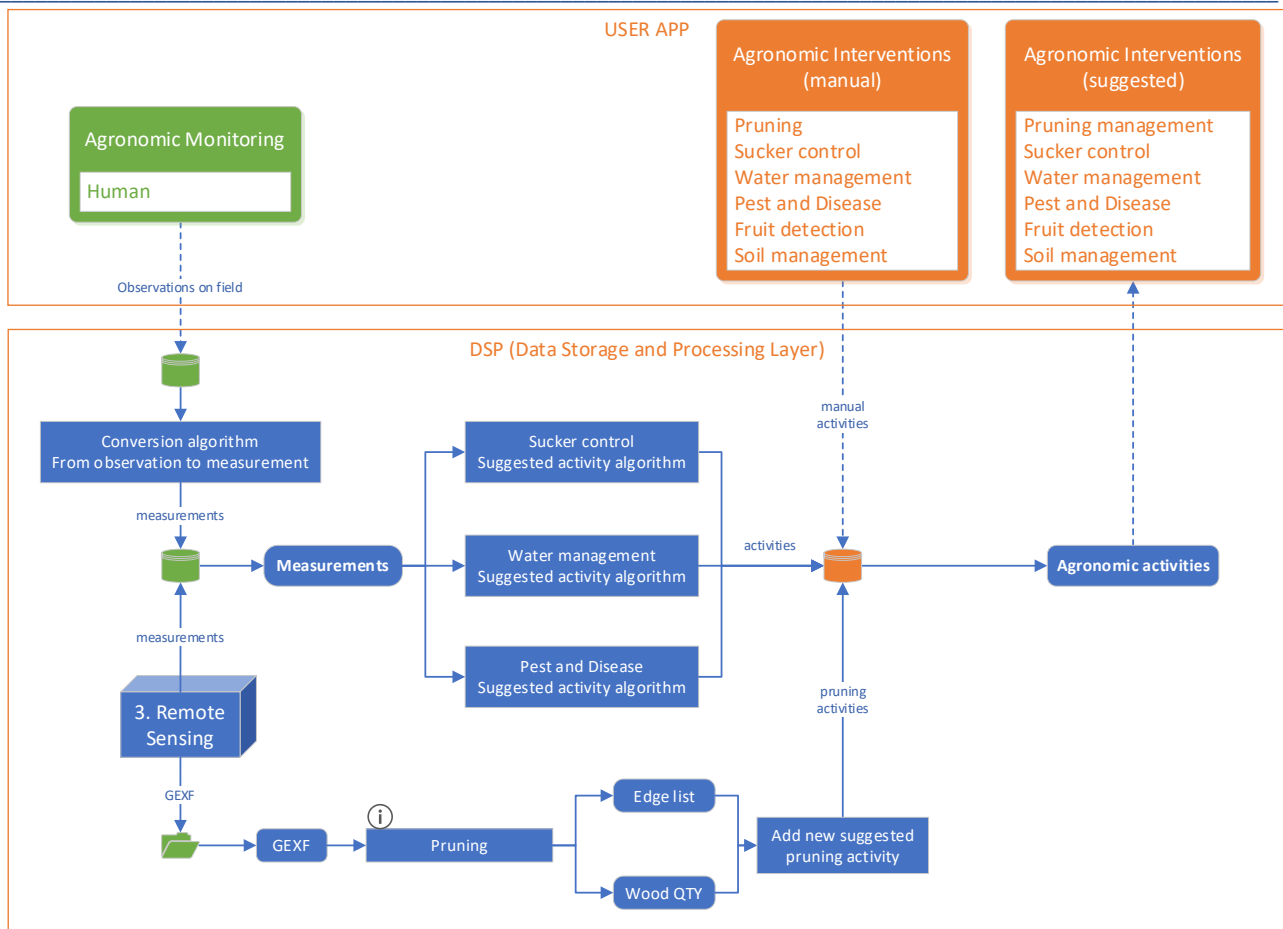


Figure 15. Suggested Activities subsystem, integration scheme.

The “Suggested Activities” subsystem is responsible for implementing the agronomic models studied by the PANTHEON project. These models, to be validated in field experimentation, allow the system to automatically generate agronomic activities based on the current and historical state of the field, acquired through the surveys, both automatically and manually. In particular, the suggested activities will be those of pruning, herbicide administration for the control of suckers, irrigation management and disease control activities.

The work of integration, in this case, consists in building the processing chain. It starts by identifying the necessary inputs to be supplied to the algorithms, whether they come from remote sensing or from human observations. Secondly, the automatic execution of the algorithms is scheduled for new input data. Finally, the output generated by the algorithms is analyzed to suggest new agronomic interventions.

In detail, starting from the measurements, the data on the quantity of suckers detected, on soil humidity and on damage from plague and diseases are analyzed, after which suggested agronomic activities are generated. These are shown in the user interface to allow agronomists or field operators to verify these suggestions and to plan their execution.

In addition, a dedicated algorithm is able to analyze the structure of the tree branches, and it allows you to generate suggestions on which branches to prune to maintain an optimal shape for plant growth. Finally, the operator has the possibility of inserting additional agronomic activities to be performed or already performed to keep track of all the interventions performed on each individual tree in the field.

2.3.9 Crop Yield Forecasting

Scope, objective, and functionalities

In the crop management system, it is important to have production estimates and a comparison with the data of previous years to evaluate the progress of production and the yield of the crop. These data, crossed with prices, provide further information on earnings forecasts.

The data that form the input are:

- Those entered by the operator upon successful harvesting, indicating the quantity and quality of the hazelnuts;
- Observations on the number of clusters observed by human operators and of mean number of nuts per cluster, as trait strictly related to the genotype (kind of cultivars grown). This input is given by monitoring of representative trees properly selected through the orchard;
- The automatic detections of the "Remote Sensing" system regarding the "Fruit Detection" task.

Integration schema

Figure 16 shows the integration scheme of the "Crop Yield Forecasting" subsystem:

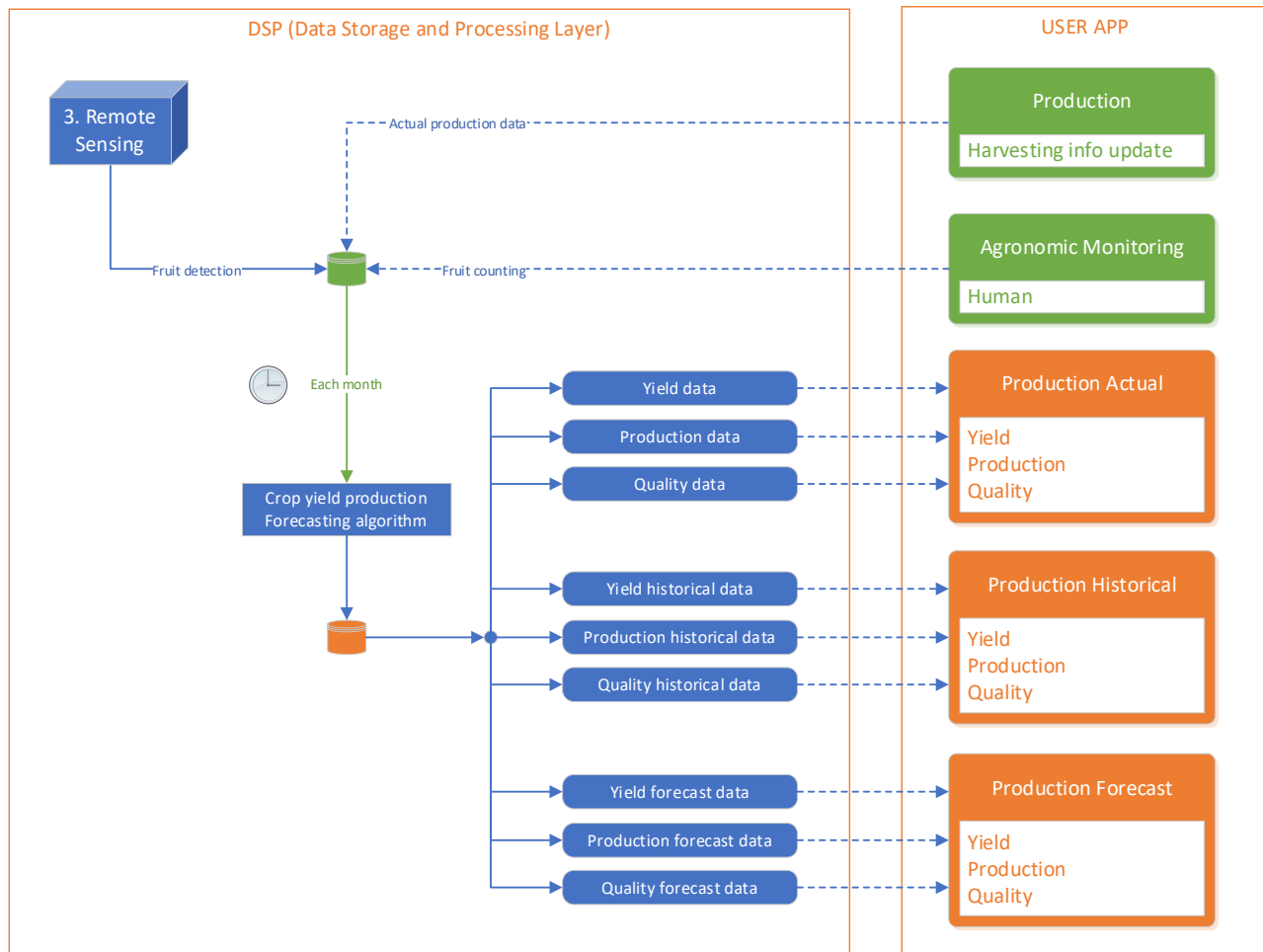


Figure 16. Crop Yield Forecasting subsystem, integration scheme.

The “Crop Yield Forecasting” subsystem is responsible for providing information about the field production, for the current harvest, the history of past harvests and indications on possible future harvest. The forecasts are made by applying an agronomic calculation model that is part of the project validation activities. The data useful for the estimation of the harvest are the human observations in the field, the production history, and the output of the “Fruit Detection” task of the “Remote Sensing”.

The integration work, in this case, consists in the implementation of the Forecasting algorithm based on the proposed agronomic model. Differently, the display of the history and the current harvest will be based exclusively on the recovery of the data stored in the database.

In detail, the database hosts data relating to fruit count, some acquired automatically through "Remote Sensing" and others entered via the application interface. Furthermore, upon completion of a harvest, data on production, quality and actual yield are entered. Starting from this data, the production forecasts are updated every month based on the new data and the results obtained are stored in the database. In the

application interface, graphs with forecast data will be available, compared with real production data, both for the current year and for previous years.

2.3.10 Map Layer

Scope, objective, and functionalities

All spatially explicit (aerial) datasets are stored as multi layered georeferenced `TIF` files. In addition, the spectrally enriched point clouds of the trees - taken by the UGV - are stored in `LAZ` format. The reconstructed tree geometry is stored in `PLY` format for visualization purposes and `GEXF` format for pruning optimization. Metadata for each final product (acquisition date etc) which are necessary for successful visualization in the map widget are stored in the database collection "files".

A Google-Map widget is used, with the appropriate API, to show the geo-referenced results of the processing in the user interface. The map consists of a basic cartography, that is the satellite layer of the field, over which other layers are placed on top showing the `GeoTIFF` images of the `NDVI` vegetative index, the soil moisture or the distribution of pest and disease in the field. Through these views, it is possible to immediately understand which areas of the field is affected by problems.

Furthermore, for more detailed information, a layer can be superimposed on the map which, starting from the `GeoJSON` data format, shows the `Shapefile` including heights and canopy sizes associated with the individual trees. In the user interface, in addition to the map widget, a `JavaScript` widget (`three.js`) is used inside the detail view of the single tree to show the tree geometry 3D view.

Integration schema

Figure 17 shows the integration scheme of the "Map Layer" subsystem:

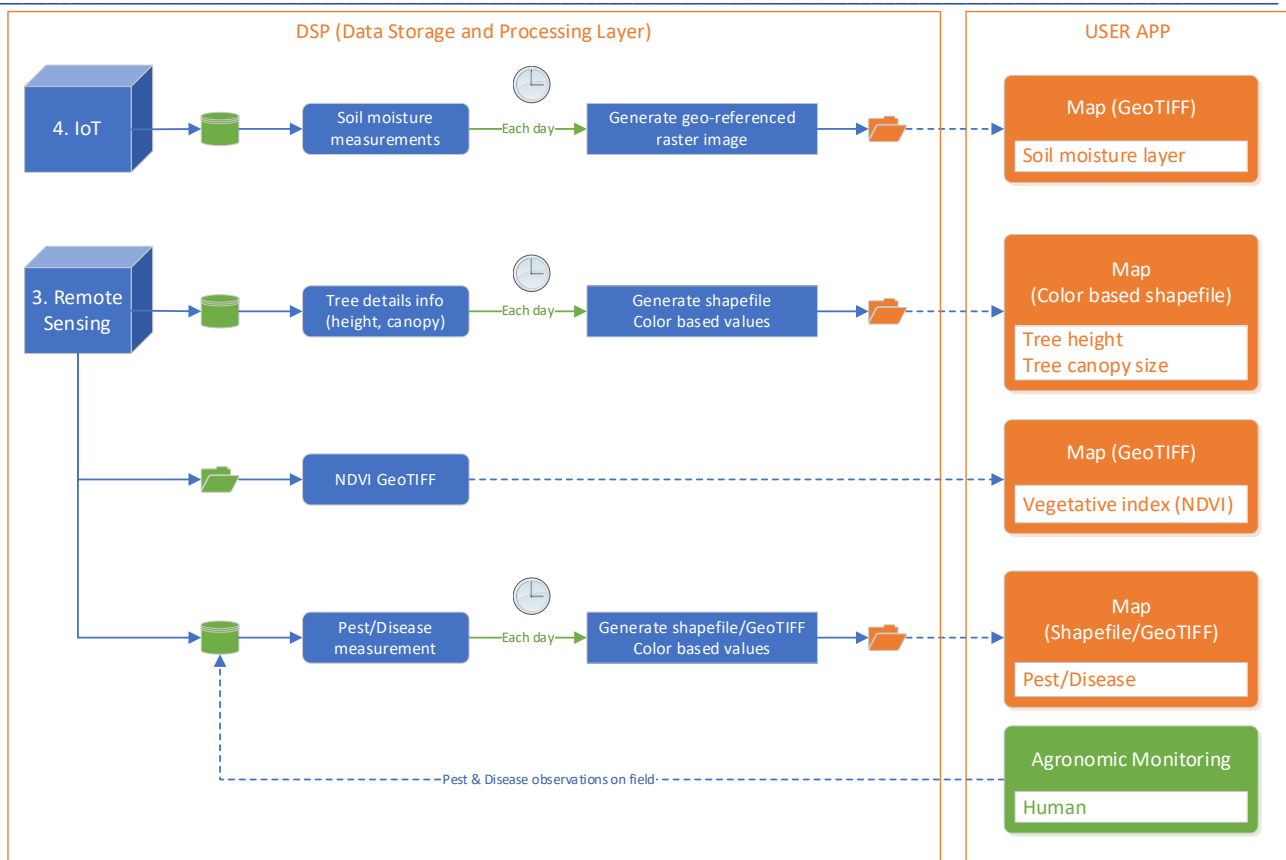


Figure 17. Map Layer subsystem, integration scheme.

The "Map Layer" subsystem is responsible for generating elements that can be used in the user interface for georeferenced viewing on a map of:

- Information from the IoT sensor network;
- Cartographic layers from image processing chains;
- Synthetic data produced downstream of the processing chain.

The work of integration, in this case, consists in regularly scheduling algorithms to produce the cartographic layers, and in geo-referencing the calculated synthetic indicators on the map.

In detail, various views on map are available in the application interface that show georeferenced data relating to the field and individual trees in a simple and immediate way.

The soil moisture data detected by the sensors will be used for the generation of a specific layer.

With the data acquired by UAV and UGV, cartographic layers are produced, in which it is possible to view:

- Height of trees;
- Size of the crown;

- Vegetative index of the field.

There is also a layer showing information on damage and disease of individual trees that is generated using both data acquired by sensors and observations collected manually.

2.4 Design validation

The integration of the remote sensing data processing in the SCADA architecture is challenging, because during the field campaigns a heterogenous mixture of data types is acquired. There is image data, which can be illustrated as spatially explicit data in the frontend after many pre-processing steps. There is lidar data, which is characterized by massive data volumes and also rather difficult to visualize. Then there is ground truth data, which is usually tied to the geographical coordinates of the tree that was measured on the field. Therefore, all raw data that is collected during the campaigns requires at least some amount of processing before it can be illustrated in the frontend of the SCADA system. These processing steps are often highly intricate, and involve complex workarounds for data processing issues, training of advanced machine learning models and the use of proprietary APIs and software packages to create the orthomosaic. This is very challenging from an integration perspective, since the input and the output of the remote sensing campaign is known, but the exact details of the processing chains undergo many iteration steps.

To facilitate the communication of the design of the processing chains, visual workflow diagrams were produced and distributed by the partners. During the meetings, the functionality of the SCADA application was demonstrated, and the data structures produced by the remote sensing processing chains were outlined. A fundamental aspect of the integration of the remote sensing processing chains was the definition and implementation of a proper database design. The database scheme allows for a very flexible definition and handling of processing chains which allows for adding new algorithms and functionalities on demand. The frontend can then just query measurements from the database, while the processing chains that are called on the backend are irrelevant to the mapping and data visualization widgets.

This chapter 2 has presented the integration strategy and project. The general scheme of the system was illustrated, and a bottom-up model was applied for the development and integration of the subsystems.

For each subsystem, the purpose and interconnection scheme were described, providing details on the interfaces used and the operation of specific blocks.

The above has undergone a process of verification and validation by the PANTHEON project partners. SIGMA, as leader of WP6 (Integration, Testing and Validation), has proposed solutions regarding the aggregation of the developed components, with automation mechanisms, where appropriate.

The partners, leaders in the development of subsystems and agronomic models, proposed revisions and improved the design of what was proposed, contributing to the drafting of the reference schemes. The production of the integration diagrams led to interesting points for discussion in the choice of the most appropriate path.

The next chapter will describe the deployment of the system infrastructure, specifically the installation and configuration elements present on the servers and robots used.

3 System Infrastructure Deployment

Downstream of the development of the integration plan, all components are installed, configured, and tested to complete the integration into the SCADA system infrastructure. The main systems involved will be described in the next paragraphs, which are:

- DSP Server;
- DCP Server;
- UAV;
- UGV.

The scheme in Figure 18 shows a synthesis of the composition of the systems:

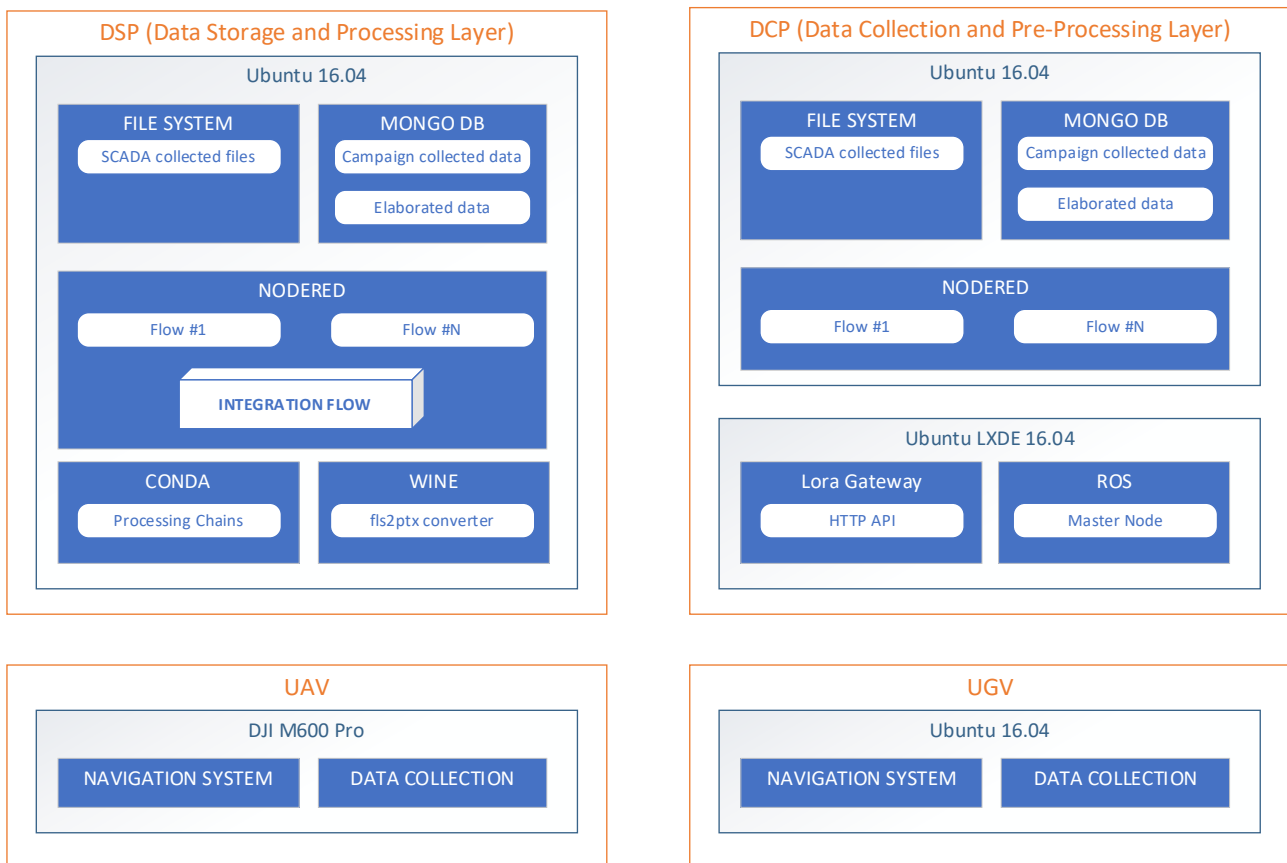


Figure 18. System infrastructure integration overview.

The philosophy behind the composition of the System is to make extensive use of operating environments, languages, libraries, interfaces, software, and hardware components that are widely used, open-source and compliant with standards.

On the one hand, this approach involves a greater effort in the integration task, given the number of third-party components to manage, but at the same time the selection of well-tested and standard components ensures the low risk of integration.

The other advantage of this approach is the ability to focus development efforts on the specific objective tasks of the PANTHEON Project, rather than on the basic components.

For example, the operational software baseline is uniform throughout the system and it includes the use of the `Linux Ubuntu` operating system (in the `16.04 LTS` version, the last one released at the kick-off time of the Project), the `ROS` environment (in the `Kinetic Kame` version because it is primarily targeted at the `Ubuntu 16.04`), the `MongoDB` database and the `Python` environment.

The following paragraphs provide the details of the integration work carried out in the systems indicated above.

3.1 DSP (Data Storage and Processing) Server

The DSP server is the central element of the system, playing the role of data collector, processing, analysis, and application back-end.

The operating system installed on the machine is `Ubuntu 16.04` in `LTS` (Long Term Support) version, selected in the initial phases of the project, following the guidelines agreed with the project team [10]. The selection of this version made it possible to maintain the same system throughout the project: in fact, the `LTS` version is still currently supported after several years.

`Ubuntu` is one of the most popular `Linux` distributions, which guarantees a broad support of the community: it also has a level of usability that makes it immediately accessible to all members of the project team.

The `Desktop` version will be used throughout the development phase, knowing that the final system can then seamlessly migrate to the `Server` version, which is more suitable after the final release.

A special structure of folders and subfolders has been configured on the machine for the rational storage of files acquired in the field. For storing all other data, an instance of the `MongoDB` database was pulled.

`MongoDB` is a non-relational, document oriented `DMBS`. It was chosen based on the operational scenario of the PANTHEON SCADA System, which is a scenario that partly follows the IoT environment, where it is necessary to manage heterogeneous data, perform big data analysis and ensure scalability after release.

Details regarding the definition of the folder structure and database collections can be consulted in Deliverable D3.2 (Data Management) [11].

The `Node-Red` programming environment was installed on the machine, used for the creation of automated flows, allowing the various components and tasks of the system to be aggregated and interfaced. There is also the `Python` environment based on `Conda` environment for the execution of the `Processing Chains`. The central server hosts the modules developed specifically for the PANTHEON project in relation to the Work Packages assigned to the various partners. In particular:

- The user application developed ad-hoc for the PANTHEON system. Its full description can be found in the specific development document [12];
- The "Global Planner" algorithm that generates the path that the UGV platform must automatically follow during the data acquisition mission;
- The "Pruning Management System" algorithm which automatically suggests which branches to prune by analyzing the 3D reconstructions of the trees;
- The Processing Chain algorithms for processing the collected data.

3.1.1 Node-RED

`Node-RED` is a flow-based visual programming tool that allows you to wire together hardware devices, sensors, APIs, and online services as part of the Internet of Things. It runs on `node.js` and provides a browser-based flow editor; the flows created in `Node-RED` are stored using `JSON` format.

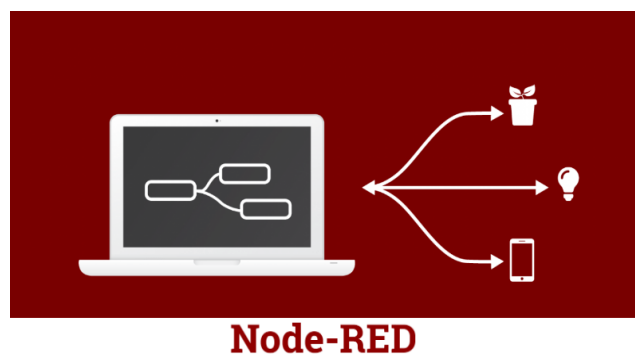


Figure 19. Node-RED flow-based programming tools.

The tool, originally developed by IBM, is open-source, and it complies with the selection criteria of PANTHEON project tools.

Figure 20 shows a screenshot of the node-red operating console, which allows you to implement the execution flows. The editor is quite intuitive: on the left there is the palette where you can pick from a wide range of nodes which can be used to create `JavaScript` functions, on the right you can find the debug, info, setting and help bar. Then flows can be deployed to the runtime in a single-click.

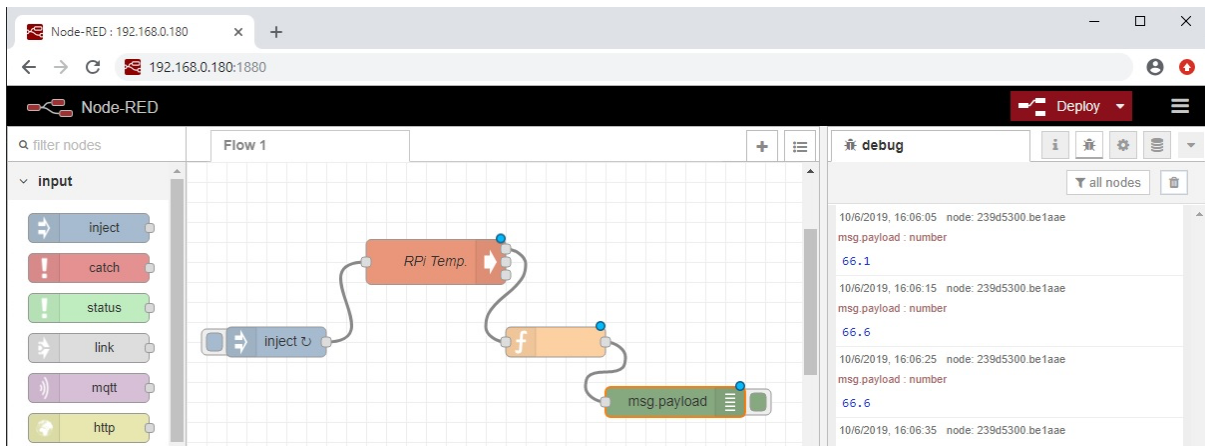


Figure 20. Node-RED development console.

For the PANTHEON project, Node-RED come in handy to manage and automate a lot of different tasks on the server side like the storage and elaboration of new data from the UGV and UAV campaigns, from the ROS node and allows the communication between IoT sensor, database, and the web application.

The tool is open to third-party developers who can expand the functionality by implementing new nodes to be used within the flows. Table 4 describes some of the most used nodes in the PANTHEON integration process:

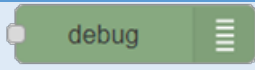

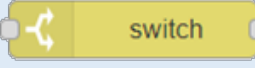
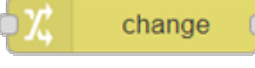
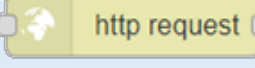
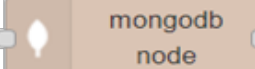
| Common Nodes | Description |
|---|---|
|  | Displays selected message properties in the debug sidebar tab and optionally the runtime log. |
|  | A JavaScript function to run against the messages being received by the node. |
|  | Route messages based on their property values or sequence position. |
|  | Set, change, delete or move properties of a message, flow context or global context. |
|  | Sends HTTP requests and returns the response. |
|  | A simple MongoDB output node. Can manipulate a chosen collection. |

Table 4. Description of common Node-RED nodes used.

One example of the flows integrated is the alarm check flow that controls the temperature value from the IoT sensors and monitors if it goes below zero, shown in following Figure 21:

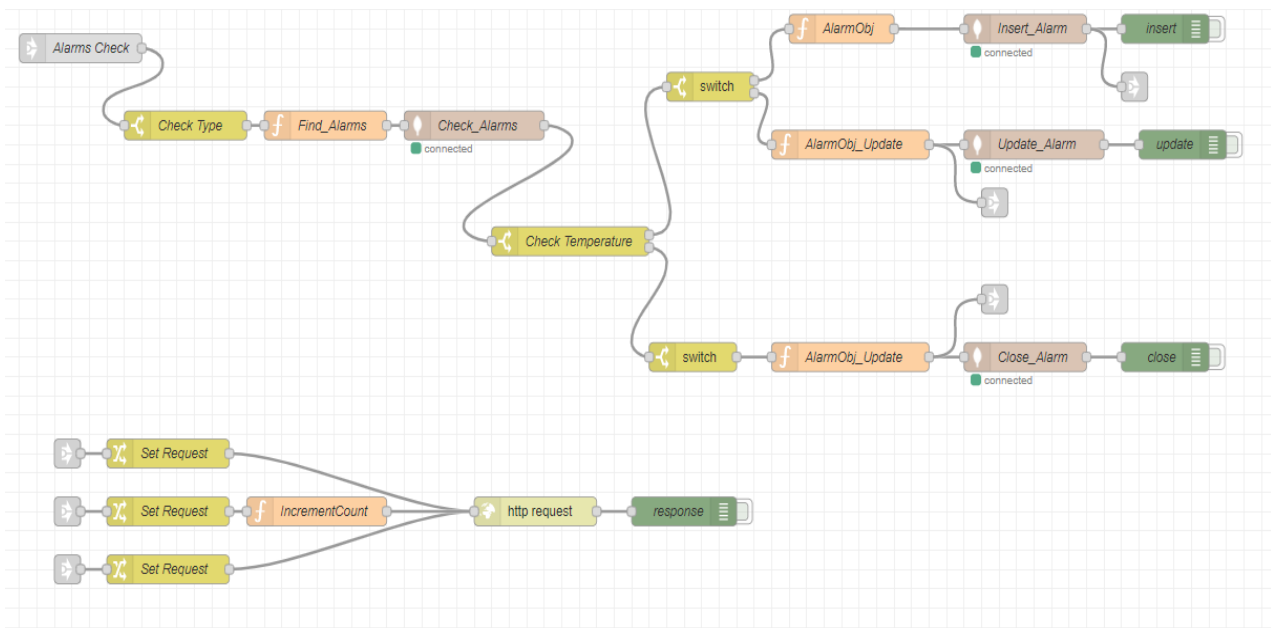


Figure 21. PANTHEON integration flows example

The flow gets input from a topper flow that processes the measurements from sensors and saves them on MongoDB; the first node filters all the type of measurements to get only the `temperature_min`, then the function node creates a query to check if there are any alarms already active and send the query to the database.

After that, another switch checks if the `temperature_min` is below 0 or not, if it is `true` the flow proceeds to either `Insert` or `Update` the alarm, based on the first query to the database, if the temperature is over 0 degrees, the flow closes the alarm if it is opened.

In all three cases the flow also sends an `http request` to the `backend` to update in real-time the web application.

3.1.2 User Application

The developed user application allows you to interact with the SCADA system and exposes the operational interfaces for end users, namely agronomists, farmers, system managers, etc.

Figure 22 shows the logical structure of the application, where it is possible to identify specific middleware, libraries and runtime environments used in the implementation, which have been installed and configured appropriately on the DSP server machine.

User Application Architecture

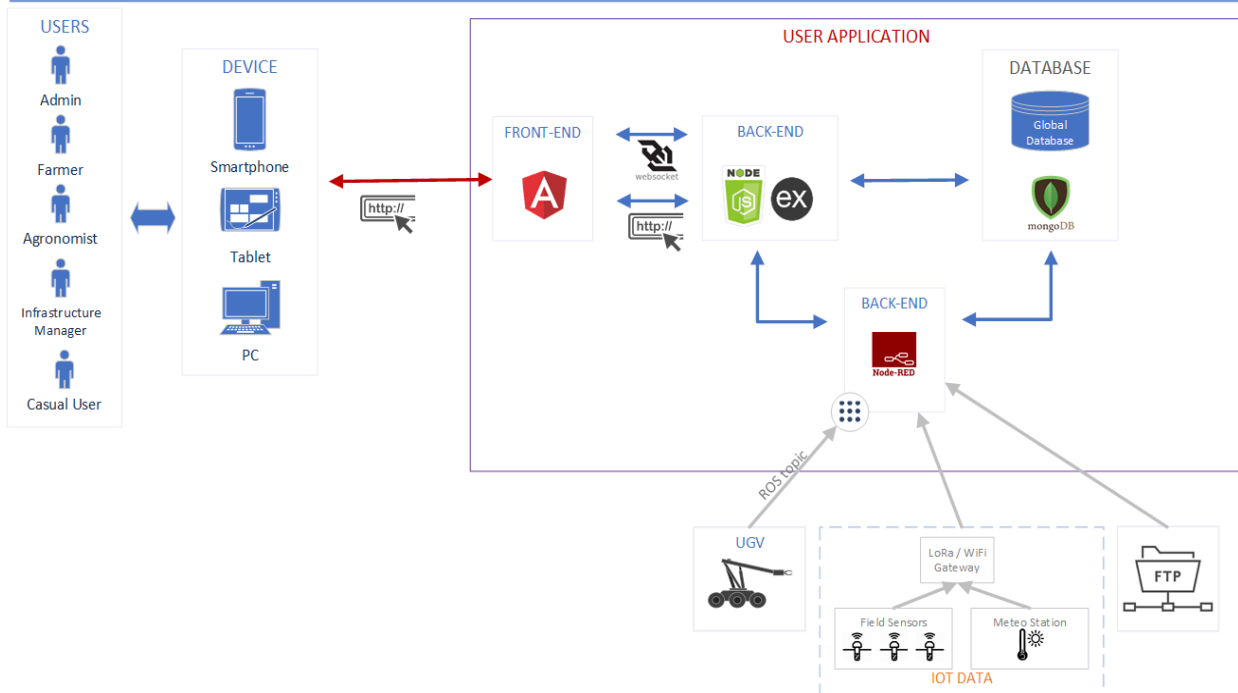


Figure 22. User application architecture.

The main source of data that the user application has access to is the system database (MongoDB), and in this scenario the backend takes care of implementing all the data management features.

Therefore, the backend is a fundamental component because through it you interact with all the data stored by the SCADA system. Its main function is to provide resources to the user interface (frontend) allowing it to view, enter or modify the data stored in the database.

In addition, the backend acts as a WebSocket Server enabling you to view on the user interface the relevant events that occur in the field in real-time. These events can be generated by actions performed manually by other operators using other clients (e.g.: smartphone, tablet) or by automatic events generated by the processing of the flows implemented in Node-RED.

The backend was created using node.js as an interpreter for the server-side JavaScript and TypeScript language, the Express library to provide entry points to the Web API and the Express-ws library to provide entry points for the WebSocket.

Process manager PM2 is used to keep the backend process node.js active on the server.

The frontend module provides a user interface which an operator can access, both via fixed and mobile devices, to view all the information on the status of the trees and on the events that are occurring in the field. Furthermore, the data acquired by remote sensors and the IoT network and further widgets, such as

the weather forecast, which support agronomists in the decision-making process of the actions to be taken in the field, are also shown through maps and graphs.

For the development of the frontend the Angular framework was used with the TypeScript language, which is also based on `node.js`. Moreover, as an additional library for the UI, it was used Nebular which provides both support components for the visualization, and libraries for user authentication and page protection.

To release it on the server, you need the support of a web server: it has been chosen to use Nginx.

3.1.3 Global Planner

The mission instructions are generated by the global planner and stored in CSV files on the DSP Server. The UGVs tasked to perform the campaigns, download these files, and start the execution of the task manager which will oversee carrying out the related tasks.

The global planner is responsible for computing the paths that the robots need to follow to perform any agronomical activity as well as the required action that the robots need to fulfil to achieve the desired objective. As input from the user's interface, the global planner requires the list of the farming operations that must be carried out, the map of the field in which the operations will be performed and the IDs of the trees which are the objects of the campaign.

The global planner requires the following components:

- MATLAB libraries;
- IBM ILOG CPLEX.

3.1.4 Pruning Management System

The pruning management system is responsible for providing to the user a possible pruning for a specific tree. As input from the user's interface, the pruning management system requires the ID and the typology of the tree that the user desires to prune.

The pruning management system will return a GEXF file with the graph representation of the original tree, a list of IDs of the nodes and the edges associated to the suggested branches to prune with respect to the GEXF file, and a PLY file showing the original tree with the pruning suggestions highlighted in green.

The pruning management system requires the following components:

- MATLAB libraries;
- Python library 'NetworkX'.

3.1.5 Processing Chains

The entire functionality of the remote sensing processing chains is wrapped in a Python 3.6.9 based Conda environment. With two notable exceptions, open-source libraries and packages were used to implement the desired functionality. The Agisoft metashape-api is a proprietary package which is used to create the orthomosaics. Additionally, a proprietary SDK from Faro was used to convert the files from the laser scanner into PTX files. The list of other required libraries mainly consists of a few very common Python libraries to facilitate the image processing, namely `numpy`, `opencv`, `pillow`, `scikit`, `networkx` and `rasterio`. The management of the file storage as well as the metadata is done with MongoDB.

All files used in the processing chains are stored in a common folder `‘/home/data/mongodb/robots/<id_platform>’` followed by a campaign identifier. The folder `‘products’` contains final campaign outputs such as harmonized orthomosaics. The folder `‘reports’` contains processing related metadata. All acquired data is stored in the folder `‘sensors’`, which is followed by a sensor specific string handle (Figure 23).

```
(pantheon_env) pantheon@bartolo:/home/data/mongodb/robots/UAV/campaigns/camp_2020-07-12_14-58-00$  
├── products  
│   └── validation  
├── report  
└── sensors  
    ├── Sony_a5100_UAV  
    ├── Teax_ThermalCapture-2.0  
    └── Tetracam_MCAW-6
```

Figure 23. A representative campaign folder.

3.2 DCP (Data Collection and Pre-Processing) Server

The DCP component represents that part of the infrastructure located in the field that allows to perform data collection and pre-processing activities.

PANTHEON architectural proposal provides for the presence of a server machine that locally reproduces a subset of the activities performed by the central DSP, to ensure local operation of the system both in the presence and absence of a communication network.

Currently, centralizers have been installed in the field that allow the detection of environmental parameters (weather and soil humidity / temperature) and the monitoring of UGV missions in real-time.

The diagram in Figure 24 shows, from a logical point of view, the current infrastructure deployed in the field:

Precision Farming of Hazelnut Orchards (PANTHEON)

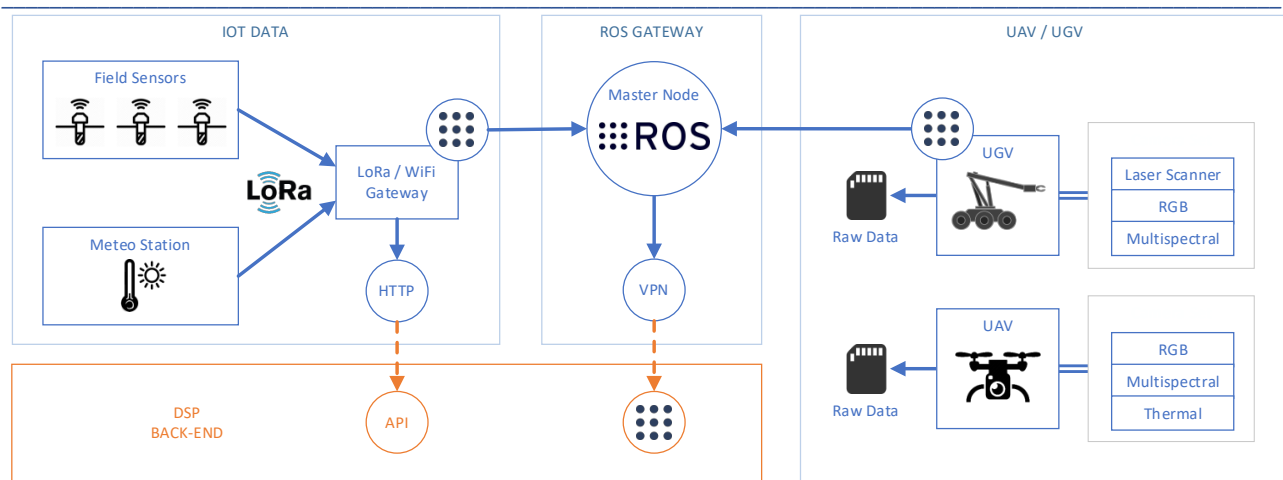


Figure 24. DCP deployed infrastructure

A Mesh Wi-Fi network covers the experimental field and server warehouse, implemented using UniFi e airMax technologies.

The weather station consists of a set of sensors for monitoring weather conditions such as air humidity, air temperature, air pressure and rainfall. Soil sensors are responsible for monitoring of soil properties such as soil moisture and soil temperature; the sensors are inserted in the soil at two different depths (15cm and 40cm respectively). Both weather station and soil sensors communicate using LoRa protocol.

The LoRa/Wi-Fi gateway is implemented on a Raspberry Pi 3B+ board (Figure 25) and it allows the data acquired from the LoRa network to be transferred over Wi-Fi, in order to be transmitted via the Internet to the DSP server.

The LoRa network protocol allows to implement the following operational flow:

- Data coming from soil and weather are collected by the weather station and from the LoRa nodes;
- Data is sent to the Raspberry that stores them in a queue system;
- Data is sent over the global network to be received and stored by the DSP.

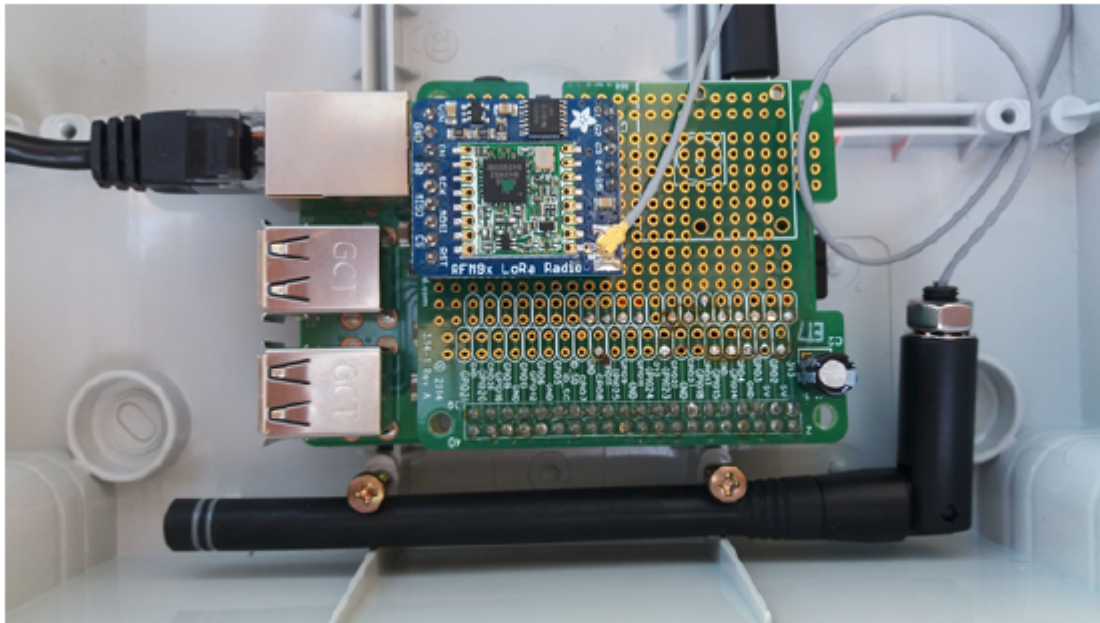


Figure 25. Raspberry board.

The ROS gateway is also implemented through a Raspberry Pi 4 Model B board, independent from the first, on which the ROS Master node runs (Figure 26). This component allows the publication of UGV information and allows the DSP server to subscribe to the corresponding Topic. Communication between remote ROS nodes is guaranteed using a standard VPN (OpenVPN).

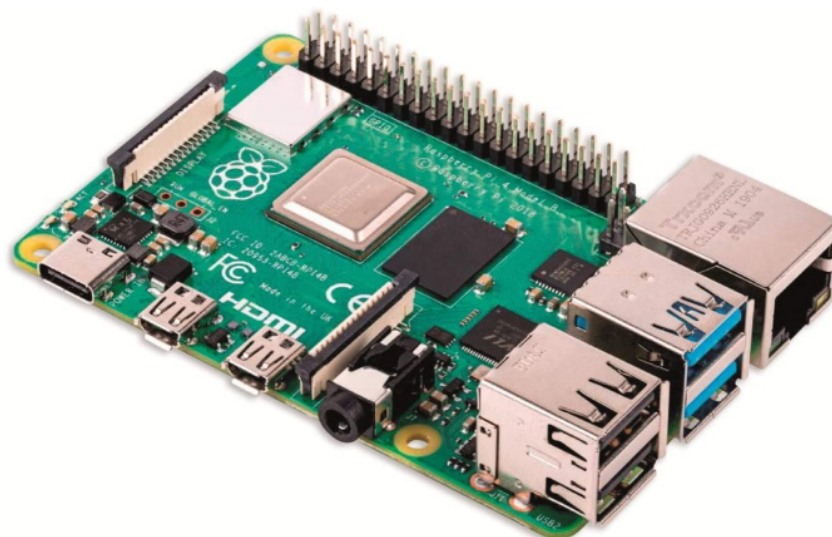


Figure 26. Raspberry Pi4 Model B board, ROS gateway

3.3 UGV (Unmanned Ground Vehicle)

The UGVs are equipped with a NUC NF697-Q170 in which Ubuntu 16.04 is installed. Figure 27 shows the outer equipment installed in the Sherpa HL robotic platform.

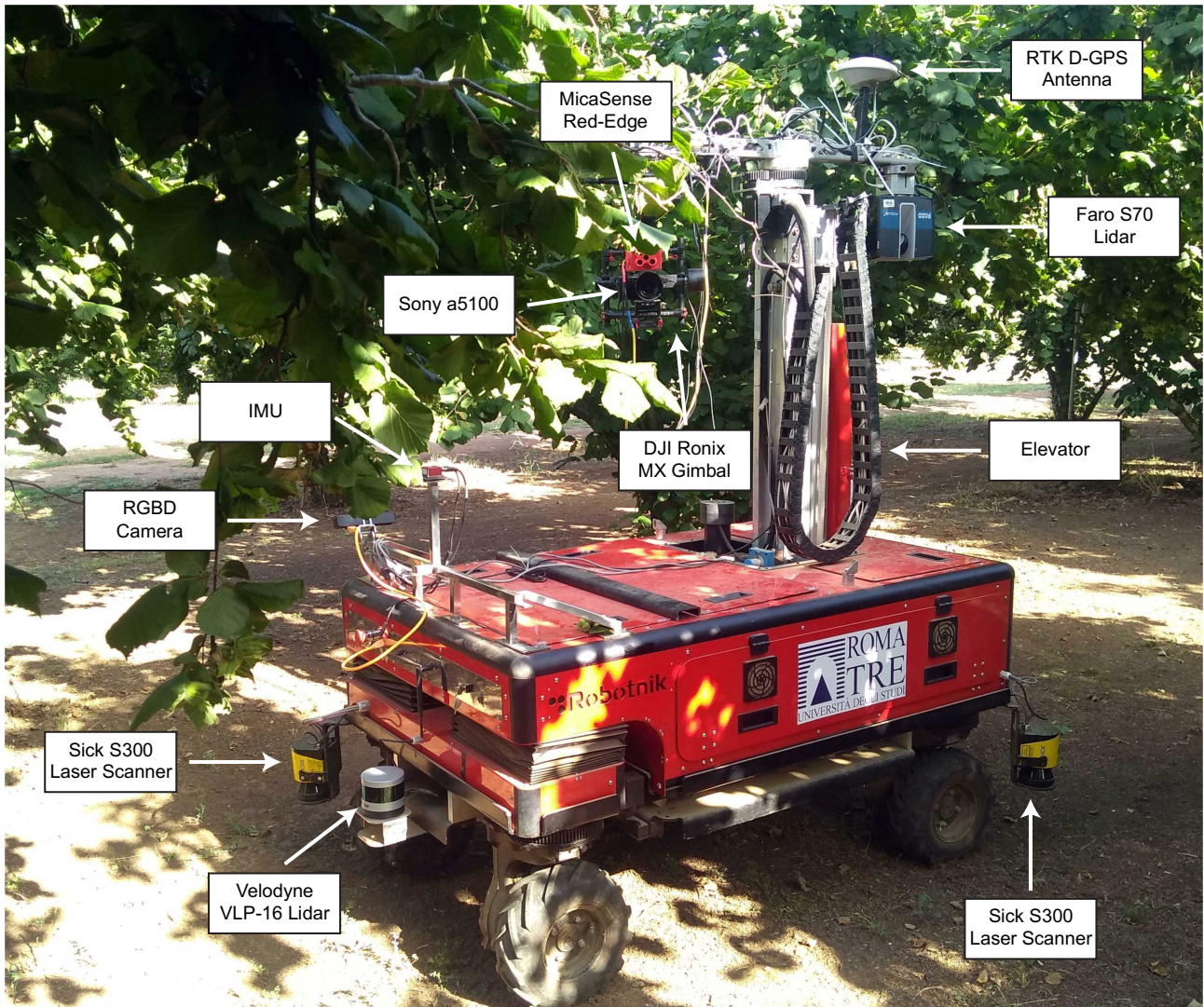


Figure 27. SHERPA HL robotic platform R-A outer parts.

Furthermore, a ROS architecture that is the result of the cooperative operation of several nodes are installed on the UGVs. In particular, the robotic platforms are equipped with the ROS Distribution Kinetic, robot specific software, which are provided in 3 stacks (metapackages), and developed ROS package:

- `rbsherpa_hl_common`: packages shared between the robots and simulation stacks, i.e., robot description, navigation, etc.;
- `rbsherpa_hl_robot`: includes all the packages required for the real robot control;
- `rbsherpa_hl_sim`: allows the simulation of the robots and its sensors in Gazebo;

- `sherpa_ros`: contains the task manager and script to connect the robot to the DSP and DCP;
- `scanning_controller`: contains the connection to the drivers and devices involved in the scanning process on the robot R-A;
- `spray_controller`: contains the connection to the drivers and devices involved in the spraying process on the robot R-B;
- `management_suckers`: includes the packages required for the suckers' management procedures;
- `faro_scanner`: contains the drivers for the `Faro_Focus S70` sensor equipped on the robot R-A;
- `ugv_rgb_camera`: contains the drivers for the `Sony a5100` camera equipped on the robot R-A;
- `ugv_ms_camera`: contains the drivers for the `MicaSense RedEdge-M` camera equipped on the robot R-A.

Further details can be found in Deliverable D3.1 (Robotic Prototypes [13]).

3.4 UAV (Unmanned Aerial Vehicle)

The UAV model used is the `DJI M600`. This model is equipped with a `DJI A3 Pro` triple-modular redundancy system and advanced intelligent flight functions, a `D-RTK` system which allows high accurate positioning and a gimbal `Ronin MX` with 3 sensors installed. All these components are controlled by an on-board computer `DJI Manifold` added on the top of the aircraft.

The `A3 Pro` flight controller provides three `GPS` modules and `IMUs` which add triple modular redundancy to reduce the risk of system failure. This system is complemented with a `RTK` module which, using a ground station, provides corrected `GPS` signals to improve its accuracy. These features can be all controlled by the on-board computer installed.

Besides the standard equipment provided by DJI, the equipment of the aircraft includes:

- A `RTK GPS` module produced by DJI;
- The gimbal `DJI Ronin MX`, which has been customized by `AERMATICA 3D` to fit 3 camera sensors;
- An onboard computer consisting of a `DJI Manifold`;
- A flight termination system compliant with the Italian legislation requirements and connected with a (optionally armed) spring-based parachute.

3.4.1 UAV Equipment

Figure 28 shows the equipment installed in the DJI M600 drone.

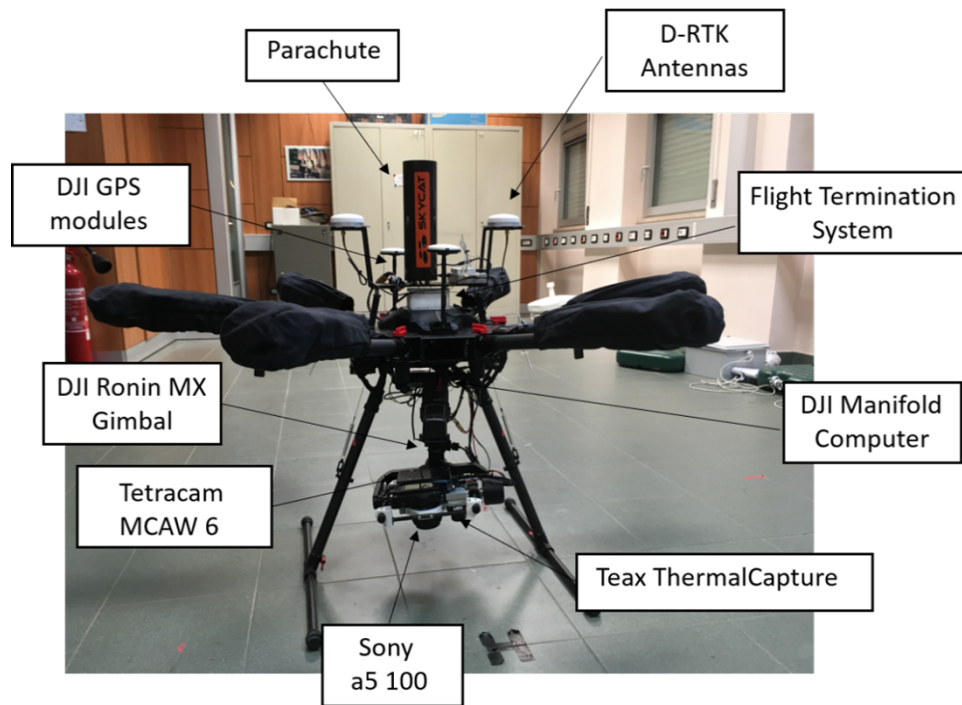


Figure 28. Components of the UAV subsystem.

During the flight mission the UAV subsystem is connected via 2 wireless communication channels with the Radio controller and a system terminator, respectively. A schema of these connections is shown in Figure 29. Additional connections with a workstation have been avoided due to security reasons and interferences created by the system.

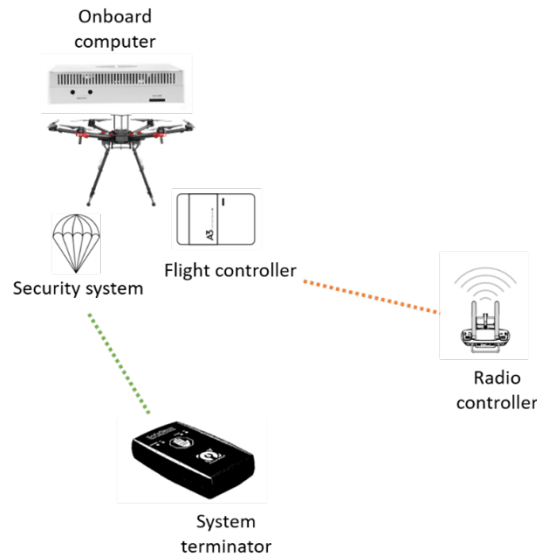


Figure 29. Wireless connections in the UAV subsystem.

The triggers of the sensors are generated by a ROS service. Additionally, the onboard computer communicates with the flight controller via ROS messages monitoring the status of the drone and triggering the sensors at the adequate instants. In this way, the coordinates, the speed, and additional useful information for the creation of the orthomosaic are obtained. With this procedure, the onboard computer can associate the UAV status with the triggering of the sensors (Figure 30) by creating a log file of the mission.

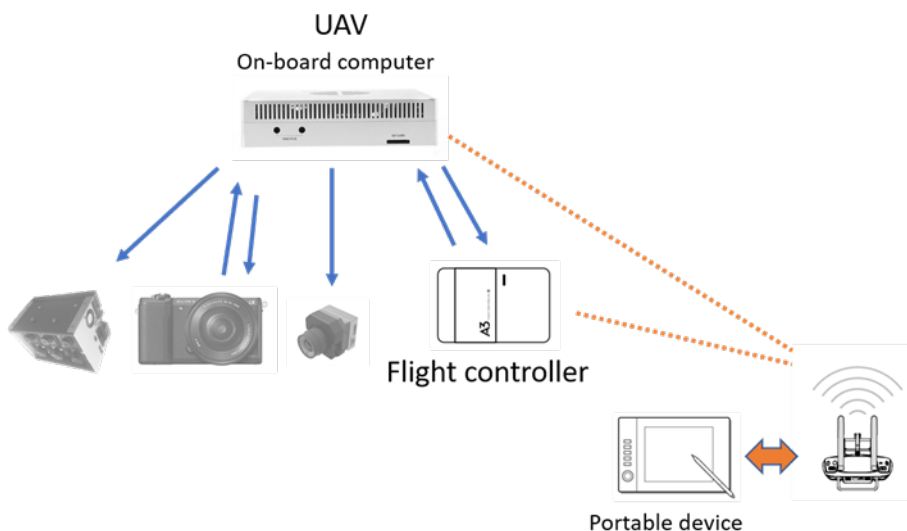


Figure 30. Schema of the connections between sensors, flight controller and portable device.

3.4.2 UAV Post-Processing

The postprocess of the images and the aligning with the UAV information (Figure 31) are not done online due to the constraints provided by the sensors used (no feedback communication provided by the thermal and

multispectral devices) and the limited download bandwidth of the images. These issues are tackled by a postflight operation performed before the uploading of the data to the server.

Given the size of the images generated by the system and the limited bandwidth of the connection with the cameras, the images cannot be downloaded online nor transferred to the onboard unit. All images are stored locally on the sensors and downloaded once the flight is done. Then, they can be uploaded to the server already synchronized with the flight log file. As mentioned, given the limitations by the sensors used (lack of feedback) and the possible failures of the sensors, an additional post-process step has been added to verify the correct functioning of the sensors during the flight. In this way, images from the 3 cameras are matched to the data obtained thanks to the ROS connection to the drone.

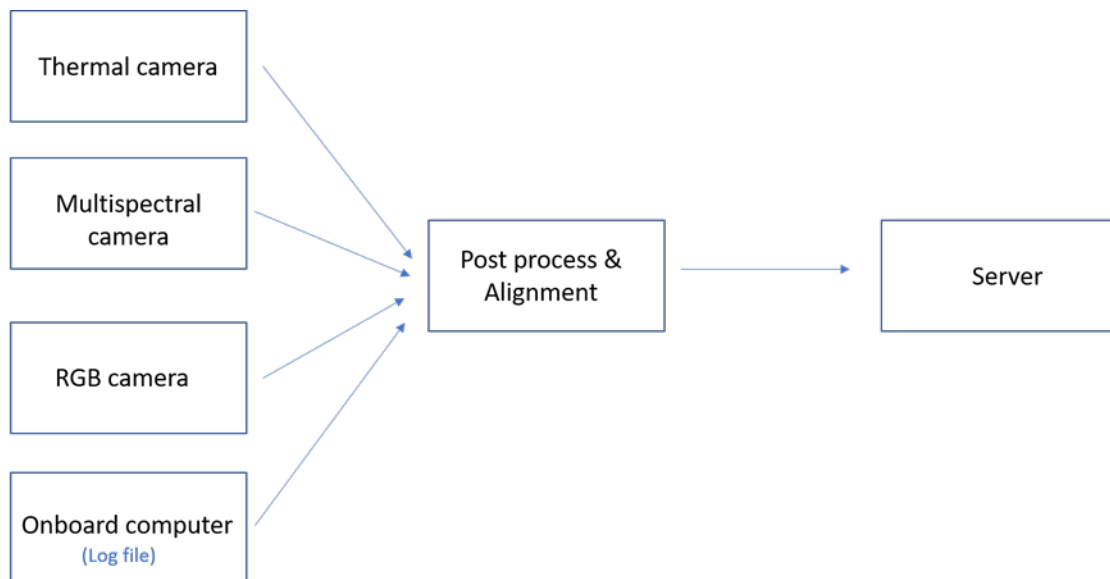


Figure 31. Flight data postprocessing.

4 Field Validation

The field validation consists in making the prototype of the SCADA system developed in a real-world hazelnut orchard operational.

The expected double validation consists both in verifying the functioning and operational effectiveness of the hardware/software infrastructure and in validating the benchmark stability objectives at the start of the project.

4.1 Infrastructure Validation

The validation process is focused on single components and their interoperability. The next paragraphs focus on the validation of the system developed up to now, organized by infrastructural components.

4.1.1 Wireless Network Backbone

Several experiments were carried out to evaluate the performances and the reliability of the WNB deployment. In particular, the following indicators were considered: i) RTT, ii) Latency, iii) Packet Loss, iv) Throughput, iv) Bandwidth and ROS rate. Details regarding the WNB performance can be found in Deliverable D3.3 [9].

According to the experimental collected data the following considerations can be drawn:

- The deployed WNB is a reliable network as the packet loss is less than 1%;
- The long-range point-to-point communication is a very reliable communication link;
- The Multi-hop Wireless network can represent (sometimes) the bottleneck of the network;
- The traffic over the Mesh network should be reduced to the strict necessary in order to guarantee overall good performances.

4.1.2 Ground Robotic Platforms

Several experiments were carried out to evaluate the performance of the unmanned vehicles navigation, monitoring and agronomic intervention capabilities within a real-world (1:1 scale) hazelnut orchard. A detailed description of the theoretical achievements along with videos describing the experimental validation can be found in the Deliverable D6.1 [14].

Briefly, the validation was conducted for the verification of the navigation architecture, the simultaneous localization and mapping, and the global planner capabilities.

4.1.3 Aerial Robotic Platforms

Several experiments were carried out to validate the performance of the aerial robotic platform regarding the monitoring of the different phytosanitary indicators of the hazelnut orchard. Additionally, the theoretical achievements in terms of novel path planning strategies and videos demonstrating the capabilities of the UAV can be found in the Deliverable D6.1 [14].

Briefly, the validation was conducted for the path planning strategies, in terms of adaptability and performance analysis.

4.1.4 IoT Agrometeorological Network

To evaluate the right functioning of the IoT network, the data stream arriving at the central server is observed and verified. The stream is processed through the Node-RED environment, the "Save Measurement" flow (Figure 32) provides the API with entry point `/api/v1/measurement/add` to the IoT gateway that sends the sensor measurement messages.

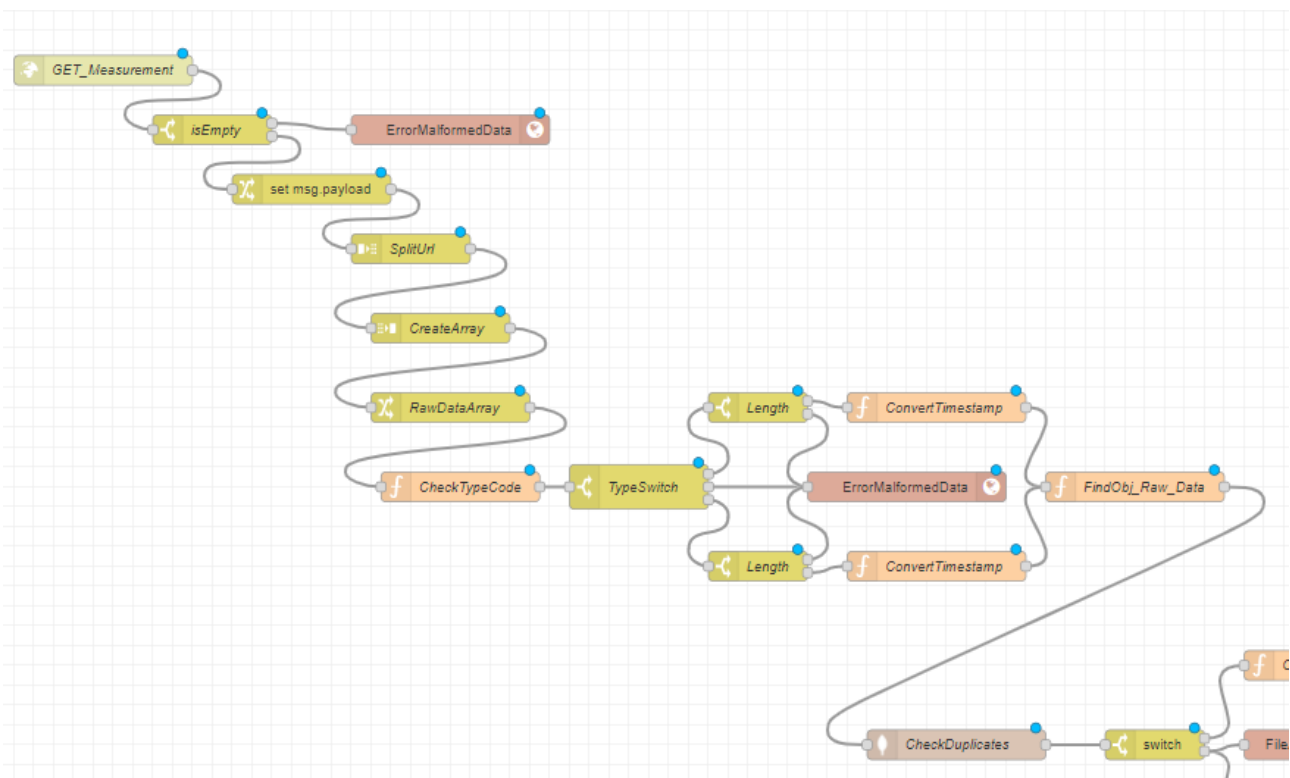


Figure 32. "Save Measurements" Node-RED flow.

The stream is processed and then performed checks on the uniqueness, validity, and conformity of the data. Then, other sub-flows take care of unpacking and storing the data of each sensor in the database.

The storing of the meteorological parameters on the database is performed by the “Save Weather Measurements” flow (Figure 33).

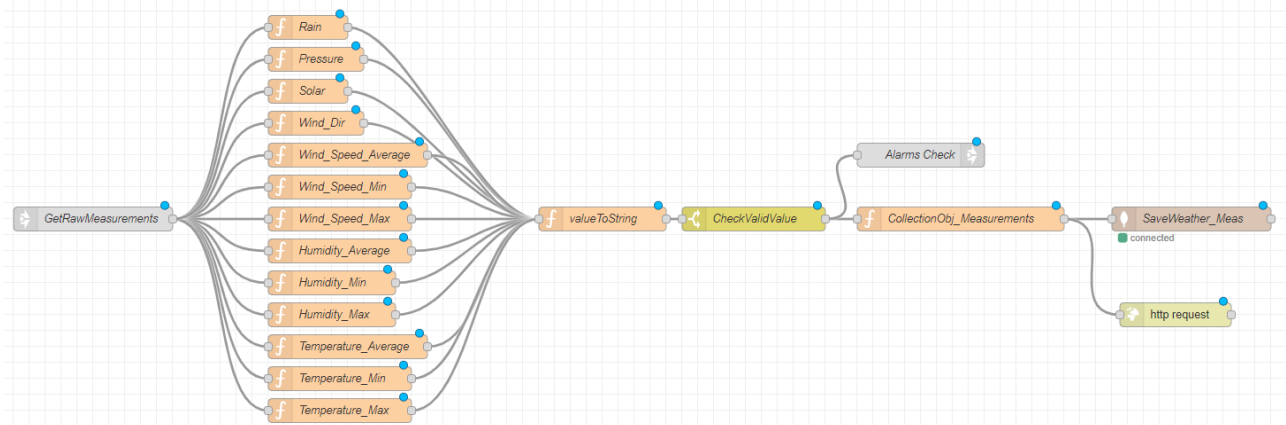


Figure 33. "Save weather measurements" Node-RED flow.

The storing of the soil parameters on the database is performed by the “Save Soil Measurements” flow (Figure 34).

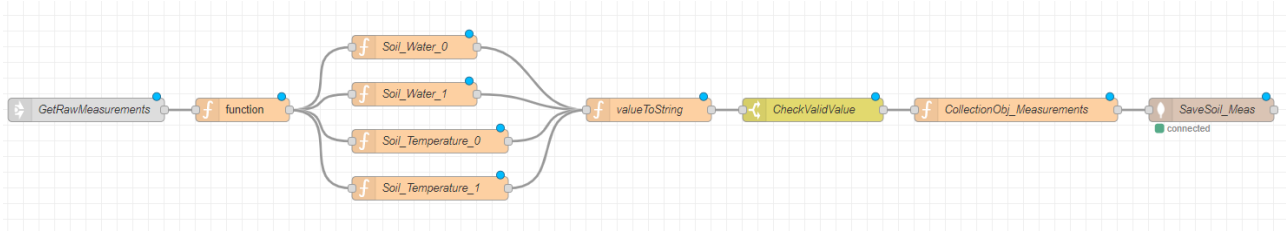


Figure 34. "Save soil measurements" Node-RED flow.

The validation of the transmission and remote acquisition of the data is done by observing the collections of the MongoDB database.

The following Figure 35 shows an example of data in the ‘rawMeasurement’ collection corresponding to the messages sent by the IoT gateway.

Pantheon.rawMeasurements

DOCUMENTS 305.8k ^{TOTAL} 71

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER

VIEW [Icons]

Displaying d

```
{
  "_id": ObjectId("6086b7ad5abb570e67343ae0"),
  "created": "2021-04-26T12:52:05.000+00:00",
  "uploaded": "2021-04-26T12:53:01.971+00:00",
  "type": "soil",
  "id_platform": "TN_08",
  "raw_data": "TN_08,20210426125205,42.2802,12.2982,285.7,2,0.156,12.700,1,1,0.204,12...."
}
```

```
{
  "_id": ObjectId("6086b7ad5abb570e67343adb"),
  "created": "2021-04-26T12:51:05.000+00:00",
  "uploaded": "2021-04-26T12:53:01.762+00:00",
  "type": "soil",
  "id_platform": "TN_07",
  "raw_data": "TN_07,20210426125105,42.2795,12.298,358,2,0.188,13.300,1,1,0.193,12.60..."
}
```

```
{
  "_id": ObjectId("6086b7ad5abb570e67343ad6"),
  "created": "2021-04-26T12:49:05.000+00:00",
  "uploaded": "2021-04-26T12:53:01.527+00:00",
  "type": "soil",
  "id_platform": "TN_05",
  "raw_data": "TN_05,20210426124905,42.2795,12.2986,275.1,2,0.218,15.400,1,1,0.293,13.1..."
}
```

```
{
  "_id": ObjectId("6086b7ad5abb570e67343ad1")
}
```

Figure 35. "rawMeasurements" MongoDB collection.

While the following figure shows an example of data stored in the 'measurement' collection in which the measurements of the individual sensors processed by the central unit are saved in a harmonized manner.

```
VIEW [Icons]
_id: ObjectId("6086b7ad5abb570e67343ae1")
processing_level: "raw"
created: 2021-04-26T12:52:05.000+00:00
id_sensor: "TN08_W_0"
id_platform: "TN_08"
type: "soil_water"
unit: "mm"
value: 0.156

_id: ObjectId("6086b7ad5abb570e67343ae2")
processing_level: "raw"
created: 2021-04-26T12:52:05.000+00:00
id_sensor: "TN08_W_1"
id_platform: "TN_08"
type: "soil_water"
unit: "mm"
value: 0.204

_id: ObjectId("6086b7ad5abb570e67343ae3")
processing_level: "raw"
created: 2021-04-26T12:52:05.000+00:00
id_sensor: "TN08_T_0"
```

Figure 36. "measurements" MongoDB collection.

Another check was performed on the user interface side, verifying the graphic display of the values collected by the IoT sensor network.

The following Figure 37 shows "chart" widgets in which it is possible to have a visual confirmation of the data arriving in real-time from the IoT agrometeorological network.

Precision Farming of Hazelnut Orchards (PANTHEON)

lot sensor data

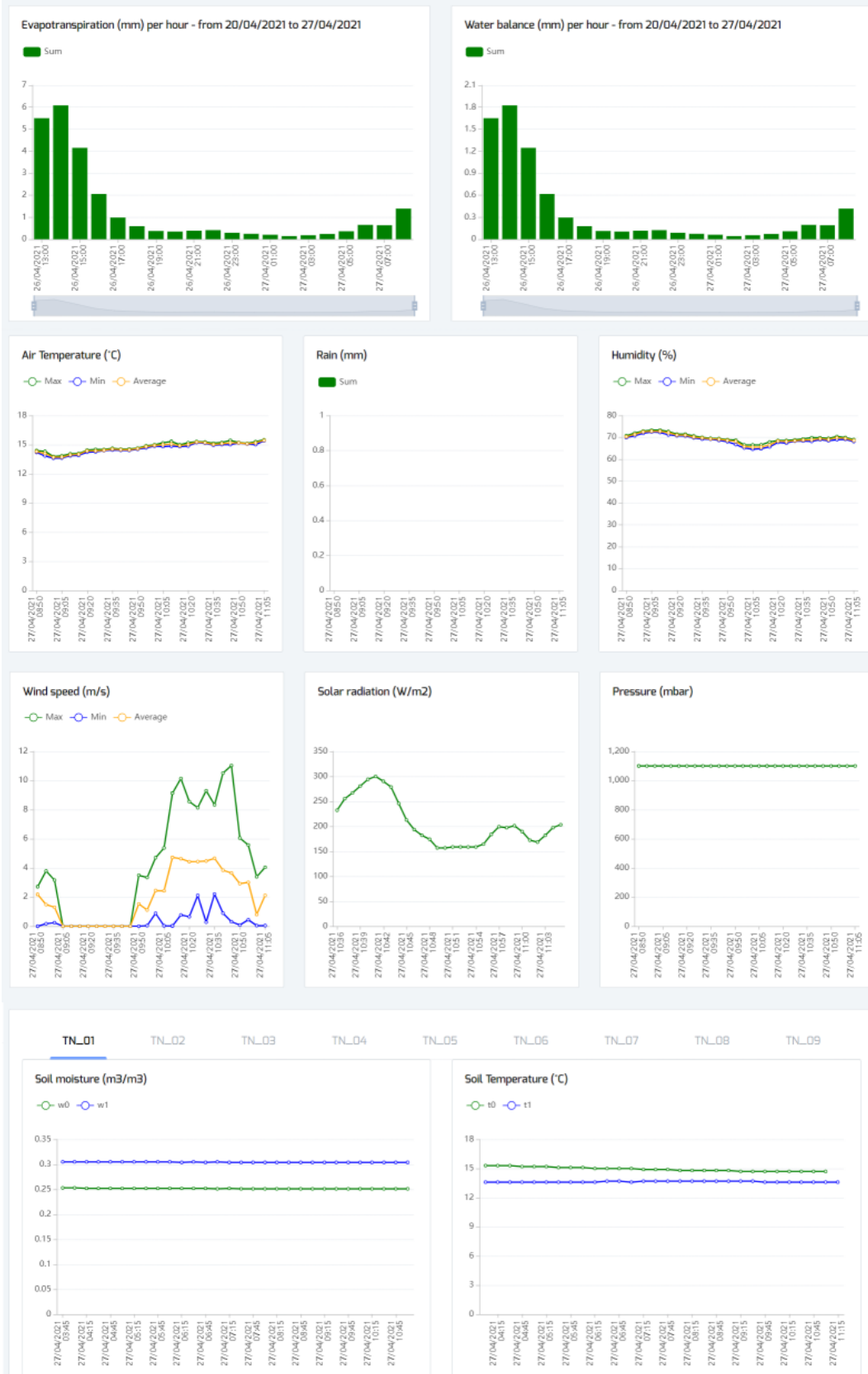


Figure 37. IoT real-time data on user interface.

4.1.5 Software Architecture

The validation of the software architecture is partially verified by the receiving and processing the real-time data from the IoT sensor network. Furthermore, it is validated by the right working of the user interface which shows the data stored in the reference database using graphical panels.

A further step in the validation of the software architecture was performed by verifying the right working of historical data management and alarms management.

From the sensors input, historical measurement data are elaborated. Through the Node-RED environment, the "Historical Data" flow aggregates the data (to be kept for analysis and statistics) on an hourly, daily, weekly, monthly, and annual basis.

In Figure 38 there is an example of a flow that aggregates the soil sensors data.

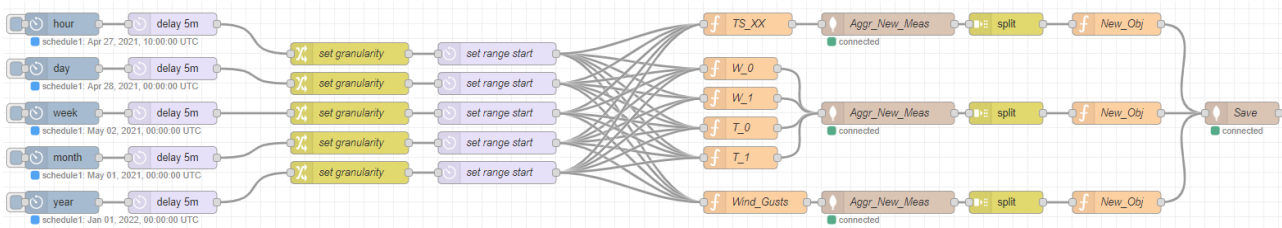
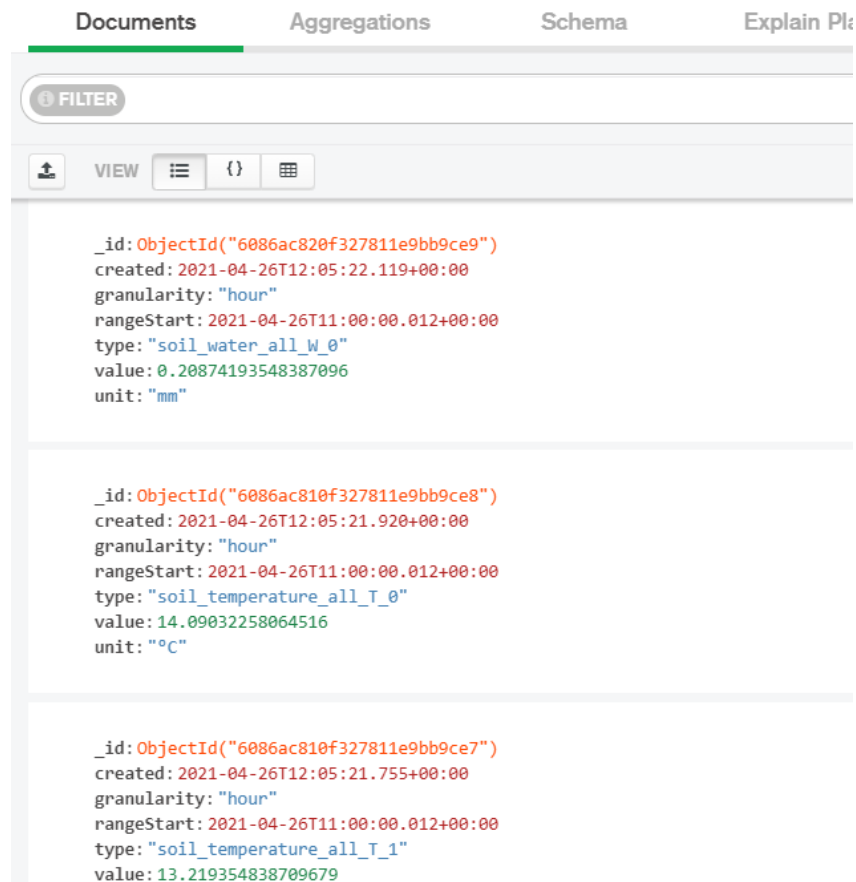


Figure 38. "Soil sensors" historical data, Node-RED flow.

In the next Figure 39, there is an example of data stored in the 'historicalData' collection; it demonstrates how the aggregation made by the system is performed correctly and the result stored on the MongoDB database.

Pantheon.historicalData



The screenshot shows the MongoDB interface for the 'Pantheon.historicalData' collection. It features a navigation bar with 'Documents', 'Aggregations', 'Schema', and 'Explain Pl'. Below the navigation bar is a 'FILTER' button and a 'VIEW' section with icons for list, JSON, and grid views. The main content area displays three document entries, each with the following fields:

- `_id`: ObjectId("6086ac820f327811e9bb9ce9")
- `created`: 2021-04-26T12:05:22.119+00:00
- `granularity`: "hour"
- `rangeStart`: 2021-04-26T11:00:00.012+00:00
- `type`: "soil_water_all_W_0"
- `value`: 0.20874193548387096
- `unit`: "mm"

The second entry has the following fields:

- `_id`: ObjectId("6086ac810f327811e9bb9ce8")
- `created`: 2021-04-26T12:05:21.920+00:00
- `granularity`: "hour"
- `rangeStart`: 2021-04-26T11:00:00.012+00:00
- `type`: "soil_temperature_all_T_0"
- `value`: 14.09032258064516
- `unit`: "°C"

The third entry has the following fields:

- `_id`: ObjectId("6086ac810f327811e9bb9ce7")
- `created`: 2021-04-26T12:05:21.755+00:00
- `granularity`: "hour"
- `rangeStart`: 2021-04-26T11:00:00.012+00:00
- `type`: "soil_temperature_all_T_1"
- `value`: 13.219354838709679

Figure 39. "historicalData" MongoDB collection.

The historical data are extracted from the database via the http API and shown in the user interface. In the user interface, you can change the date range and the granularity in showing data inside the "chart" widgets. The following figures (Figure 40, Figure 41, and Figure 42) show examples of historical data that can be analyzed through the user interface.

Precision Farming of Hazelnut Orchards (PANTHEON)

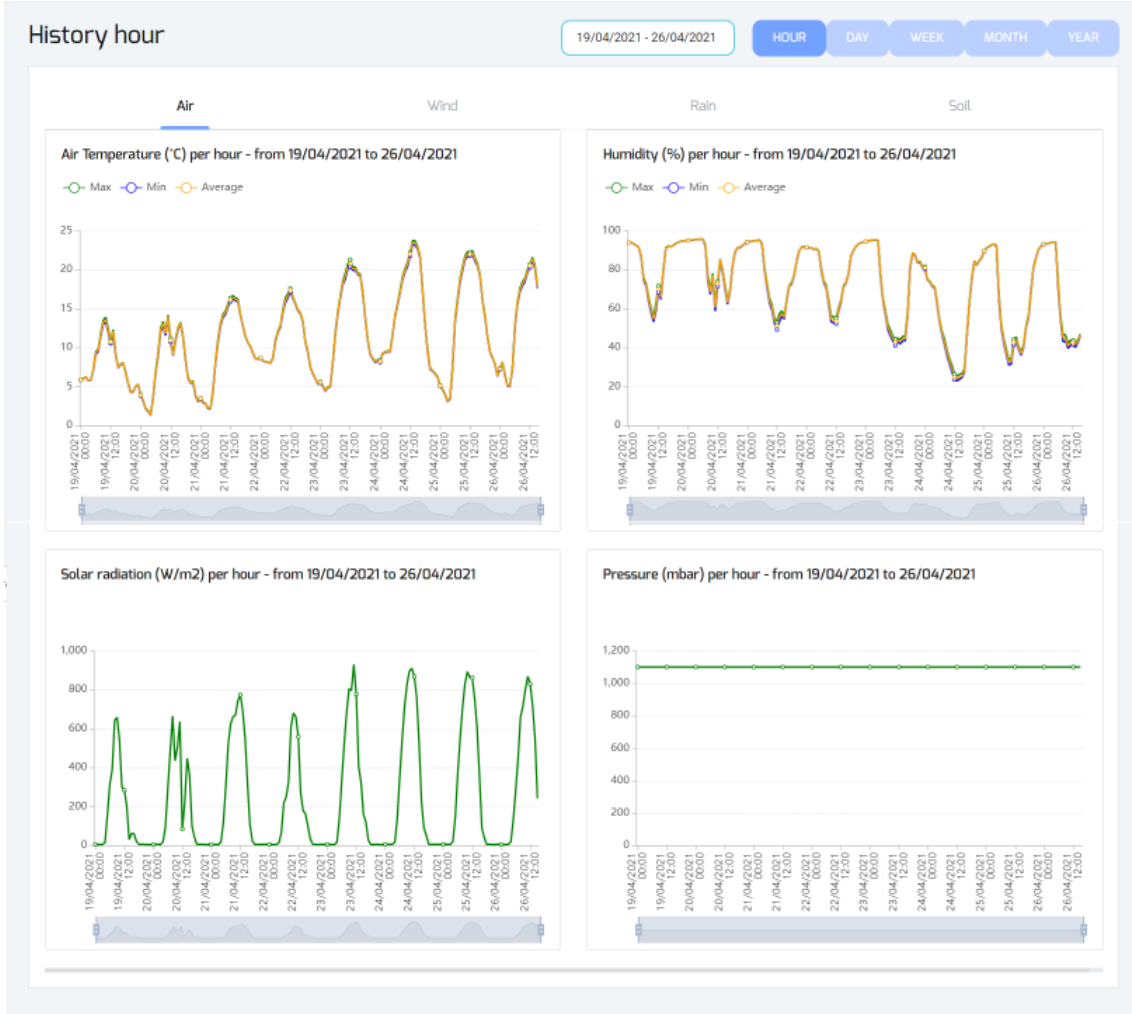


Figure 40. User interface, historical air data.



Figure 41. User interface, historical rain data

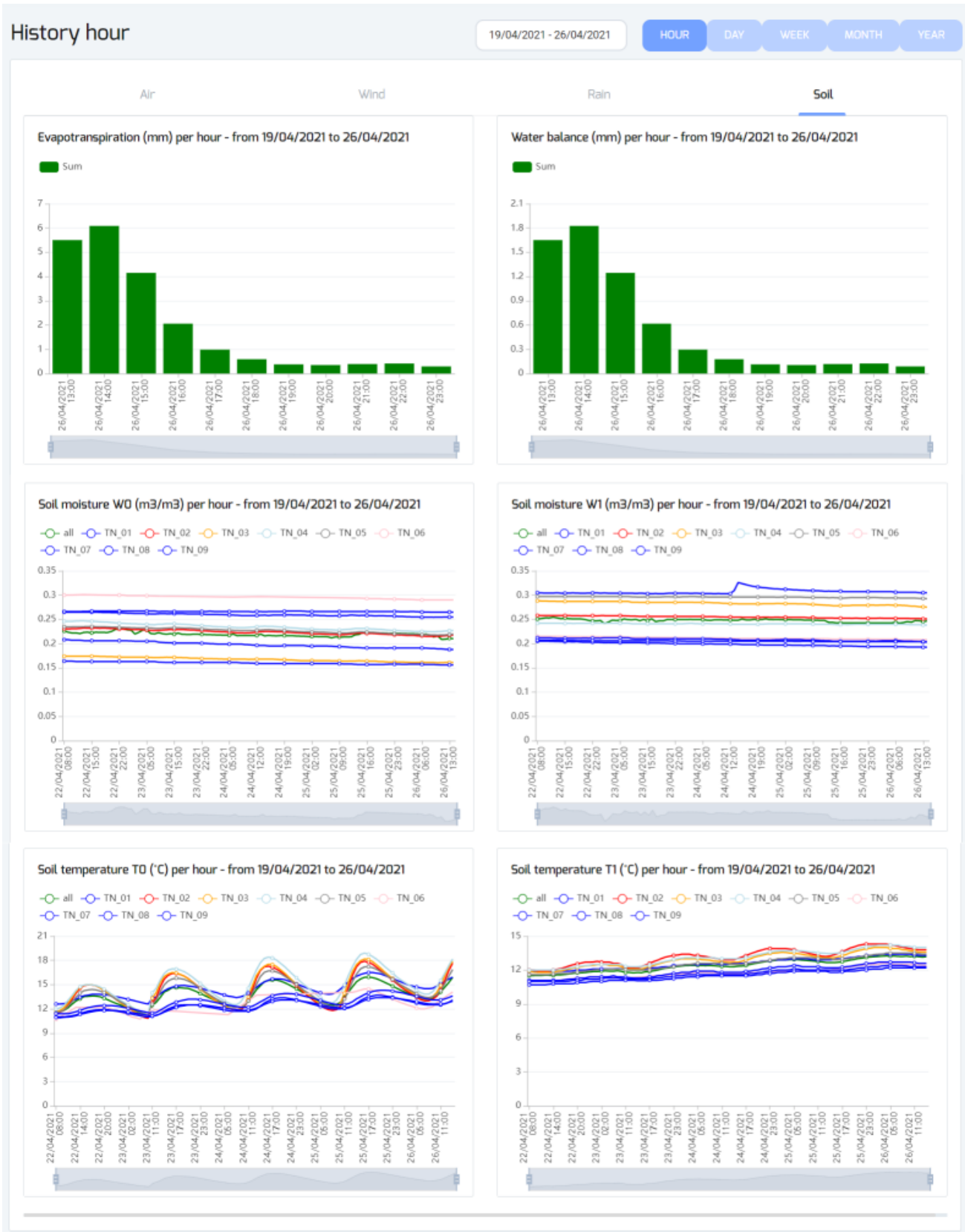


Figure 42. User interface, historical soil data.

Regarding the management of alarms, specific server-side flows have been created that analyze the data received from the IoT sensors and detect any alarm situations by activating notifications to be displayed in the user interface.

The following image in Figure 43 shows an example where the flow activates a pre-alarm if the air temperature drops below zero degrees.

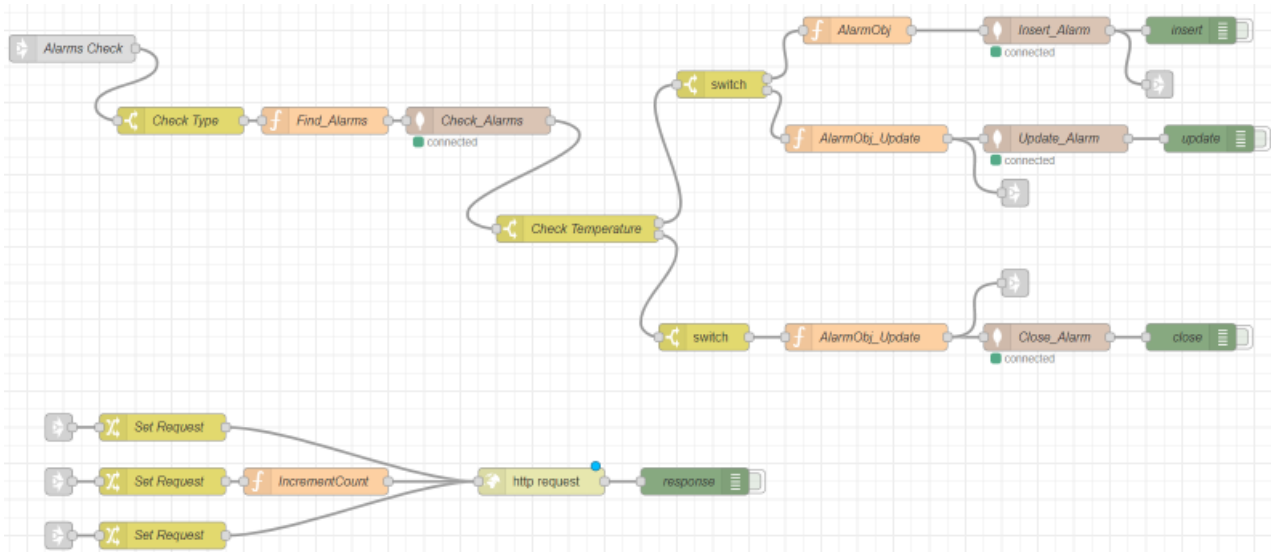


Figure 43. Sub-zero pre alarm flow.

The flow stores successfully the alarms in the specific database collection, as shown in the following Figure 44.

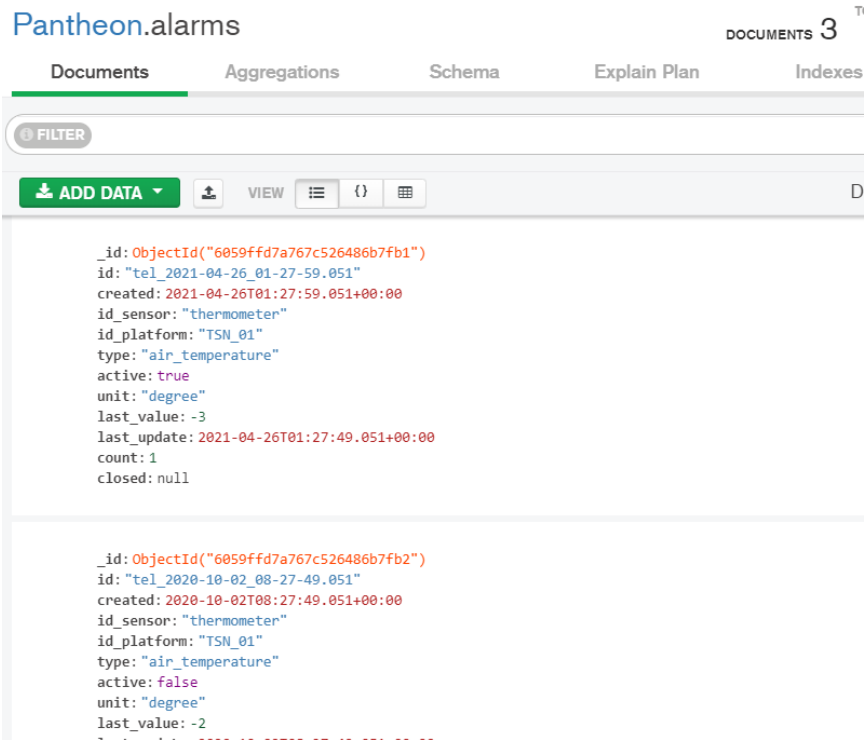


Figure 44. MongoDB "Alarms" collection.

Also on the user interface, there is a widget (Figure 45) that shows in real time the notifications of the alarms that are activated and that require the immediate attention of the user.

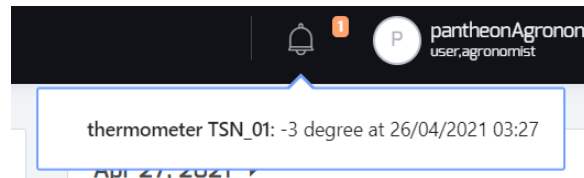


Figure 45. User interface, alarm notification.

In addition, there is an interface page dedicated to all the alarms that have occurred over time, in this table (Figure 46) all the details are shown and highlighted the current active.

| Created | Sensor | Platform | Type | Active | Closed | Value | Unit | Updated | Count |
|------------------|-------------|----------|-----------------|--------|------------------|-------|--------|------------------|-------|
| Created | Sensor | Platform | Type | Active | Closed | | | | |
| 26/04/2021 03:27 | thermometer | TSN_01 | air_temperature | true | | -3 | degree | 26/04/2021 03:27 | 1 |
| 02/10/2020 10:27 | thermometer | TSN_01 | air_temperature | false | 02/10/2020 11:27 | -2 | degree | 02/10/2020 10:27 | 1 |
| 02/10/2020 10:27 | thermometer | TSN_01 | air_temperature | false | | -1 | degree | 02/10/2020 10:27 | 1 |

Figure 46. User interface, alarms table.

4.2 Benchmark Objective Validation

This paragraph provides a summary of the benchmark objectives originally defined in the DoA [1] paragraph 1.1.5 and successively revised according to the Periodic Technical Report 2 (PTR-2).

Objective 1.1 (Ground Robot, Navigation, Sensing and Actuation) requires a Navigation System able to classify all the objects close to the ground robots either as dynamic obstacles or as trees, and a Task Manager capable of handling the execution and the completion of the required farming operations.

The former consists of four conceptual layers: control, planning, navigation, and localization. The control layer is achieved with a ROS package specifically designed for ground robot with Ackermann kinematic [15]. The planning layer is achieved with two ROS packages: the local planner package is responsible for the obstacle avoidance between waypoints and the communication with the controller [15], and the global planner package is responsible to define the set of waypoints and the routes to perform the farming operations [16]. The mapping and localization layers are achieved with SLAM system (Simultaneous Localization and Mapping), which can generate a map from scratch, or starting from a known map it can extend it over time. An example of the SLAM system is provided in Figure 47. The main features of this SLAM are the georeferencing of the position of ground robots and all elements present in the field and the classification of obstacles based on the location relative to the map.

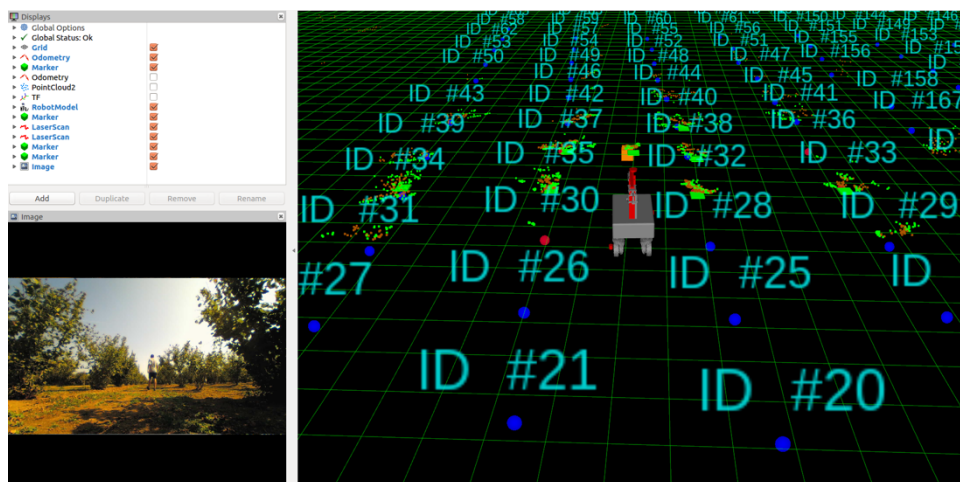


Figure 47. SLAM system recognizing a dynamic obstacle.

The latter includes the automatic system to schedule all the operations required, which are described below:

- The movements of the robot between different waypoints of the current route;
- The calls to the ROS services that enable the activation of the required activities.

Regarding the benchmarks requested in [1], different campaigns aimed to collect data have proven that the robot is able to autonomously navigate the orchard for more than 7 hours. The effectiveness of the herbicide

spraying has been validated in February 2020 (see Figure 48-left) when the benchmarking requirements have been successfully met. However, the spraying was not able to cover the sucker in its entirety without requiring a continuous movement of the robotic arms equipping the spray nozzle. For this reason, in May 2021 the spray nozzle has been changed (see Figure 48-right) to increase the area covered by the spray at the expense of a reduced distance from the plant. At the time of writing, the herbicide spraying accuracy of the new spray nozzle has yet to be validated on the plants. Validation campaigns aimed to prove its effectiveness are expected to be carried out in July 2021. Finally, as explained in the document PTR2, due to the problems that the paint spread by the paintball bullets may generate to the sensorial equipment of the robot, the mitigation action originally provided in the DoA (in the “Risk Analysis and Contingency Plan” section) has been adopted, i.e., “the marking of the branches has been included in the 3D model representation available on the user interface”.



Figure 48. Comparison between old (left) and new (right) spray nozzles.

Objective 1.2 (Aerial Robot Navigation, and Sensing) expects an automatic mapping with multispectral sensors of an orchard of 2ha in a mission using 2 sets of batteries with a resolution of 5 cm/pixel.

This objective is equivalent to the coverage of 1ha of the orchard in one single flight of the UAV. Based on the specifications required for a resolution of 5cm/pixel and the construction of a proper orthomosaic (at least 80% of overlap between pictures), this objective is reached by performing a flight at 50 meters of altitude and speed of 2m/s. As it can be seen in Figure 49, this configuration provides a flight of 16 minutes which is slightly lower than the flight autonomy of the drone with a new set of batteries. However, since the final demo period will be used as a key remote sensing campaign for the validation of several parameters

concerning water stress and diseases, during the season the flights will be performed at this altitude and speed, demonstrating that the resolution is achievable with these parameters, but focused on a more reduced area. This decision provides more robustness to the sensing campaign in case there are any technical problems during the flights and allows to cover several times during the day the trial areas of the PANTHEON orchard. This measure is also necessary given that there are other sensors installed in the UAV (Thermal and RGB sensors) and that the sets of batteries already present some degradation due to the 4 years of project. The fact that this benchmark had not been tested before the final demo is due to: i) the postponement of flights during 2020 due to the COVID-19 situation and ii) due to new flying regulations, flights at 50 meters of altitude have been prohibited at the PANTHEON orchard since July 2020. A formal request to fly at this altitude was sent by the PANTHEON consortium to the different authorities involved. This request was accepted at the end of April 2021 in the form of a conditional authorization for specific days subject to confirmation prior to each flight. In May 2021, a second request was sent to include additional dates and authorized hours was send and it has not been formally approved yet.

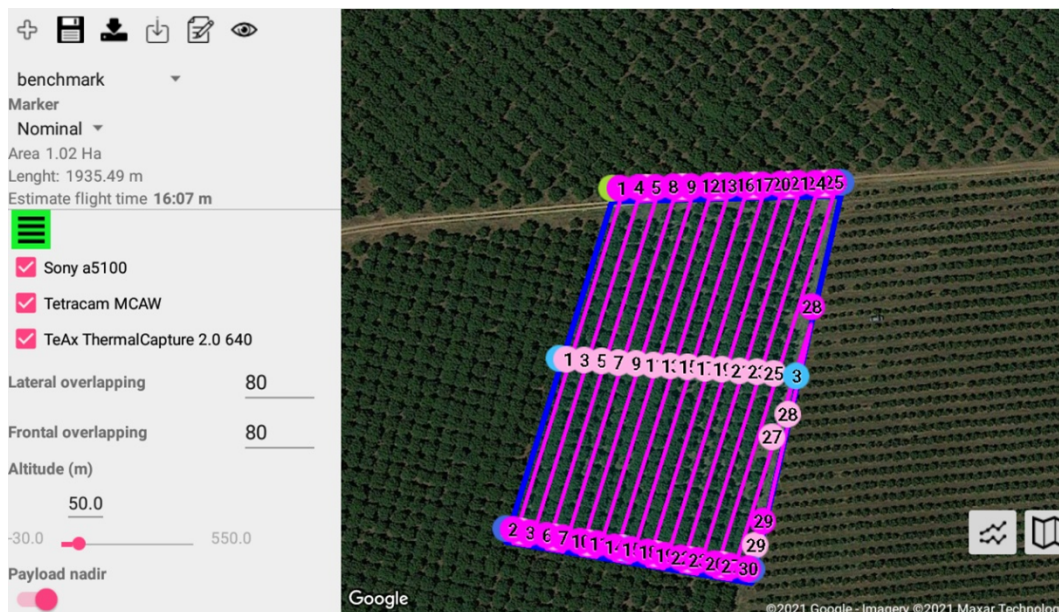


Figure 49. Planned flight for a coverage of 1 ha.

Objective 1.3 (Data Collection, Storage and Analysis) strongly depends upon the adopted architecture and the correct development of the integrated system. The data collected in the field is produced by IoT sensors, weather station, UAV and UGV. Each device is characterized by its own operating methods and peculiarities of data acquisition and transmission. The data is heterogeneous, and a specific acquisition flow has been prepared for each type. The MQTT, HTTP and ROS Topics protocols that are being used guarantee the correct

transmission of telemetry while respecting the required performances. The acquired data is uploaded to the central server and stored in the file manager and database. The storage was specifically designed to manage field data. The choice of MongoDB is ideal in IoT-type scenarios and when heterogeneous data needs to be processed. A No-SQL database is also ideal for data analysis operations, in fact the system is scalable, and it maintains adequate response times even with the exponential increase of data. The server-side programming and execution environment provides the appropriate tools to implement specific analysis algorithms, for example Apache Kafka (event streaming platform) or Spark (analytics engine for big data analysis). These open-source tools integrated into the system, which are among the most used in the industrial world, guarantee the processing of high-performance data pipelines, data stream analysis, integration, and implementation of mission-critical applications. The validation of Data Collection, Storage and Analysis was completed by testing the functionality of the data acquisition, transmission, data storage and analysis system. The correct display of the acquired and processed data on the user interface provides an additional level of validation of the developed system.

Objective 2.1 (Tree Geometry Reconstruction). The tree geometry reconstruction tool chain which performs the co-registration of the point clouds and graph-based tree modelling, as well as the branch and sucker classification algorithms is well developed and implemented in the system. Co-registration accuracy has been analysed in an experimental setup described in Deliverable D4.1 [8] and is around 3mm. The validation of the tree geometry reconstruction was performed on synthetic and real trees. The entire process is described in detail in Deliverable D4.2 [5].

Objective 2.2 (Water Stress). The validation of the radiometric and geometric pre-processing is still in progress. Due to hardware issues as well as mission planning problems as a consequence of a change of the legal status of our test site, the vegetation indices we computed were lacking in terms of consistency and geo-localisation. Additionally, there was no detectable water stress on the field due to an abundance of rainfall. Therefore, we are still working on the validation of objective 2.2.

Objective 2.3 (Pest and Disease Detection) expects to develop a detection system able to detect pest infestations and plant diseases through data collected from UAV and UGVs.

Concerning the pest infestation, a false positive rate below 10% and a false negative below 35% with respect to human inspection is expected. We focus on detecting the presence of some species of true bugs (*Insecta: Hemiptera: Heteroptera*) on sticky traps. Indeed, these insects are considered the key pests of hazelnut production, causing reduction of yield and nuts organoleptic quality worldwide.

This objective is reached by pursuing a data-driven approach and training a *You Only Look Once* (YOLO)-based Convolutional Neural Network (CNN) on a custom dataset¹ collected in a realistic outdoor environment. By exploiting several data augmentation techniques, the CNN reaches 94.5% average precision on a holdout dataset composed of 611 images. More specifically, a total of 5992 samples of true bugs are present in the images and 5813 of them are correctly detected, while 179 are false negatives, i.e., the false negative rate is 3%. Moreover, 540 false positives are obtained. Therefore, the pest detection system fulfils the defined benchmark regarding the false negative rate with respect to an ideal human operator achieving perfect detection precision equal to 100%. Note that the false positive rate is not meaningful in a detection system as it is defined as the number of false positives divided by the total number of false positives and true negatives, which are not defined in a detection system. However, the number of recorded false positives (540) is comparatively low with respect to the total number of true bugs (5992).

The detection system is deployed on a NVIDIA Jetson Xavier, which can be easily integrated on the ground robot, and on which the detection reaches 50 fps, enabling online processing. An example of predicted true bugs by the developed system is shown in Figure 50, where detected true bugs are highlighted with bounding boxes. Prediction confidence scores are also reported, and false positives are marked with red x-marks, while false negatives with red circles. The figure shows that the detection system can properly recognize almost all bugs on the traps, even if other irrelevant insects and objects are present. The only false positives are obtained with a few insects that are visually very similar, while false negatives rarely occur (only one false negative is shown).

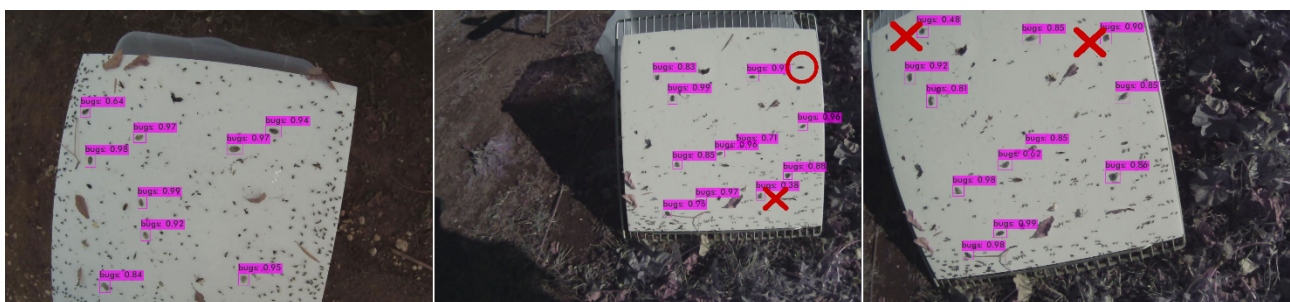


Figure 50. Examples of detected true bugs. Confidence scores are reported. False positives are highlighted with red x-marks, while false negatives with red circle.

In addition, the application of UAV-based sensors to detect the presence of hazelnut Anthracnose (*Gloeosporiumcoryli*) using multispectral visible and near infrared (VNIR) and thermal data collected from the UAV is in progress. The multispectral and thermal data will be analyzed and compared to the ground measurement for the necrotic leaf area observed in the field.

¹ Made public and available at the link <https://tinyurl.com/y4rv9wo5>

This season will be used for the validation of the parameters concerning the disease present in the orchard. This was not done in the previous season due to inconsistencies of the data acquisition caused by administrative restrictions and COVID-19.

Objective 2.4 (Fruit Detection). The automated counting of the number of nuts/clusters of nuts based on a deep learning point cloud segmentation is still in the prototype phase.

Objective 3.1 (Pruning Plant Policies) requires the development of two automated systems: one in charge of managing the presence of suckers through the application of herbicide on the plants and the other in charge of computing possible pruning decisions based on agronomical criteria.

The automatic system in charge of managing the presence of suckers in the orchard is equipped on the Sherpa R-B. The functionalities of the suckers' management system are described as follows. While the robot navigates the orchard, a YOLO-based deep neural network detects the presence of the suckers by processing the data collected by an RGBD camera mounted on the robot. Figure 51 shows the detection of the suckers.



Figure 51. Detection of a sucker with confidence level reported.

The RGBD data and the information on the detection of the suckers are then processed by two open-source libraries (ORB_SLAM2 and Kimera-Semantics) to virtually reconstruct a 3D mesh of the detected suckers, from which the surface area of the suckers is estimated, and the amount of herbicide is computed. Figure 52 depicts the 3D mesh reconstruction of a sucker with its estimated surface area of 0.8m².

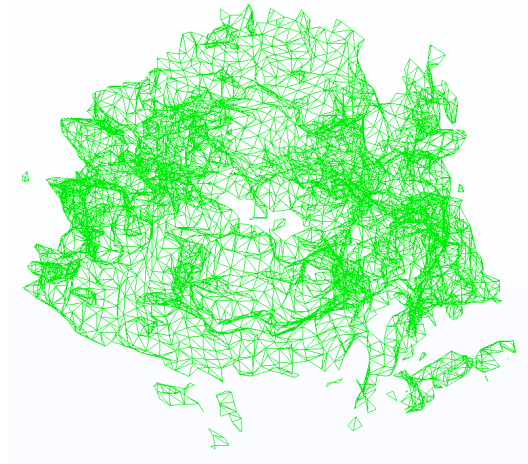


Figure 52. Sucker 3D mesh reconstruction. Surface area is equal to 0.8m².

The amount of herbicide computed by the suckers' management system has been successfully validated by an agronomical expert. Experimental validations of the automatic suckers' treatment by the robot in full autonomous mode are expected to be carried out in July/August 2021.

The automatic pruning protocol provides to the user pruning suggestions for branches that do not fit agronomical criteria such as shape, orientation, or length, in a graphical form as the ones depicted in Figure 53. The pruning protocol utilizes as input a 3D graph model produced by the Tree Geometry Reconstruction (Objective 2.1) and returns as output the IDs of the edges (branches) that should be removed according to a set of agronomical criteria (illustrated in Deliverable D5.3 [6]) as well as an estimation of the quantity of the wood that would be removed from the tree.

Validation of the pruning protocol has been conducted in early March 2021 on a set of 9 young trees, 3 for each tree type available, i.e., single trunk, symmetric bush, and free bush. For the mono-stem type a total of 39 cuts were suggested by the pruning protocol. All 39 cuts were approved and executed by the agronomical expert (100% approval rate). For the symmetric type a total of 93 cuts were suggested by the pruning protocol. 81 out of 93 cuts were approved and executed by the agronomical expert (87% approval rate). Finally, for the bush type tree a total of 113 cuts were suggested by the pruning protocol, 104 of which were approved and executed by the agronomical expert (92% approval rate). Overall, 224 cuts out of 245 suggested cuts were approved and executed by the agronomical expert, reporting an overall 91% approval rate, far above the required 70% approval rate benchmark.

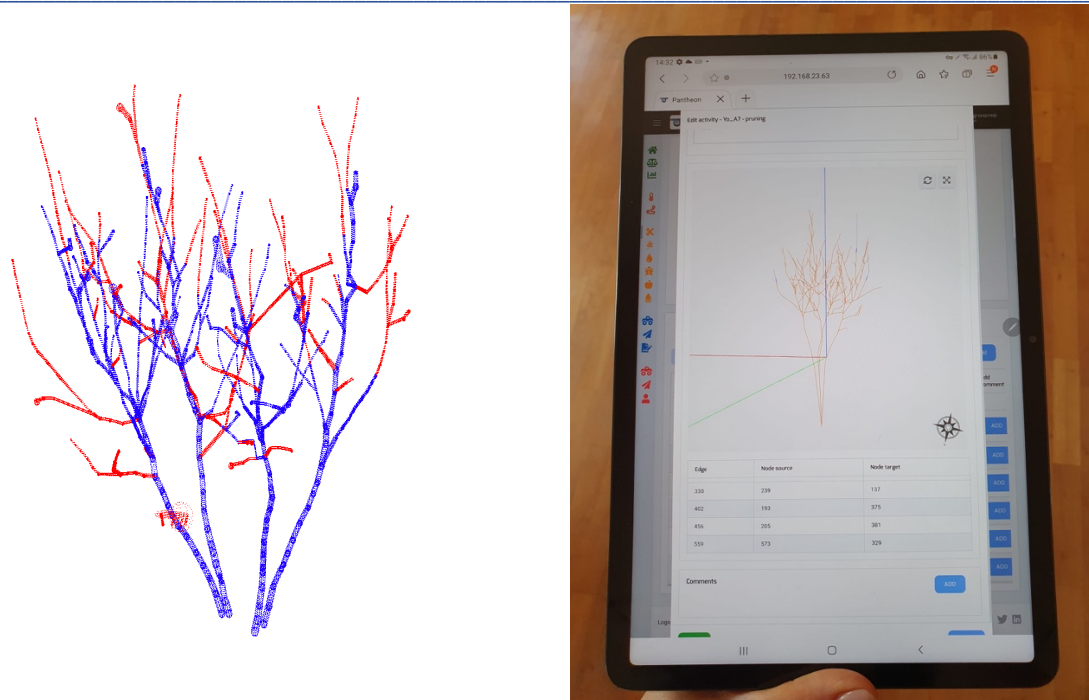


Figure 53. Graphical representation for a symmetric type of tree for the automatic pruning system as outcome of the pruning algorithm (left) and as shown in the user interface deployed on a smart device (right).

Objective 3.2 (Plant Phytosanitary Status).

- SCADA Irrigation Control System:** The main principles of the SCADA irrigation control system and the theoretical development are already defined in Deliverable D5.1 [3]. Briefly, the system uses meteorological measurements from the weather station to anticipate possible effects on the water status of the plants. This is combined with a regulation action based on the obtained indices from remote sensing. The field validation of the irrigation control system is still ongoing as some of the classical indices that are used in agriculture have proven to be unreliable for hazelnuts so far. To mitigate this issue, based on preliminary results obtained on summer 2020, the Consortium has proposed a new approach which attempts to use sap flow measurements from some selected trees as a more accurate index of the water status of the plant. Indeed, this has been possible thanks to the synergy established with another project carried out by an external research team from Padova University, which resulted in the integration within the agrometeorological network of sap-flow sensors and dendrometers both in irrigated and not irrigated plants.
- Integrated Pest Management:** Regarding the Integrated Pest Management, as reported in Objective 2.3, the YOLO-based Convolutional Neural Network (CNN) had the capability to identify the presence of some species of true bugs, in particular "*Palomena prasina*" on pheromone sticky traps with accuracy of 94.5%. To define a proper Integrated Pest Management based on this automatized bug

detection system, these traps should be deployed not inside the field but around of it, to intercept the true bugs that usually overwinter outside the orchard, thus before they go into the hazelnut plantation causing first the traumatic abortion and then subsequently the “*Cimiciato*”. Specifically, traps must be deployed in the field from the beginning of May. Notably, the detection of the true bugs with the proposed YOLO system on sticky traps works as a trigger that gives the start of i) the monitoring activity on the number of bugs into the field using the frapping method and ii) the monitoring of nut phenological stage according to what has been observed and reported in Deliverable D5.4. It should be highlighted that these two activities are very time-consuming and thus should only be carried out upon the detection of the approach of the bugs to the plantation with the proposed YOLO detection system. In addition, by resorting to the same YOLO framework, we have started another experimentation focused on the detection of hazelnut buds altered by the mite, *Phytoptus avellanae* into galls. This pest is reported causing swelling buds on generative and vegetative buds sometimes exceeding 20% of infestation in some susceptible varieties thus causing noticeable yield losses. The experimentation also focused on the best time to collect this information, i.e., either winter or early spring. According to our results, spring is ideal as the emission of leaves can facilitate the discrimination between galls and healthy buds. The results of this trial, although still preliminary, show promising results that need to be further investigated. If the good result will be confirmed, the expert system will be provided with a method able to contrast the damage activity of this pest. Indeed, should the YOLO system detect that the 20% threshold has been exceeded, a treatment with sulphur could be applied when the average temperature reaches the 12/13° C and when the new shoots have 3-4 fully developed leaves. Our trials also showed that the presence of mite's galls is concentrated in an area between 150 and 300 cm above the ground, hence our monitoring and control activities will be concentrated in this area.

Objective 3.3 (Fruit Production Estimation): A mathematical modelling for estimating the fruit production is currently being developed. Briefly, a model consisting of an integrator for each tree subject to monolateral (negative) disturbance and measurement noise was selected as starting point. Intuitively, this modelling is based on the observation that the number of hazelnuts on a tree can only decrease over time. Furthermore, it is assumed that the disturbance applied on each tree is strongly correlated with its neighbours and depends nonlinearly on the state (i.e., a percentage of the nuts that are lost, and not an absolute value). For the estimation the idea is to use an Extended Kalman Filter with intermittent observations which, thanks to this strong correlation of the disturbance, would allow to infer the number of lost hazelnuts also for the non-measured trees. The relevance of the methodology is now being tested in simulation by using synthetic data.

From a first analysis it appears clear that, to limit the number of needed measurements, it is important that a strong assumption is made in the modelling regarding the correlation of the disturbance between different trees in the various areas.

Currently, the quantity of data collected on the orchard in Ronciglione does not suffice to tune and validate the model. This can be explained by the lack of data collected in the 2020 summer campaign due to the ongoing Covid-19 pandemic and the exceptionally anomalous expected yield of summer 2021 due to adverse weather conditions experienced at the beginning of 2021. To cope with this problem, the partner FERRERO made available to the consortium data collected over the years in various Turkey's provinces. While analyzing the provided data, it emerged that the measurement noise associated with the counting procedure used in Turkey is very high, at the point to mask completely the hazelnut loss disturbance, making thus the data unusable (in several seasonal data collection it appeared that the number of hazelnuts was increasing overtime instead of decreasing). On the contrary, the data collected in Ronciglione by the Consortium partner UNITUS, although it is not sufficient for tuning a model being the dataset not large enough, has a much lower uncertainty but it required a huge amount of work and time to be performed.

5 Traceability Matrix

The traceability matrix (Table 5) provides the correspondence between the initial project requirements [2] and the SCADA integrated subsystems.

| REQUIREMENTS | | SUBSYSTEMS |
|---------------------------------|------------------------------|---|
| Orchard Management | Irrigation | UGV Campaign IoT Remote Sensing Suggested Activities |
| | Pruning | UGV Campaign Remote Sensing Suggested Activities |
| | Sucker's detection / removal | UGV Campaign Remote Sensing Suggested Activities |
| | Pest and disease detection | UAV Campaign Remote Sensing Suggested Activities |
| Functional Specification | Wireless Network Backbone | IoT UGV Campaign Real-time Alarms |
| | GPS-RTK Positioning System | UGV Campaign UAV Campaign Map Layers |
| | RB-SHERPA Robots (UGV) | UGV Campaign |
| | UAV | UAV Campaign |
| | IoT Network | IoT Real-time Alarms |
| | DB Architecture | Historical Data Crop Yield Forecasting |
| | Central Unit | Remote Sensing Historical Data Suggested Activities Alarms |

Table 5. Requirement's traceability matrix.

6 References

- [1] L. Giustarini, T. Udelhoven, E. Garone, V. Cristofori, C. Carletti and A. Gasparri, "PANTHEON, H2020 Proposal numer: 774571. Sustainable Food Security - Resilient and resource-efficient value chains," 2017.
- [2] L. Giustarini, "H2020 PANTHEON, Deliverable D2.1, Requirement. Specification and Benchmark".
- [3] N. B. Rosselló, A. Gasparri and E. Garone, "H2020 PANTHEON, Deliverable D5.1, Water Management Control".
- [4] S. Ahlswede, S. Lamprecht and R. Retzlaff, "H2020 PANTHEON, Deliverable D4.5, Pest and disease detection".
- [5] S. Lamprecht and S. Ahlswede, "H2020 PANTHEON, Deliverable D4.2, 3D Tree Models".
- [6] M. Santilli, V. Cristofori, C. Potenza, R. F. Carpio, C. Silvestri and M. Paolucci, "H2020 PANTHEON, Deliverable D5.3, Pruning Management Protocol".
- [7] V. Cristofori, L. Giustarini, M. Paolucci, E. Garone, S. Lamprecht and A. Gasparri, "H2020 PANTHEON, Deliverable D5.2, Sucker's Management Control".
- [8] S. Lamprecht, "H2020 PANTHEON, Deliverable D4.1, Multispectral LiDAR Pint Clouds".
- [9] R. Carpio, J. Maiolini and A. Gasparri, "H2020 PANTHEON, Deliverable D3.3, Communication Infrastructure".
- [10] E. Graziani, "H2020 PANTHEON, Deliverable D2.2, Guidelines for Components and Documentation compatibility".
- [11] E. Graziani, S. Samà and R. F. Carpio, "H2020 PANTHEON, Deliverable D3.2, Data Management".
- [12] E. Graziani, "H2020 PANTHEON, Deliverable D3.4, User interface".
- [13] N. B. Rosselló, A. Gasparri, E. Garone and R. Carpio, "H2020 PANTHEON, Deliverable D3.1, Robotic Prototypes".
- [14] C. Potenza and N. B. Rossello, "H2020 PANTHEON, Deliverable D6.1, Robotic Vehicles Field Validation".
- [15] R. F. Carpio, C. Potena, J. Maiolini, G. Ulivi, N. B. Rossello, E. Garone and A. Gasparri, "A Navigation Architecture for Ackermann Vehicles in Precision Farming," *IEEE Robotics and Automation Letters (RA-L)*, 2020.
- [16] R. F. Carpio, J. Maiolini, C. Potena, E. Garone, G. Ulivi and A. Gasparri, "MP-STSP: A Multi-Platform Steiner Traveling Salesman Problem Formulation for Precision Agriculture in Orchards," *IEEE International Conference on Robotics and Automation*, 2021.