



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

1

Type of document 3.4 – V1.0

User Interface

Dissemination level	PUBLIC
Submission Date	2019-10-31
Work Package	WP3
Task	T3:5
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 to describe the user interface of the PANTHEON project.

The document focuses on end user application definition and development, to create a multi-user and multi-device application, suitable for smart farming scenarios, with a specific focus on the management of hazelnut fields.

The idea is to provide PANTHEON users with a command and control console for activities management, such as pruning, water management, etc., which represent the core activities of this research project.

Briefly, the main contributions of this deliverable are listed below:

- Description of application Use Cases
- Software requirements specification
- Architecture definition
- Application design including mock-up definition and user interface description

Table of Content

1	System Overview.....	9
1.1	System Software Architecture.....	10
1.2	User Interface.....	13
1.2.1	Goal of the User Interface	13
1.2.2	User Interface Responsibilities	13
1.3	Document Overview.....	14
2	Use Cases	16
2.1	Actors	17
2.2	Use Case Schema “UC_DCP_Farm”	18
2.3	Use Case Schema “UC_DCP_Acquisition”	20
2.4	Use Case Schema “UC_DCP_Activity”	21
2.5	Use Case Schema “UC_DCP_Planning”	23
2.6	Use Case Schema “UC_DCP_Monitoring”	24
2.7	Use Case Schema “UC_DCP_CasualUsers”	25
2.8	Use Case Schema “UC_DCP_FieldWorkers”	26
2.9	Use Case Schema “UC_DSP_DataAnalysis”	28
2.10	Use Case Schema “UC_DSP_Infrastructure”	29
3	Functional Requirements	30
3.1	General.....	30
3.2	Farm	31
3.3	Tree	32
3.4	Harvest	33
3.5	History	33
3.6	Data Acquisition	34
3.7	Agronomic activity	35
3.8	Pruning.....	35
3.9	Sucker Control.....	36
3.10	Water Management.....	37
3.11	Pest and Disease.....	38

3.12	Missions.....	38
3.13	IoT Data	39
3.14	Monitoring	39
4	Interface Requirements	40
4.1	Front-end Back-end Interface	40
4.2	Back-end Database Interface	40
4.3	Back-end ROS Interface.....	40
5	Non-functional Requirements.....	41
5.1	Access Security	41
5.2	Availability	41
5.3	Confidentiality.....	41
5.4	Integrity	41
5.5	Usability.....	42
5.6	Flexibility	42
5.7	Maintainability	42
5.8	Modifiability	42
5.9	Verifiability	43
5.10	Installability	43
5.11	Interoperability	43
5.12	Portability	44
5.13	Reusability	44
5.14	Computer Resource.....	44
5.15	Design and implementation constrain	44
5.16	Precedence and Critical Requirements	44
6	Application Architecture	45
6.1	Overview	45
6.2	Required States and Modes	47
6.2.1	Modes.....	48
6.2.2	States.....	49
7	Application Design.....	51

7.1	ORM Data Model.....	51
7.1.1	Farm	51
7.1.2	Data Acquisition	53
7.1.3	Activity.....	54
7.1.4	Mission	55
7.1.5	Monitoring.....	56
7.1.6	Data Storage and Processing Centre (DSP).....	57
7.2	Interface Design Specifications	58
7.3	Prototype	59
7.3.1	User Interface Widget Mock-up	59
7.3.2	User Interface Page Mock-up.....	66
7.3.3	User Interface Dialog Mock-up.....	76
7.3.4	User Interface Look and Feel.....	79
7.3.5	Back-end Design	95
8	References.....	97

Table of Figures

Figure 1 - The global architecture of the software system.	10
Figure 2 - Data exchange between the DCP and the DSP components.	12
Figure 3 - Use case diagram, farm management	18
Figure 4 - Use case diagram, farm configuration	19
Figure 5 - Use case diagram, data acquisition management	20
Figure 6 - Use case diagram, agronomic activity management	21
Figure 7 - Use case diagram, mission management	23
Figure 8 - Use case diagram, real-time data monitoring	24
Figure 9 - Use case diagram, casual user farm data reading	25
Figure 10 - Use case diagram, field worker activity management	26
Figure 11 - Use case diagram, data analysis	28
Figure 12 - Use case diagram, infrastructure management	29
Figure 13 - Architecture of the end-user application	45
Figure 14 - Web App Modes Diagram	48
Figure 15 - Web App States Diagram	49
Figure 16 - Farm field model structure	51
Figure 17 - Farm elements hierarchical structure	52
Figure 18 - Data acquisition model structure	53
Figure 19 - Agronomic activity model structure	54
Figure 20 - Mission planning model structure	55
Figure 21 - Real-time data monitoring model structure	56
Figure 22 - Data Storage and Processing centre model structure	57
Figure 23 - Farm map widget mock-up	59
Figure 24 - Farm map widget, interactive mode	59
Figure 25 - Farm map widget, target selection in interactive mode	60
Figure 26 - Farm map widget, target selection in non-interactive mode	60
Figure 27 - Info box widget mock-up	60
Figure 28 - Date selector widget mock-up	61
Figure 29 - Task list widget mock-up	61
Figure 30 - Tree 3D viewer widget mock-up	62
Figure 31 - Tree pruning selector widget mock-up	63
Figure 32 - Mission summary widget mock-up	64
Figure 33 - Task summary widget mock-up	64
Figure 34 - Weather Forecast Summary widget mock-up	65
Figure 35 - Notification box widget mock-up	65
Figure 36 - Web App layout mock-up	66
Figure 37 - Dashboard page mock-up	67
Figure 38 - Farm page mock-up	68
Figure 39 - Farm activity list page mock-up	69
Figure 40 - Farm production page mock-up	70
Figure 41 - Mission page mock-up	71
Figure 42 - History page mock-up	71
Figure 43 - IoT Data page mock-up	72
Figure 44 - Monitoring page mock-up	73
Figure 45 - Task detail page mock-up	74
Figure 46 - Element detail page (sector or row)	75
Figure 47 - Element detail page (tree)	75
Figure 48 - Add Task dialog mock-up	76
Figure 49 - New Mission dialog mock-up	77

<i>Figure 50 - Start Mission dialog mock-up</i>	<i>78</i>
<i>Figure 51 - Stop Mission dialog mock-up</i>	<i>78</i>
<i>Figure 52 - Web App, desktop layout screenshot</i>	<i>79</i>
<i>Figure 53 - Web App, tablet layout screenshot</i>	<i>79</i>
<i>Figure 54 - Web App, smartphone layout screenshot</i>	<i>80</i>
<i>Figure 55 - Dashboard page screenshot</i>	<i>81</i>
<i>Figure 56 - Production page screenshot</i>	<i>82</i>
<i>Figure 57 - History page screenshot</i>	<i>83</i>
<i>Figure 58 - UAV/UGV Data Acquisition page screenshot</i>	<i>84</i>
<i>Figure 59 - Manual Acquisition page screenshot</i>	<i>85</i>
<i>Figure 60 - Pruning page screenshot</i>	<i>86</i>
<i>Figure 61 - Sucker Control page screenshot</i>	<i>87</i>
<i>Figure 62 - Water Management page screenshot</i>	<i>88</i>
<i>Figure 63 - Pest and Disease page screenshot</i>	<i>89</i>
<i>Figure 64 - Mission page screenshot</i>	<i>90</i>
<i>Figure 65 - IoT Sensor Data page screenshot</i>	<i>91</i>
<i>Figure 66 - Monitoring page screenshot</i>	<i>92</i>
<i>Figure 67 - Element (tree) detail page screenshot</i>	<i>93</i>
<i>Figure 68 - Task Detail page screenshot</i>	<i>94</i>
<i>Figure 69 - Back-end architecture design</i>	<i>95</i>

Abbreviations and Acronyms

DCP	Data Collection and Pre-processing
DSP	Data Storage and Processing
GPS	Global Positioning System
IoT	Internet of Things
IPM	Integrated Pest Managements
NAS	Network Attached Storage
ORM	Object Relational Model
REQ-FF	Functional Requirement
REQ-IF	Interface Requirement
REQ-NF	Non-Functional Requirement
ROS	Robot Operating System
RS	Requirement Specification Document
SCADA	Supervisory Control And Data Acquisition
SRS	Software Requirements Specification
UAV	Unmanned Aerial Vehicle
UC	Use Case
UGV	Unmanned Ground Vehicle
UML	Unified Modeling Language
UI	User interface
WP	Work Package

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.

The final result will be an improved average state of health of the orchard, and in an increased effectiveness of Integrated Pest Managements (IPM) activities. In conclusion, the proposed architecture has the potential to increase the production of the orchard while, at the same time, being more cost-effective and environmentally-friendly.

1.1 System Software Architecture

The user application fits into the bigger picture of the system software architecture [2]. This architecture is defined for the management of large amounts of heterogeneous data coming from hazelnut fields.

The system must be able to operate both in real-time, for plantation monitoring, and also in batch mode, to process large collections of historical data for both predictive analyses and support strategic decisions. We opted for open-source libraries and tools.

An example of architecture of data collection and processing system capable of meeting the above requirements is shown in Figure 1:

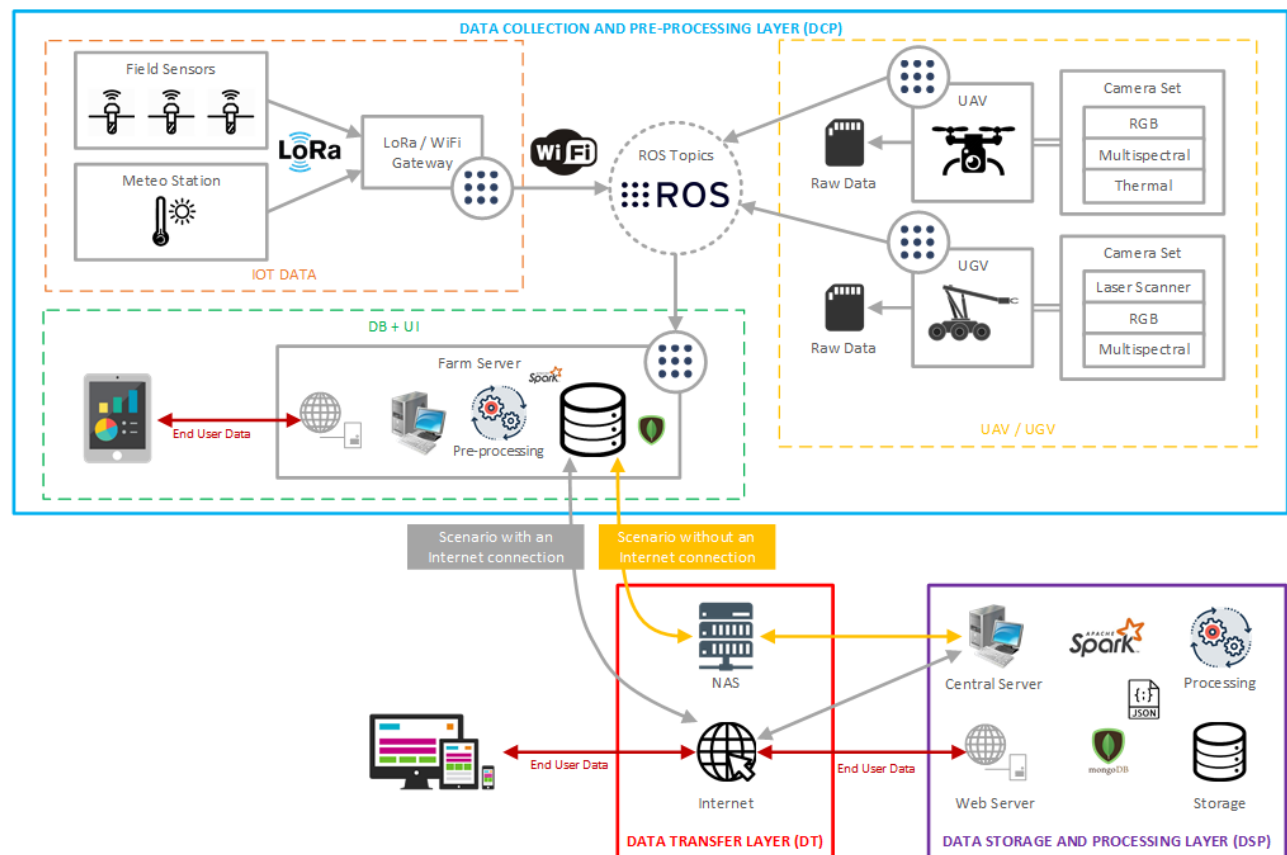


Figure 1 - Sketch of the global architecture for the software system.

The architecture is composed of three main components, which implement three operational levels:

- The “Data Collection and Pre-processing” layer (DCP layer in the following): this component is replicated for each hazelnut field and is dedicated to the collection of data coming from the various sources located in the field, i.e.: sensors, weather stations, ground robots (UGV) and drones (UAV).
- The “Data Transfer” layer (DT layer in the following): this is a middleware that deals with data transfer between the other two levels, bidirectionally, and between the global system and the final users;
- The “Data Storage and Processing” layer (DSP layer or centre in the following): this is a centralized unit in which all data coming from the various DCP components are stored and where analyses are carried out, mainly for knowledge extraction and decision support.

These three components will be described in more detail in the next paragraphs.

Data Collection and Pre-processing layer

Through a local communication network, the data coming from the collection nodes (sensors, weather stations, UGV and UAV) will be conveyed to the local farm server located in the warehouse near the hazelnut fields. The ROS protocol is used for data communication: it is able to manage data transfers with all the collection nodes mentioned above (including the IoT nodes via a gateway with the LoRa network) and is based on the publish/subscribe mechanism, which allows a decoupling between data collection and data processing. However, data can also be stored on the internal mass storage of the various devices and then transferred manually to the local farm server. This guarantee, on the one hand, the possibility of not losing acquired data even when a malfunction of the communication network occurs. On the other hand, it consents to avoid occupying a too wide communication band part, in case, for example, of large spectral images acquisitions from the UGVs.

The farm server acts as an entry point for collecting and managing all the data coming from one hazelnut field. It is configured as a ROS node to communicate with the various collection nodes and will store data using MongoDB, a NoSQL database system. All raw data acquired from the field will be stored on the database together with the result of data processing carried out locally or in the data storage level, as described below.

More specifically, some pre-processing activities will be carried out on this system to:

- perform operations of data cleaning and transformation, oriented for example to eliminate grossly incorrect data and to standardize formats;
- carrying out pre-aggregations to reduce the amount of data to be transmitted to the DSP layer and to make them more suitable for the subsequent analyses;
- monitor, through a local software application, activities on the collected data and provide information to the farmers on the real-time field status.

The local application will be Web-based, in order to be accessible using various devices. This application can be accessed directly by the operators in the field using the local farm server or using mobile devices, such as tablets and smartphones. An Internet connection is not required to access the application since it operates on the local database and so the network available on the field can be used for this purpose.

Data Transfer layer

Data exchange between the database, stored in the local farm server, and the central database, located in the DSP layer, will occur using an Internet connection when available. In case the area is not covered by an Internet connection, a portable device equipped with a large mass storage device, called NAS (Network-Attached Storage), will be used for data transfer. In this situation, the NAS device will be physically transported from the farm to the central database. Figure 2 shows the two communication scenarios: with and without the presence of an Internet connection.

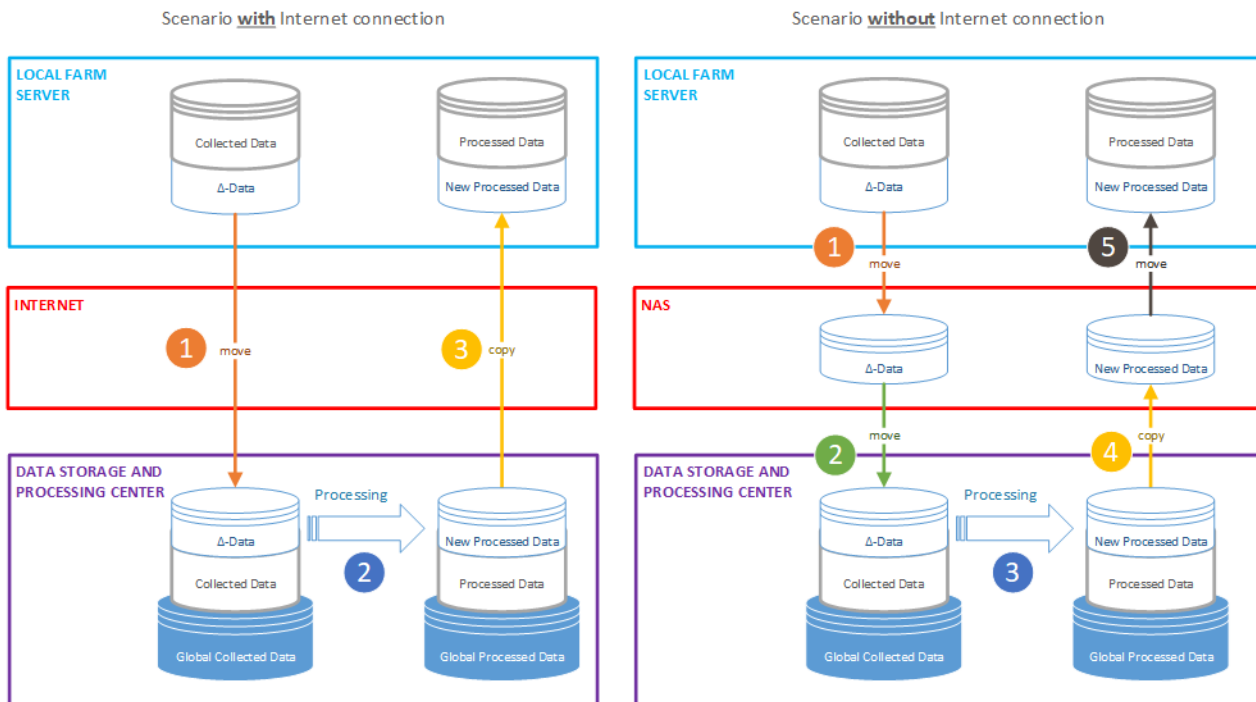


Figure 2 - Data exchange between the DCP and the DSP components.

In both cases, only the data collected from the last data transfer (usually called Δ -data) is actually copied. In the first scenario, Δ -data is directly transferred from the local to the central database and added to the “Global Collected Data” (1). The results of data analysis carried out in the DSP centre are stored in a special archive called “Global Processed Data” (2). The results obtained from Δ -data (called “New Processed Data” in Figure 2) are transferred back to the farm server (3), so that they can be exploited by users operating on the field, even when the DSP centre is not directly accessible or the communication is poor. In the second scenario, data transfer needs an intermediate step involving the storage and the transport of Δ -data in NAS devices.

Data Storage and Processing layer

The DSP centre is equipped with a computer infrastructure that is based on a cluster of computers whose nodes can be dynamically increased according to overall application storage and processing requirements. These requirements are driven by: the data volume to be stored, the data replication policies, the physical distance between the DSP centre and the hazelnut fields (e.g., located in different countries) that can be relieved by geographical clustering, and the need to support high data processing workloads.

All collected data will also be stored in a MongoDB database, in order to be easily exchanged with the databases stored in farm servers of the DCP layer. Data processing and analysis is activated at the DSP centre when new raw data arrives from the DCP layer. The data processing results are also stored in the database.

All of the above-mentioned choices follow the so called “data lake” approach, in which a large repository is used for storing any type of data, coming from different sources and possibly heterogeneous, for later use, usually aimed at knowledge extraction.

1.2 User Interface

1.2.1 Goal of the User Interface

The application should support the agronomists in charge of the orchard for monitoring, decision-making and agronomic interventions.

The User Interface will show a synoptic view of all the data collected by the remote sensors. In addition, the UI will provide the possibility to obtain more detailed information by clicking on the graphical representation of a specific object (e.g. a specific tree). The UI will display the farming operations, suggested by the decision-making system, once an object has been selected.

The possibility to select activities to be performed will be available for users. The UI will send such selection to the Farming Activity Planner and the Planner outcome will be shown on the screen. The possibility to start farming operations will then be enabled, and, if selected, real-time information coming from the present status of selected activities will be available and displayed.

Comment insertion and data storage in Data Repository will also be allowed; user will have the possibility to search and visualize the collected data for comparison and analysis too.

1.2.2 User Interface Responsibilities

The main responsibilities of the web application and the features it should provide to the users are summarized here:

- Farm management
- Planning of tree data collection missions
 - UAV
 - UGV
- Manual data acquisitions (observations)
- Agronomic operations management [3]
 - Pruning
 - Sucker control
 - Irrigation
 - Pest & Disease
- Monitoring of data collection mission
- Monitoring of the IoT data collection performed by sensor network
- Notifications and warnings of the decision support system
- Infrastructure monitoring & configuration
- Analysis of current data
- Analysis of archived data

These operations can be grouped into 2 categories: operations that can be performed directly on the hazelnut field and operations that can be performed connecting to the central server through internet.

Basically, the difference between the 2 use case categories is the app network connection and the data source the app works on. In the first case (on field use), internet connection is not required, the application will connect directly to the local server via Wi-Fi and the data source consist of information on the single hazelnut field. In the second case (global use), the application needs Internet access to connect to the central

server and the data source consist of, potentially, all the information on hazelnut fields covered by the PANTHEON system.

Indeed, the definition of roles (assigned to the users) will allow to manage the access to information and operations for each user. A farmer will only be able to manage their own hazelnut fields. An agronomist will have access to operations for all the farms they work on. An administrator can configure the infrastructure without visibility, for example, on agronomic operations, and so on.

1.3 Document Overview

This document describes the analysis of the requirements for the end-user application. According to this analysis, the application should provide:

1. a user interface to monitor the status of the hazelnut field
2. functionalities of a decision support system
3. analysis facilities

The goal of this document is to specify the application software requirements starting from the definition of the main architecture components. Moreover, the identified requirements will guide the development of the application in the next development steps: detailed design, implementation and testing of the software. This document covers the entire application design phase up to the definition of the UI mock-up.

In this document every requirement is identified by a unique code, for traceability reasons in software development [4]. The requirement coding will be composed according to the following formalism:

REQ-[TT]-[CC].[NN]

- **REQ** = Requirement
- **TT** = Type of requirement:
 - “FF” functional requirement
 - “IF” interface requirement
 - “NF” non-functional requirement
- **CC** = Category of the requirement. A two-digit sequence number starting from “01” for every type of requirement group. Used to group requirements based on common features.
- **NN** = Two-digits sequence number starting from “01” for every category group. Used to create a unique requirement code.

The document consists of following sections:

- This section (1) provides an overview of the system architecture, user interface and this document structure.
- The second section (2) provides the use case analysis.
- The third section (3) lists the application functional requirements.
- The fourth section (4) list the application interfaces requirements.
- The fifth section (5) lists the application non-functional requirements.
- The sixth section (6) provides an overview of the defined application architecture, including the identifications of states and modes.
- The seven section (7) describes the design of the application, including the definition of data model, interface specification and user interface mock-up.
- Finally, the last section (8) provides the document bibliography.

2 Use Cases

To determine the software requirements, a set of application use cases has been defined. They provide a good high-level analysis from outside of the system.

The use cases allow describing the basic system use scenarios through the actors interfacing with them. Actors are not necessarily human users but may be, often, other external information systems that interface with the one being developed. In this specific case, user interface development, the actors are essentially the human who will use it.

The identified use cases have a name and the actors able to initiate them. Moreover, they are included in a diagram that will assign a code for requirement traceability. Use cases are presented according to the standard UML diagram format. They appear grouped in separate functional areas to facilitate readability.

In each diagram, the relevant use cases are those with filled background; a description will be provided for them. The use cases with empty background are not relevant; they are "abstract", composition of included use-cases.

The use case coding will be composed according to the following formalism:

UC-[DCP|DSP]-[Cat]-[L1.L2.L3]

- **UC** = Use Case
- **DCP|DSP** = Data Collection and Pre-processing layer or Data Storage and Processing layer
- **Cat** = Free text representing the application category of use case.
- **L1.L2.L3** = Three level use case hierarchy. Each level is a 0-9 progressive number. L1 is the top level of the hierarchy. These three digits identify uniquely each use case inside the diagram.

2.1 Actors

Agronomist

The agronomist takes care of guiding the operations that determine the quality and quantity of agricultural production. So, in the PANTHEON scenario, they use the application to manage plants' pruning, sucker control, irrigation, and pest and disease control [3].

Farmer

This actor is the farm owner, in this case the owner of the hazelnut field where the PANTHEON system is installed. They are interested in monitoring all the activities carried out in the field, related to nut production.

Field Worker

Field workers are those who carry out field activities. The application provides a personalized calendar of interventions to be performed for each worker. They can also report each activity status of completion to the system via user interface; this will update the overall work status. Activities include both manual operations and robot activity supervising.

Casual User

The casual user is the unregistered system guest. Due to the lack of appropriate authorizations, this user cannot perform or access certain pieces of information concerning field management. They can only access the farm general information and see a basic overview.

Admin

The administrator's task is configuring system parameters for the hazelnuts field. They can access system logs and display the operating parameters for the installed devices and sensors. Admin also plays the superuser role in the farm management.

Infrastructure Manager

The infrastructure manager is in charge of monitoring the entire PANTHEON system, the DSP layer and the organization of the various DCP centres. Unlike the "Admin" user, who works at single field level (DCP), this actor uses the application to manage the configuration and control parameters of the system as a whole.

2.2 Use Case Schema “UC_DCP_Farm”

This section contains the use cases related to farm data management.

In detail, Figure 3 shows the diagram for the use cases related to the hazelnut field and corresponding harvest management.

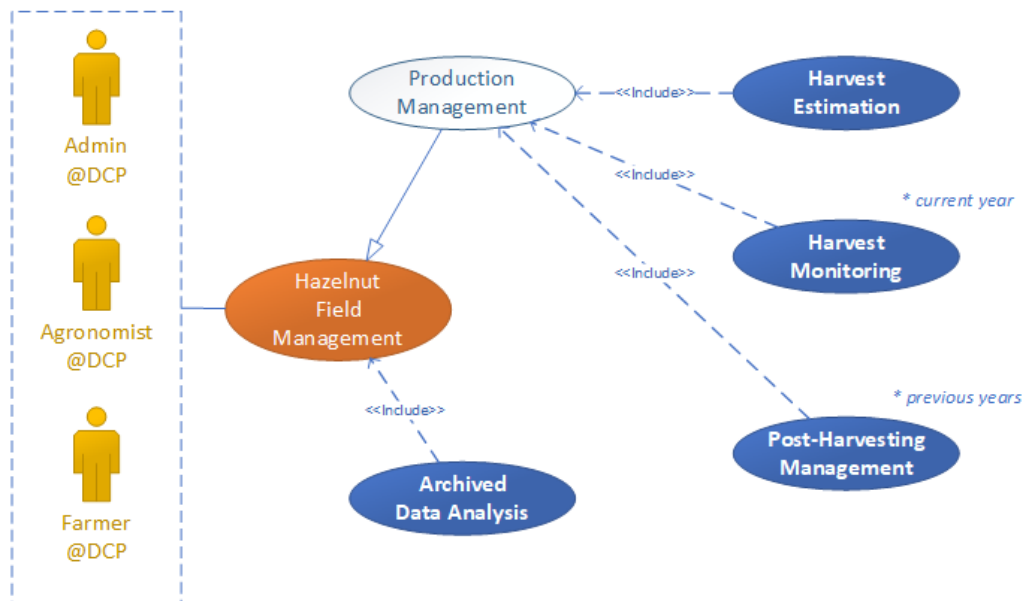


Figure 3 - Use case diagram, farm management

Here we present the description of the use cases illustrated in the diagram in Figure 3:

UC-DCP-Farm-1.0.0

Hazelnut Field Management: the user opens the application page showing the farm data in this use case (for example, the geographical location, the hectare extension, the farm owner, etc). This information can be edited by the administrator or by the farmer (farm owner).

UC-DCP-Farm-1.1.1

Harvest Monitoring: the user opens the farm production page in this use case. The whole current year harvest data are available here. Additional contextual information such as pending, in progress or completed status is shown for each harvest.

UC-DCP-Farm-1.1.2

Post-Harvesting Management: the user opens the page with the information on the entire harvest datasets, both for current and past years.

UC-DCP-Farm-1.1.3

Harvest Estimation: the user sees production estimation for the current year.

UC-DCP-Farm-1.2.0

Archived Data Analysis: user opens the page containing the farm archived data. For example, the timeline data for rainfalls, temperatures etc can be displayed. Filters can be applied to data visualization.

Figure 4 shows the use case diagram concerning the configuration of the hazelnut field.

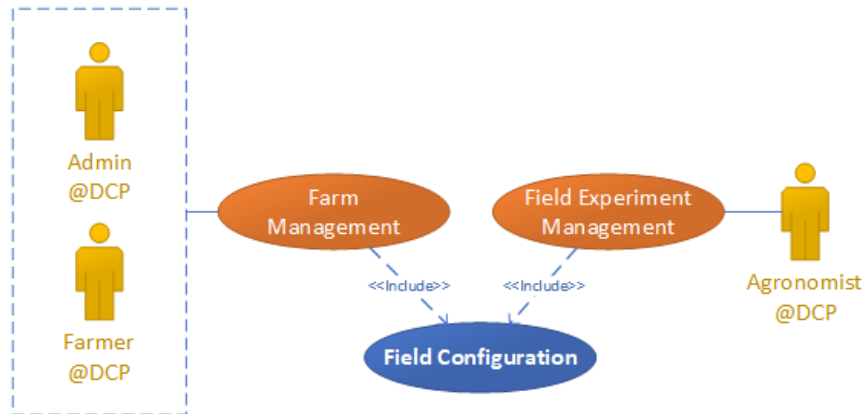


Figure 4 - Use case diagram, farm configuration

Here we present the description of the use cases illustrated in the diagram in Figure 4:

UC-DCP-Farm-2.0.0

Farm Management: administrator and farmer can set the field configuration. This procedure should be executed every time a new field is inserted into the PANTHEON system and the data can also be modified subsequently.

UC-DCP-Farm-3.0.0

Field Experiment Management: the agronomist configures an experimentation scenario on the hazelnut field. This allows separation between experimental and real hazelnut farm management scenarios.

UC-DCP-Farm-2.1.0

Field Configuration: this use case is included both in Farm Management and in Field Experiment Management, and allows the agronomist to configure and group hazelnut trees in a hierarchical grouping. Usually in real scenarios the hazelnut field is organized in sectors, rows and trees.

2.3 Use Case Schema “UC_DCP_Acquisition”

This section contains the use cases related to farm data acquisitions.

In particular, Figure 5 shows the diagram for the use cases related to the management of the activities for hazelnut field data acquisition.

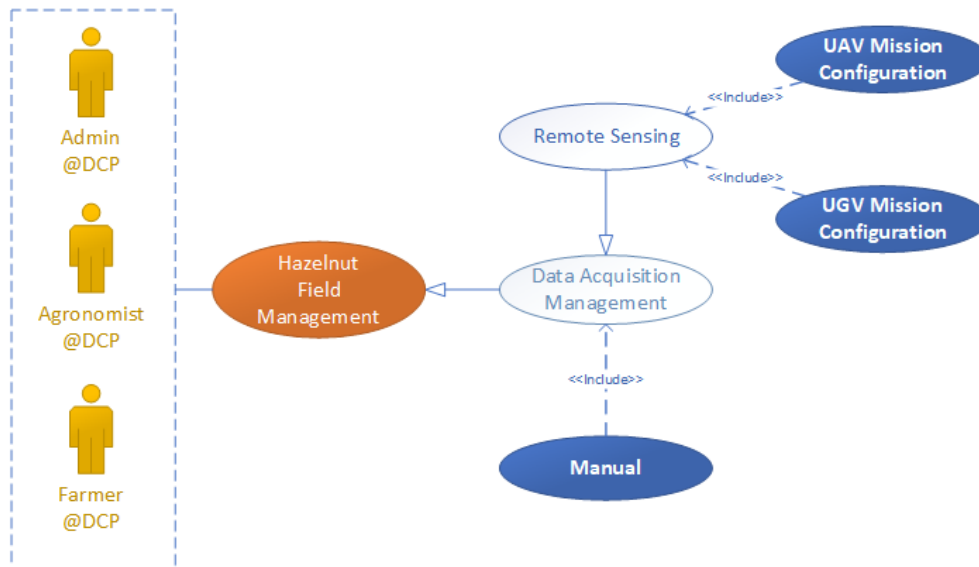


Figure 5 - Use case diagram, data acquisition management

Here we present the description of the use cases illustrated in the diagram in Figure 5:

UC-DCP-Acquisition-1.1.1

UAV Mission Configuration: the operator configures a data acquisition mission that should be performed by the UAV platform. The operator sets the type of intervention, date and time, and target (i.e. the scanning area). The target definition provides the trajectory waypoints to the platform.

UC-DCP-Acquisition-1.1.2

UGV Mission Configuration: the operator configures a data acquisition mission that should be performed by the UGV platform. The operator sets the type of intervention, date and time, and target (i.e. the scanning area). The target definition provides the trajectory waypoints to the platform.

UC-DCP-Acquisition-1.2.1

Manual: the operator inserts manual observations performed within the hazelnuts field into the system. Observations can be of various types (for example a tree should require sucker treatment or pruning) and the collected data are integrated with the automatic surveys carried out by sensors and robots. This use case is useful when the problem to be addressed needs a quick response, for example, if bugs are detected on a tree, the application must be performed before those bugs move to other areas of the field.

2.4 Use Case Schema “UC_DCP_Activity”

This section contains the use cases related to agronomic activity definition.

In particular, Figure 6 shows the diagram for the use cases related to the core agronomic activities management in the PANTHEON project.

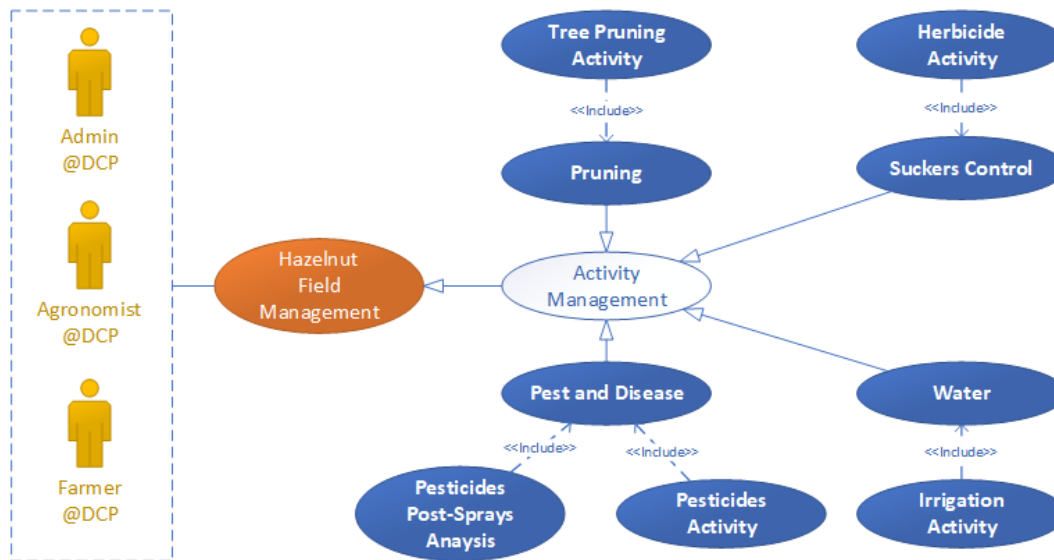


Figure 6 - Use case diagram, agronomic activity management

Here we present the description of the use cases illustrated in the diagram in Figure 6:

UC-DCP-Activity-1.1.0

Pruning: the operator opens the pruning activities management page, where they can analyse field pruning, supported by cartography map, review a summary of the pruning activities and visualize three-dimensional representations of the tree’s branches.

UC-DCP-Activity-1.1.1

Tree Pruning Activity: the operator defines the pruning operation for a tree, i.e. branches to be cut or marked. Furthermore, cut type and other pruning parameters can be defined here.

UC-DCP-Activity-1.2.0

Sucker Control: the operator opens the page to manage sucker control activities; the growth status of the field sucker, supported by cartography map, can be analysed in the page. A summary of the sucker’s treatment activities is also available.

UC-DCP-Activity-1.2.1

Herbicide Activity: the operator defines the sucker’s treatment activity for the trees under. The amount of herbicide for each tree can also be set.

UC-DCP-Activity-1.3.0

Water: the operator opens the water management activities page. The moisture soil status, supported by cartography map, can be analysed in the page. Irrigation activities summary are also available.

UC-DCP-Activity-1.3.1

Irrigation Activity: the operator defines tree or sector irrigation activities. Irrigation duration can be set. Weather station observations and weather forecasts support the operator in this process.

UC-DCP-Activity-1.4.0

Pest and Disease: the operator opens the Pest and Disease control. The plantation health status, supported by cartography map, can be analysed on the page. A summary of the different activities carried out using pesticides is also available.

UC-DCP-Activity-1.4.1

Pesticides Activity: the operator defines activities requiring pesticides use for affected trees or sectors. The quantities for each tree can be established. Weather station observation, in particular wind direction and force, are also available as complementary information.

UC-DCP-Activity-1.4.2

Pesticides Post-Sprays Analysis: the operator checks the trees health status after pesticide spray application. The types of used pesticides and the remaining quantities for current year in accordance with the laws in force can be checked in this page.

2.5 Use Case Schema “UC_DCP_Planning”

This section contains the use cases related to agronomic activities planning definition.

In particular, Figure 7 shows the diagram of the use cases related to the PANTHEON project mission management.

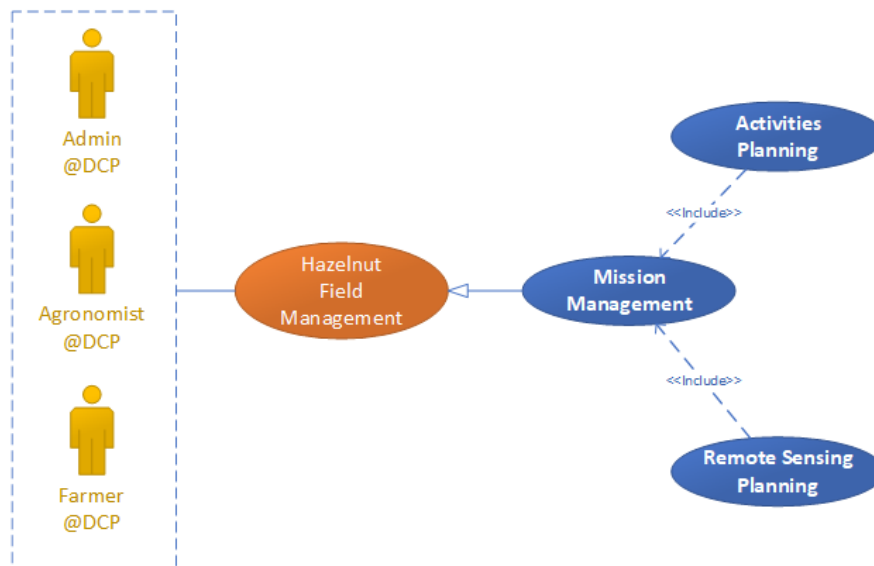


Figure 7 - Use case diagram, mission management

Here we present the description of the use cases illustrated in the diagram in Figure 7:

UC-DCP-Planning-1.1.0

Mission Management: the operator opens the mission page and visualizes all the applications defined in the calendar and the current status. The operator can filter the dataset to get an overview of the missions performed and those to be still carried out. For the missions already completed, execution details and results of the application can be analysed.

UC-DCP-Planning-1.1.1

Activities Planning: the operator plans the agronomic activities by creating a new mission. Missions should be created following a logic based on the application type and on who will perform the task, either appropriately configured UGV and field workers.

UC-DCP-Planning-1.1.2

Remote Sensing Planning: the operator plans the data acquisition missions carried out via UAV and UGV. The definition of the mission includes calendar planning and field sectors to be scanned.

2.6 Use Case Schema “UC_DCP_Monitoring”

This section contains the use cases related to autonomous systems real-time monitoring.

In particular, Figure 8 shows the diagram of the use cases related to the IoT data coming from the sensors installed on the field and the UAV/UGV missions monitoring.

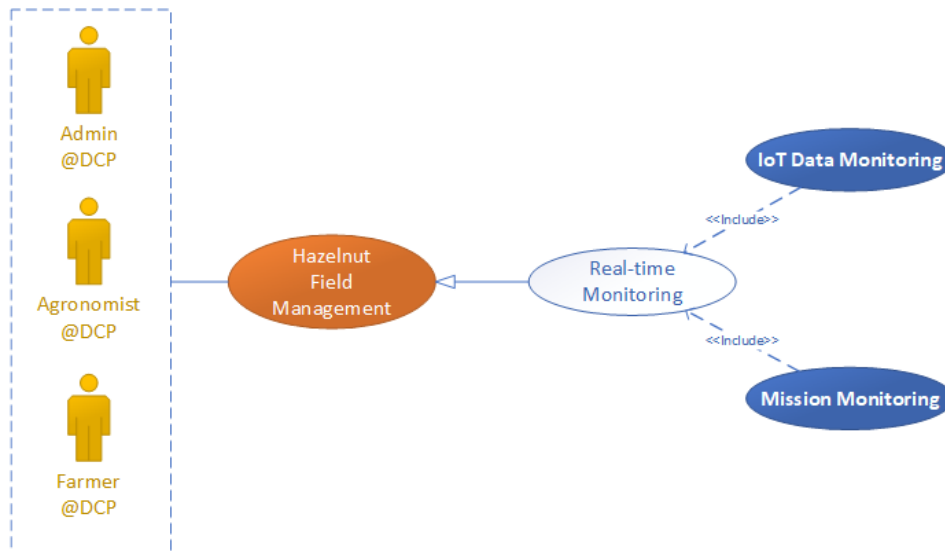


Figure 8 - Use case diagram, real-time data monitoring

Here we present the description of the use cases illustrated in the diagram in Figure 8:

UC-DCP-Monitoring-1.1.0

IoT Data Monitoring: the operator opens the field sensors data monitoring page. A weather station and a set of soil moisture sensors are installed in the field. Data are shown according to different representations, for example the last acquired datum can be represented either as text, as real-time graph where the value(s) dynamically update as soon as new values are collected and as cartography map where sensor locations are displayed.

UC-DCP-Monitoring-1.2.0

Mission Monitoring: the operator opens the UAV and UGV platforms monitoring page. The position of the platforms should be displayed in real time over the cartography map. Moreover, for each platform the current status of the installed equipment should be reported (for example camera battery level or laser scanner activation etc.). The ground robot monitoring is carried out for both data acquisition missions and activities execution missions (e.g. pruning or herbicide spraying), while the UAV is used only in acquisition tasks.

2.7 Use Case Schema “UC_DCP_CasualUsers”

This section contains the use cases related to casual user actor.

In particular, Figure 9 shows the diagram of the use cases that are a subset of the “UC_DCP_Farm”: the executor is the casual user.

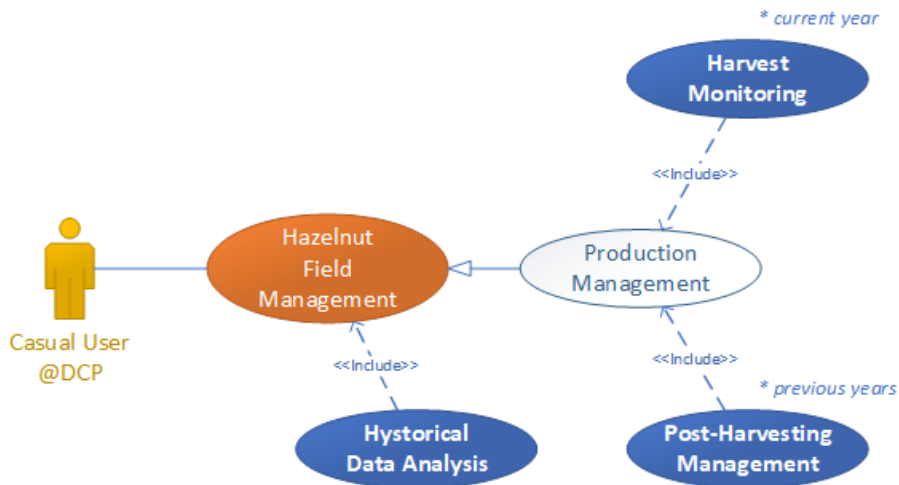


Figure 9 - Use case diagram, casual user farm data reading

Here we present the correspondence between current use cases and those in the "UC_DCP_Farm" diagrams:

UC-DCP-CasualUser-1.0.0

Hazelnut Field Management: corresponding to “UC-DCP-Farm-1.0.0”, but casual user cannot edit the data.

UC-DCP-CasualUser-1.1.1

Harvest Monitoring: corresponding to “UC-DCP-Farm-1.1.1”.

UC-DCP-CasualUser-1.1.2

Post-Harvesting Management: corresponding to “UC-DCP-Farm-1.1.2”.

2.8 Use Case Schema “UC_DCP_FieldWorkers”

This section contains the use cases related to the field worker actor.

In particular, Figure 10 shows the diagram of the use cases that are a subset of the “UC_DCP_Activity” and “UC_DCP_Planning”. The field worker use cases differ because they allow to read only some information and to confirm the completion of those activities the worker is in charge of. Please, note that the activities are both manual and automatic, performed by the worker (directly) and by a UGV, respectively.

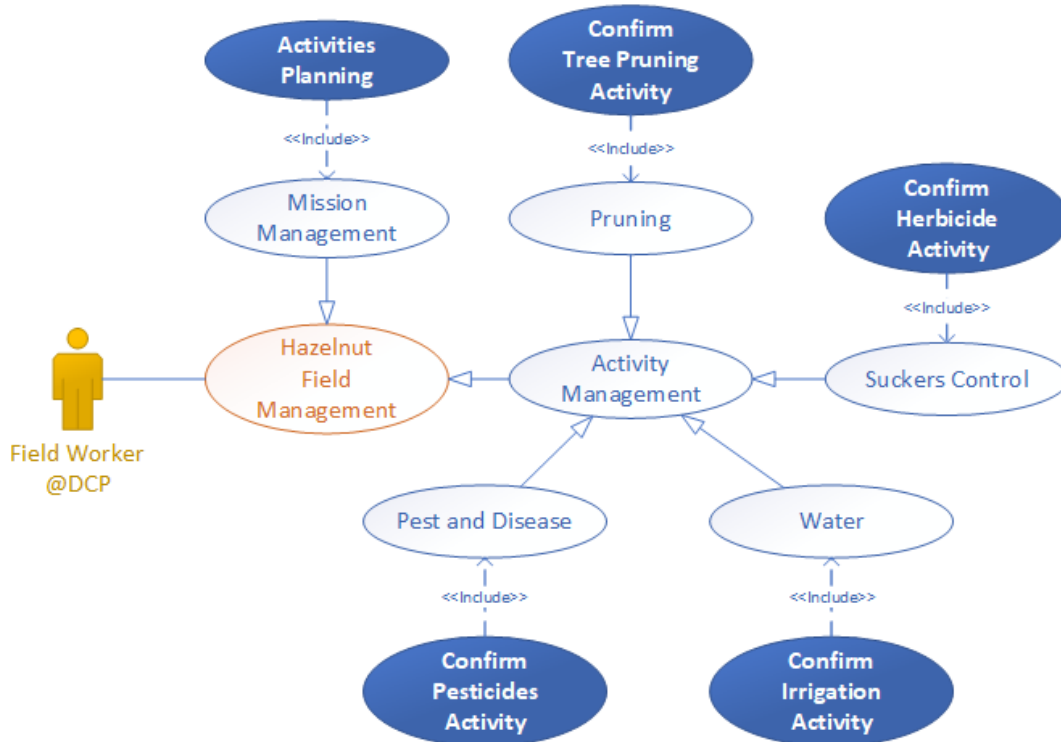


Figure 10 - Use case diagram, field worker activity management

Here we present the correspondence between current use cases and those in the “UC_DCP_Activity” and “UC_DCP_Planning” diagrams:

UC-DCP-FieldWorker-1.1.1

Activities Planning: similar to “UC-DCP-Planning-1.1.1”, but field worker cannot create or edit activities, they can only read their assigned activities. The worker activities are organized in missions.

UC-DCP- FieldWorker -2.1.1

Confirm Tree Pruning Activity: actions subset of “UC-DCP-Activity-1.1.1”, but the worker can only view their assigned activity and confirm the execution of the work.

UC-DCP- FieldWorker -2.2.1

Confirm Herbicide Activity: actions subset of “UC-DCP-Activity-1.2.1”, but the worker can only view their assigned activity and confirm the execution of the work.

UC-DCP- FieldWorker -2.3.1

Confirm Irrigation Activity: actions subset of “UC-DCP-Activity-1.3.1”, but the worker can only view their assigned activity and confirm the execution of the work.



UC-DCP- FieldWorker -2.4.1

Confirm Pesticides Activity: actions subset of “UC-DCP-Activity-1.4.1”, but the worker can only view their assigned activity and confirm the execution of the work.

2.9 Use Case Schema “UC_DSP_DataAnalysis”

This section contains the use cases related to the data analysis activity performed at DSP level. The application should be in “Global Mode”.

In particular, Figure 11 shows the diagram of the use cases related to the management of all data collected from PANTHEON fields and from the data analysis activities.

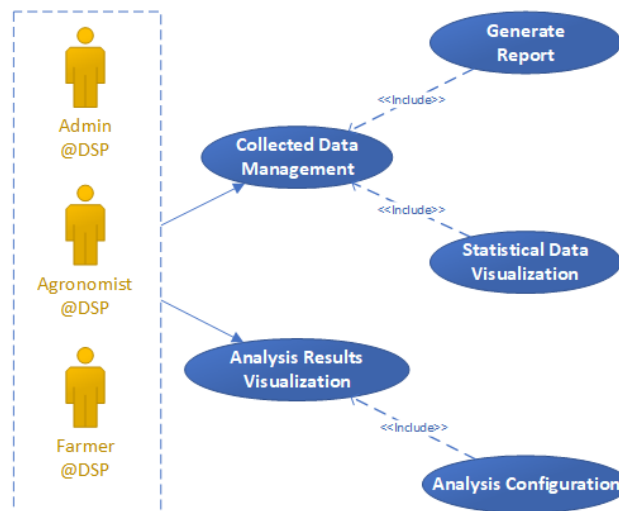


Figure 11 - Use case diagram, data analysis

Here we the description of the use cases illustrated in the diagram in Figure 11:

UC-DSP-DataAnalysis-1.0.0

Collected Data Management: the operator opens the page of collected data from all the PANTHEON fields. The operator can perform searches and filtering operations to extract the desired data subset. Access to specific data depends on the user’s role.

UC-DSP-DataAnalysis-1.1.0

Generate Report: the operator, after selecting the data subset, can start the generation of a report. The report format can be chosen among the available ones.

UC-DSP-DataAnalysis-1.2.0

Statistical Data Visualization: the operator, after selecting the data subset, can start the generation of related statistics, graphs and charts.

UC-DSP-DataAnalysis-2.0.0

Analysis Results Visualization: the operator opens the page that displays the results of analysis performed by the PANTHEON system. The system uses all field data to perform the analysis activities, but the user can view only a subset of the results based on their system role. For example, a farmer should be not able to view sensitive result of other fields.

UC-DSP-DataAnalysis-2.1.0

Analysis Configuration: the operator can configure, where available, the parameters for data analysis processes execution. The permissions to view and edit parameters depend on the user’s role.

2.10 Use Case Schema “UC_DSP_Infrastructure”

This section contains the use cases related to the management of the system infrastructure.

In particular, Figure 12 shows the diagram of the use cases related to the application user’s management and the monitoring of the DCP’s and DSP’s status, performed by the infrastructure manager.

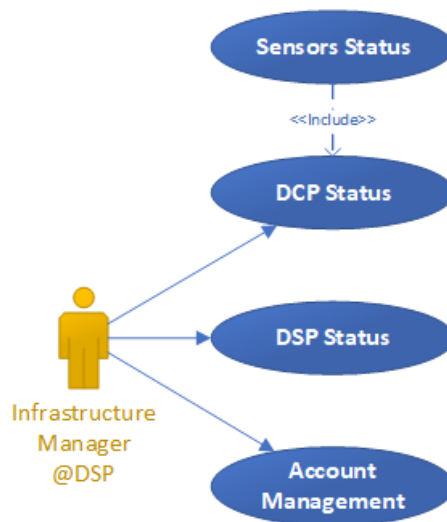


Figure 12 - Use case diagram, infrastructure management

Here we present the description of the use cases illustrated in the diagram in Figure 12:

UC-DSP-Infrastructure-1.0.0

DCP Status: the infrastructure manager opens the page containing the infrastructure status of the “Data Collection and Pre-processing” layer. The page should contain information on the farm server, UAV/UGV platforms and LAN network devices. The system keeps an infrastructure status log, so the available data are both current and past.

UC-DSP-Infrastructure-1.1.0

Sensor Status: the operator can open a specific page to monitor the status of sensors installed in the hazelnut field. These data concern soil moisture sensors, weather station sensors and, eventually, all sensors installed on the UAV/UGV platforms.

UC-DSP-Infrastructure-2.0.0

DSP Status: the operator opens the page containing "Data Storage and Processing" layer infrastructure status. The page should contain information on the central server and the storage system. The system keeps an infrastructure status log, so the available data are both current and past.

UC-DSP-Infrastructure-3.0.0

Account Management: the operator opens the PANTHEON application users and farms management page. The infrastructure manager has the responsibility to create, modify, delete users from the system and assign them specific roles. Hazelnuts fields adhering to the PANTHEON system registration and also associations between users and farms creation belong to this activity.

3 Functional Requirements

Functional requirements for PANTHEON end-user application are listed below. During this project phase requirements refer to web app in “Farm” mode.

The requirements listed here are divided into paragraphs, grouped by areas of interest.

3.1 General

REQ-FF-00.01

Application should be developed using web technologies. In detail, the front-end layer should be based on HTML, CSS and JavaScript tools, while the back-end layer should contain micro-service features.

REQ-FF-00.02

Application should provide users with a sign-in panel for authentication. At least one role is associated to each user. Roles are groups that identify data access grants for users.

REQ-FF-00.03

Application should grant access only to data for which user is enabled. Moreover, according to the operator role, user interface should show only a subset of farm panels and information.

REQ-FF-00.04

User interface should provide a navigation menu to grant access to the various application sections.

REQ-FF-00.05

Application user interface should be organized in several sections. In detail, the following section should exist:

- Farm
 - Dashboard (Main page)
 - Production
 - History
- Data Acquisition
 - UAV
 - UGV
 - Manual
- Agronomic activity
 - Pruning
 - Sucker Control
 - Water Management
 - Pest & Disease
- Planning
 - Mission
- Real-Time
 - IoT Sensors
 - Monitoring

REQ-FF-00.06

Aggregated data view on field map through heatmaps should be allowed.

REQ-FF-00.07

System warning messages should be notified to the user. They should refer to events like:

- Soil moisture threshold exceeded
- Air temperature threshold exceeded
- Sensor malfunctions
- UAV/UGV mission in progress

REQ-FF-00.08

Application should ask the user's confirmation every time a database elements creation or a deletion operation is submitted (i.e.: new activity creation or mission removal from calendar).

REQ-FF-00.09

Field maps should be accessible both in interactive and in static mode. In interactive mode, user interacts on map component by mouse and selects available elements, engaging related tasks; in static mode, map shows where the element under analysis is located.

3.2 Farm

REQ-FF-01.01

"Farm" section should display general farm data and synthetic indices.

REQ-FF-01.02

Farm information should be displayed, such as:

- Owner
- Geographical position
- Field extension in acres
- Cultivated hazelnut varieties

REQ-FF-01.03

Farm geographical position should be represented on cartographic map in user interface.

REQ-FF-01.04

Generally, farm fields are organized in sectors and rows. The application should be able to manage this configuration.

REQ-FF-01.05

The application should be able to manage groups of trees and other groups different than default ones.

REQ-FF-01.06

Each tree group is identified by a unique name and should include the following elements:

- The list of the trees
- Geographical coordinates for the polygon enclosing the selected group
- Trees deployment pattern layout (for sector only)

- Trees age (for sector only)

REQ-FF-01.07

The application should manage several configurations related to the same physical farm.. For example, the PANTHEON project test scenario is a logical configuration for the Stelliferi's hazelnut field, and the configuration includes a tree subset.

REQ-FF-01.08

The application should display information on hazelnut phenological phases.

REQ-FF-01.09

A panel with areal weather forecasts should be displayed on farm main page.

REQ-FF-01.10

For any activity (either agronomical task or data collection) the related target should always be defined..

3.3 Tree

REQ-FF-02.01

Application should provide hazelnut field information details up to the specific tree level.

REQ-FF-02.02

Both static and dynamic information on each single tree should be displayed. In detail:

- Static data: for example, ID, position, species, year of plantation, etc.
- Dynamic data: for example, timestamp, 3D model, estimated harvest, pest & disease index, phenology, water stress, agronomical actions, etc.

REQ-FF-02.03

Each tree should have a unique ID and specific GPS coordinates to identify it on the map.

REQ-FF-02.04

Detailed information for each single tree should be displayed. In particular:

- A three-dimensional tree model representation
- Last acquired scan information
- Tree properties
- Executed and planned activities for the tree

REQ-FF-02.05

Three-dimensional representation model should always refer to the last available scan. Previous scans should be part of historical data.

REQ-FF-02.06

When user selects a date earlier than the current one, last three-dimensional model produced before selected date should be displayed.

REQ-FF-02.07

The application should allow to compare three-dimensional representations of the tree belonging to two different dates to analyse tree evolution due to agronomic activities.

3.4 Harvest

REQ-FF-03.01

In harvest management, hazelnut varieties should be distinguished. In detail, at least these varieties should be recognized:

- “Tonda Gentile Romana”
- “Nocchione”
- “Tonda di Giffoni”
- “Tonda Gentile delle Langhe”
- “Mortarella”

REQ-FF-03.02

The application should display harvest forecast for current year.

REQ-FF-03.03

If harvest is performed in more than one step, the application should display information for all harvesting sessions.

REQ-FF-03.04

Estimation accuracy level should be displayed together with harvest estimate, when available.

REQ-FF-03.05

Harvest estimate is updated after new acquisitions; evolution of the historical estimate should be displayed.

REQ-FF-03.06

For harvest estimate, each sector estimate should be displayed. In this scenario, it isn't relevant to show the single tree information.

REQ-FF-03.07

Final harvest data should be displayed. This information should also be detailed grouping harvested hazelnuts by:

- Varieties
- Fruit net weight and nutshells
- Quality

REQ-FF-03.08

If the system provides market quotations, harvest economical return should be displayed by the application.

3.5 History

REQ-FF-04.01

In “History” section, the whole farm historical data should be available.

REQ-FF-04.02

“History” section should allow to analyse archived farm information. In detail:

- Production
- Data acquisition activities
- Agronomic tasks
- Missions performed on the field
- Sensors collected data

REQ-FF-04.03

Data (like temperature, moisture, rainfalls, etc) should be displayed using timeline graphs. Timeline visual comparison should also be allowed.

REQ-FF-04.04

All completed activities and missions, in “executed” state, should be part of historical data.

3.6 Data Acquisition

REQ-FF-05.01

Users should be allowed to add manual acquisitions to the system, such as:

- Tree health observations
- Bug detection
- Damages involving wildlife animals, like boars or squirrels
- Sucker detection or pruning needs for selected trees or areas

REQ-FF-05.02

Adding information concerning any manual acquisition should be allowed.

REQ-FF-05.03

Photo attachment together with text input should be allowed for any manual acquisition.

REQ-FF-05.04

UAV and UGV acquisition mission configuration should be available in the data acquisition section.

REQ-FF-05.05

UAV/UGV mission configuration should need to specify at least:

- Mission type
- Acquisition target (from what trajectory waypoints will be retrieved)
- Activity start and duration time
- User in charge of the task

REQ-FF-05.06

The following UAV/UGV acquisition task status should be managed:

- Ready: acquisition task has been configured
- Planned: acquisition task is included in a mission, meaning it has been planned, but not executed yet
- Executed: task completed

REQ-FF-05.07

For manual acquisition the only available state is “executed”.

3.7 Agronomic activity

REQ-FF-06.01

The following agronomic activities status should be managed:

- Suggested activities that have not been yet performed by an agronomist, who will later approve or reject them if they consider them unnecessary
- System suggested activities already approved or manually inserted activities
- Activities that have been included in a mission, meaning they have been planned but not yet executed
- Executed activities
- Suggested activities rejected by the agronomist

REQ-FF-06.02

The application should let the user apply custom filters to the activities list specifying different values or ranges.

REQ-FF-06.03

New activities should be added manually.

REQ-FF-06.04

Detailed information for each task should be displayed. In particular:

- Cartographic map identifying activity target
- Activity type
- Activity specific properties
- Activity status
- Comments or notes

3.8 Pruning

REQ-FF-07.01

In single tree pruning definition, the application should let user interact with tree branch three-dimensional structure model. In detail, user should be able to:

- Rotate and zoom in/out
- Select single branches within the tree structure
- Mark branches to set pruning type

REQ-FF-07.02

The branch three-dimensional model should highlight tree section that is going to be removed by user selected branch cut.

REQ-FF-07.03

When a new pruning activity is created user should specify task target as first step.

REQ-FF-07.04

When user sets a single tree as pruning activity target, the application should display the three-dimensional tree model to allow marking of the cutting branch.

REQ-FF-07.05

When a tree cluster is set as pruning activity target, the application should let the user choose the cutting mode (like edge or topping).

REQ-FF-07.06

“Pruning” section should display the pruning activities list grouped by task status.

REQ-FF-07.07

Pruning activities state should be:

- Activities suggested by the system, yet to be defined and planned
- Activities defined but not yet planned
- Activities include in a planned mission
- Activities in progress, the UGV is marking pruning branches; task owner should confirm real ending, meaning that branches have been really removed by UGV or field worker
- Pruning completed

3.9 Sucker Control

REQ-FF-08.01

“Sucker Control” section should display, on cartography, detected suckers’ field heatmap.

REQ-FF-08.02

When creating a sucker control task, user should define herbicide type and quantity for each tree.

REQ-FF-08.03

For system suggested sucker control activities, recommended herbicide quantity should be displayed.

REQ-FF-08.04

“Sucker Control” section should display herbicide dispense activity list grouped by task status.

REQ-FF-08.05

Available status for sucker control activity should be:

- System suggested herbicide spraying
- User approved but not yet planned spraying

- Planned herbicide spraying
- Activity in progress, UGV is spraying the herbicide; activity owner must still confirm task ending
- Activity completed

REQ-FF-08.06

Application should allow user to verify herbicide spraying efficiency on any single tree, through two three-dimensional scan comparison, the first before, and the second after the operation.

37

3.10 Water Management

REQ-FF-09.01

In irrigation section, detailed information about weather should be displayed. In detail:

- Rainfall
- Minimum, maximum and mean air temperatures
- Daily temperature trend on an hourly base
- Next day's weather forecast

REQ-FF-09.02

In irrigation management section, soil water stress status should be displayed on cartography through heatmap.

REQ-FF-09.03

In irrigation management section, a past climate indices summary should be available, for example showing a timeline graph to the user.

REQ-FF-09.04

The "Water Management" section should display irrigation activities list grouped by task status.

REQ-FF-09.05

Water management activity status should be:

- System suggested irrigation activity
- Irrigation activity approved by the user but not yet planned
- Irrigation activity planned
- Irrigation in progress, irrigation system active; task owner should confirm effective task ending
- Irrigation completed

REQ-FF-09.06

Application should take into account the field irrigation system configuration, which it affects the task target definition.

REQ-FF-09.07

Main setting parameter in activity definition should be the irrigation duration. Flow rate and irrigation point positions are "Farm" properties and cannot be modified by user during task definition.

3.11 Pest and Disease

REQ-FF-10.01

Application should display available product list for pest control treatment.

REQ-FF-10.02

Application should record all treatments executed during the year and warn if thresholds for any product have been exceeded.

REQ-FF-10.03

System should provide the users with suggestions on default treatment when a disease has been detected, according to available bibliography.

REQ-FF-10.04

“Pest & Disease” section should display list of pesticide spraying activities grouped by task status.

REQ-FF-10.05

Pest control activities status should be:

- System suggested pest control activity
- Pest control activity approval/definition, by the user but not yet planned
- Planned pest control spraying mission
- Mission in progress, UGV platform is spraying pest control in selected points; task owner should confirm effective task ending.
- Spraying completed

3.12 Missions

REQ-FF-11.01

Application should allow user to create missions to plan homogeneous activities execution, meaning tasks that can be carried on by a specifically configured UGV platform.

REQ-FF-11.02

Application should manage the following mission states:

- Planned
- In progress
- Completed (executed or cancelled)

REQ-FF-11.03

User interface should display a calendar widget were all planned missions are reported.

REQ-FF-11.04

Application should allow user to activate monitoring mode for the missions “in progress”.

REQ-FF-11.05

Application should allow user in charge of mission to confirm operation start.

REQ-FF-11.06

Application should allow user in charge of mission to confirm operation completion, letting them insert notes if necessary.

3.13 IoT Data

REQ-FF-12.01

Application should display information coming from on field weather station, in detail:

- Wind chill
- Rainfalls
- Air temperature
- Atmospheric pressure

REQ-FF-12.02

Application should display data coming from soil moisture sensors.

REQ-FF-12.03

Soil moisture sensors measure data at different depth levels, so application should display them for all levels.

REQ-FF-12.04

In the “IoT Data” section field sensor position should be displayed on map.

REQ-FF-12.05

Application should highlight values exceeding configured thresholds.

REQ-FF-12.06

Application should display sensor collected values in a real-time updated graph.

3.14 Monitoring

REQ-FF-13.01

Application should allow mission monitoring, especially UAV and UGV platform operations.

REQ-FF-13.02

In “Monitoring” section, UAV/UGV platforms, and related on board sensors, live state should be displayed.

REQ-FF-13.03

UAV and UGV positions should be shown in real-time over cartographic map.

REQ-FF-13.04

In “Monitoring” section, irrigation system state should be displayed.

REQ-FF-13.05

“Monitoring” section should be available only when missions are in progress.

4 Interface Requirements

4.1 Front-end Back-end Interface

REQ-IF-01.01

Front-end and back-end should be able to mutually communicate through common internet or intranet networks.

REQ-IF-01.02

When the client is physically located on field, only an intranet communication may be required; while, when client is elsewhere, an internet access should be available.

REQ-IF-01.03

Front-end and back-end exchange messages using webservices and web socket features.

REQ-IF-01.04

Webservices should be used by front-end to request information to back-end and to wait for its replies.

REQ-IF-01.05

Web sockets should be used by back-end to send notifications to front-end when new information concerning tasks that front-end should display was published through the ROS network.

REQ-IF-01.06

Webservice mutual communication data must comply to the JSON protocol.

4.2 Back-end Database Interface

REQ-IF-02.01

Back-end should be able to connect to a MongoDB database, both in local and in remote way.

REQ-IF-02.02

Full access to database should be granted to back-end user, both in writing and in reading tasks.

REQ-IF-02.03

Remote MongoDB database should be reachable on a static IP address through fixed TCP protocol port.

4.3 Back-end ROS Interface

REQ-IF-03.01

Back-end should be able to read data published on a ROS network.

REQ-IF-03.02

Back-end should only act as subscriber, due to the fact that no publishing tasks on that network is required.

5 Non-functional Requirements

5.1 Access Security

REQ-NF-01.01

The application provides 4 security levels for the processed data:

- level 0) data, in read-only mode, can be viewed by everyone, both registered and unregistered users.
- level 1) data accessible by all registered users after login with own credentials.
- level 2) specific data accessible only by users with specific assigned role.
- level 3) system data accessible by the administrator only.

REQ-NF-01.02

The role assigned to the user drives types of information the operator can view or modify.

REQ-NF-01.03

System administrators are in charge of assigning roles to the users.

REQ-NF-01.04

Authentication management shall be delegated to third-party services.

5.2 Availability

REQ-NF-02.01

At the “prototype” maturity level, the application has to be reachable 24/7 with percentage of at least 90%.

REQ-NF-02.02

At the “product” maturity level, the application has to be reachable 24/7 with percentage of at least 99%.

5.3 Confidentiality

REQ-NF-03.01

Anonymous users, and those who are not on assigned role, do not have visibility of the other users' data (for example, you will be able to view details of an activity but you cannot view who planned or will perform it).

5.4 Integrity

REQ-NF-04.01

Data coming from the smart farming system will not be altered in any way by the application. They will be acquired only to render the information comprehensible to operators, through the implementation of an appropriate graphic user interface.

5.5 Usability

REQ-NF-05.01

The user interface, at “prototype” level, will be developed in English, so understanding this language will be a requirement to use the software.

REQ-NF-05.02

The user interface should be structured according to the modern command and control platforms. This includes the presence of sections for all types of activities and a navigation menu.

REQ-NF-05.03

The user interface should be built using components and graphic layouts commonly used in modern web applications, to make it easy to use.

5.6 Flexibility

REQ-NF-06.01

The application should be structured so as to foresee the use of multiple languages in the future. The user can choose the preferred language in their profile settings page.

REQ-NF-06.02

The layout frame and the development philosophy of the user application should be implemented so that the software can adapt to other smart farming scenarios, in particular to other crops.

REQ-NF-06.03

The application should implement the configuration management of the agricultural field using an appropriate level of abstraction to be valid for any general field of hazelnuts, not only the field selected for the validation of the project.

5.7 Maintainability

REQ-NF-07.01

The web application, reachable via Internet, must be installed on DSP to ensure a high degree of maintainability.

REQ-NF-07.02

For the web application installed on DCP (Farm mode) a manual update and maintenance procedure should be provided, give that it is not possible to rely on the Internet connection for remote or automatic updates/maintenance.

5.8 Modifiability

REQ-NF-08.01

The application design should have a clear design separation between front-end and the back-end components.

REQ-NF-08.02

All the configuration parameters, used by the application, should be stored into the PANTHEON system database. No configuration parameter should be hard coded in the source code.

43

5.9 Verifiability

REQ-NF-09.01

The application development environment should provide the connection to a test database to verify the implementation without compromising the data stored in the production database.

REQ-NF-09.02

During unit test phase, the front-end software should be able to access local test data without relying on back-end service calls.

REQ-NF-09.03

The services, on back-end side, should be implemented exposing a programming interface in order to be tested by dedicated tools, in the absence of the front-end software.

5.10 Installability

REQ-NF-10.01

As a web application, the software should be installed by the system administrator on the target web server, i.e. on DCP, in case of an application installed locally at the farm, or on DSP, in case of the application accessible via Internet.

REQ-NF-10.02

If the application is installed locally at the farm, it has to be configured to associate the software with the identifier of the field to be managed.

REQ-NF-10.03

The application should have a version number associated with the release.

5.11 Interoperability

REQ-NF-11.01

The application should be accessible by the most common web browsers, running on devices both desktop and mobile.

5.12 Portability

REQ-NF-12.01

The application should be developed with responsive web technologies, in order to be usable regardless of operating system installed on the device, both desktop and mobile.

5.13 Reusability

REQ-NF-13.01

The web application should be developed using standard software components that can be customized according to the context.

REQ-NF-13.02

The application should be developed following a structure and an approach (web framework) that can be reused in other smart-farming contexts with non-radical modifications.

5.14 Computer Resource

REQ-NF-14.01

The application should be usable by any device connected to the network with an installed browser compatible with HTML5, CSS3 and JavaScript.

5.15 Design and implementation constrain

REQ-NF-15.01

The application should be deployed on an “Apache” web server installed on “Ubuntu” 16.04.

REQ-NF-15.02

The front-end module has to be developed using the “Angular” web framework.

REQ-NF-15.03

The back-end can be developed with any language allowing to interface with MongoDB database, ROS and expose services accessible from the front-end module.

5.16 Precedence and Critical Requirements

REQ-NF-16.01

The end-user web application interfaces with the PANTHEON system through the data stored in the database and through the data coming from the ROS messaging publish / subscribe system. The web application cannot be properly used without the presence of these two data sources.

6 Application Architecture

6.1 Overview

Figure 13 shows the architecture of the end-user application [2].

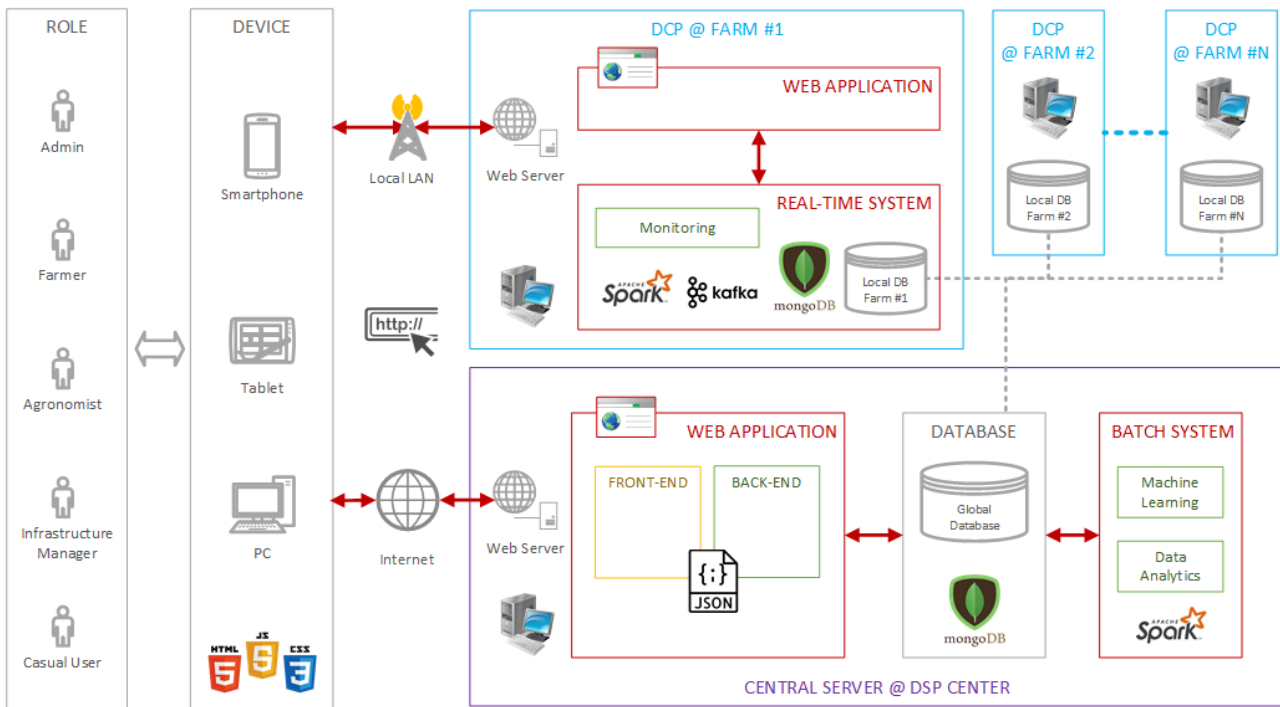


Figure 13 - Architecture of the end-user application

The application is Web-based and provide to the users two main features, which are supported by two sub-components of the software system:

- the real-time monitoring system, which operates in each hazelnut farm on the data collected locally, combined appropriately with the results of large-scale analyses carried out in the central system. This component is in charge of producing the indicators mentioned in previous sections able to describe the current status of the field, such as the water stress of trees and the presence of pests or diseases; this system will also monitor the weather conditions and their impact on the health of the plantation;
- the batch processing system, which support decision making and predictive analysis by operating in the DSP centre on all the available data that has been collected in various hazelnut fields and stored in the central server. This component is in charge of applying analytics and machine learning techniques for knowledge discovery over agricultural data such as time-series analysis, data clustering, automatic classification, and outlier's detection, with the already mentioned goals of product estimation and automated prediction, among others.

A data anomaly detection algorithm will also be developed on the batch processing system to detect malfunctions of the SCADA infrastructure. It will include data driven and model-based approaches to validate the gathered measurements. This validation will exploit the correlation between the measurements provided

by the ground and aerial robots, as well as statistical change detection/isolation algorithms to perform the early detection of malfunctions. This will allow maintenance operations before a fault has a significant effect on the system.

Many users have different roles in accessing system features via a Web browser and through different devices. The application is equipped with a front-end, which present the result of data analyses to the user, and a back-end, which accesses the database (local or central) and manipulate the data stored in it. The front-end will be designed with the goal of producing modern, easy-to-use and standard-based user interfaces.

Finally, driven by the software and hardware requirements, the following choices have been made for data organization and for the software solutions used for the development and usage the application:

- All the data will be represented in JSON, an open-standard file format that can be used for describing both structured and unstructured information;
- The batch will be implemented using Spark, an open-source distributed and general-purpose framework for data analytics over big data that supports machine learning;
- For implementing the real-time system, we will use Kafka, an open-source stream-processing platform for handling efficiently real-time data feeds, and Spark Streaming, a component of Spark supporting real-time analysis over stream;
- As we have already mentioned, both the local and the global database will be implemented using MongoDB, an open-source, NoSQL database management system that stores and manipulates data in JSON format;
- Linux will be used as the operating system of all the computers, C++ will be used for the automation of robots, and Python will be used as the host language for data processing;
- HTML, CSS and JavaScript will be used for the implementation of the client component of the application.

6.2 Required States and Modes

The PANTHEON user application has a double soul. The first one, the user connects his client device, though Internet connection, to the Global web server. In this case, the DSP database will be the reference data source (with possibility to access to the aggregated data of all the hazelnuts farms).

The other one, the user connects his client device to the Farm web server. In this case, the Farm database will be the reference data source. It is possible to connect to the Farm web server directly on site using the installed PANTHEON LAN or remotely via Internet only if the Farm web server has an available Internet connection.

Summarizing, the application has two operative modes determined primarily by the used data source:

- Farm Mode
- Global Mode

Regardless of the operating mode, the application will be, always into a well-defined state. For the PANTHEON web app, the following operating states are identified:

- Not Logged
- Farm Data
- Farm Data Edit
- Activity Planning
- Activity Planning Edit
- IoT Monitoring
- UGV/UAV Monitoring
- Farm Data Analysis (View / Edit)
- Global Data Analysis (View / Edit)
- Configuration Management (View / Edit)
- Infrastructure Management (View / Edit)

The following Table 1 shows the available states of the app in the different operating modes:

APPLICATION	STATES										
	NOT LOGGED	ACTIVITY PLANNING		FARM DATA		MONITORING		DATA ANALYSIS		MANAGEMENT	
		V I E W	E D I T	V I E W	E D I T	IoT	UGV UAV	FARM	GLOBAL	CONFIGURATION	INFRASTRUCTURE
FARM	X	X	X	X	X	X	X	X		X	
GLOBAL	X	X		X				X	X		X

Table 1 - Application States and Modes

6.2.1 Modes

The following Figure 14 shows the two operating modes of the PANTHEON end-user application:

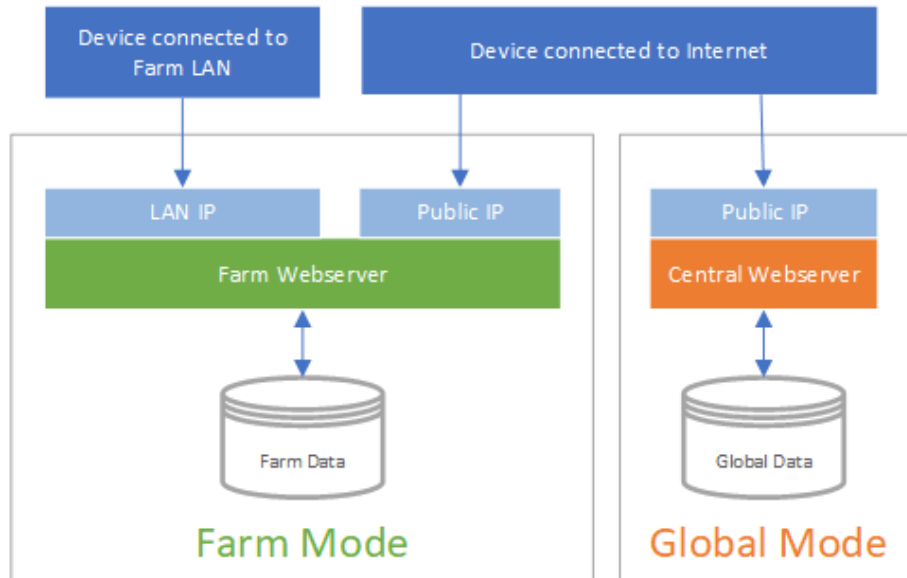


Figure 14 - Web App Modes Diagram

The diagram in Figure 14 shows the two operating modes of the PANTHEON application: the “Farm” and the “Global” mode. In particular, 4 logical layers can be identified: device, network, web server and database.

The real difference, between the 2 modes, is the web server where the application is running and, consequently, the data source used. When the user connects to a farm web application, installed on the DCP workstation, the application operates in “Farm” mode, when the user connects to the application installed on the central web server installed on the DSP, the application operates in “Global” mode.

6.2.2 States

The following Figure 15 shows the states diagram of the PANTHEON application:

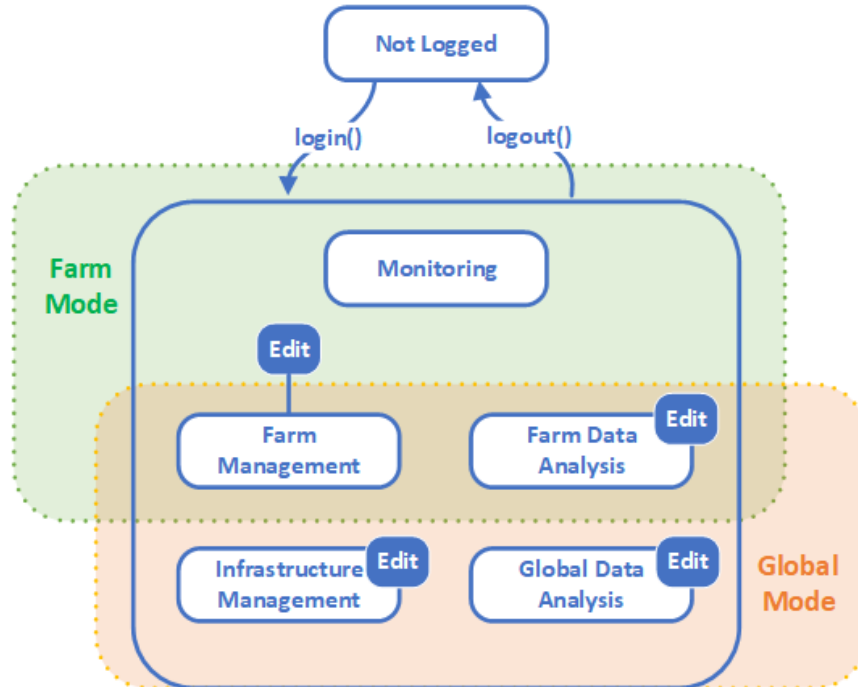


Figure 15 - Web App States Diagram

The diagram shows the states of the PANTHEON end-user application. Excepting “Not Logged”, the other states are grouped according to the operating modes, as indicated in the Table 1.

In the “Not Logged” state the application contents are public. In this state, a casual user can read information about hazelnut fields, in particular the data subset made public by administrator, for example the geographical location of the farm, current weather, type of cultivation, extension of the field, etc. To perform any considerable operation, the user should “login” the system with own credentials. Vice versa, with “logout” action, the operating status returns to “Not Logged” state.

The states, allowing to perform considerable operation, are grouped inside the hide “Logged” main state since, in principle, it is possible to go into a specific state through “login” action. In the same way the “logout” action can be performed from any state.

In Figure 15, there is additional four “Edit” states. Starting from the basic “View” state, it is possible to enter to (corresponding) data editing state, generally with the command “modify” or “new”. From the “Edit” state you can only go back to the starting state through the “save” or “cancel” command.

Following, the description of the 8 states identified.

Farm Management

This is the main state of the application and it allows management of the farm data and the agronomic operations to be performed on the hazelnuts field. Editing of information about the hazelnut field, manual surveys, configuration of data acquisition missions, definition and scheduling of agronomic operations are activities that can be performed in the “Farm Management” state. Some data are specific and protected, so,

only authorized user can gain access to them. The authorization depends on the user's assigned role. On the same way, authorization to enter in the "Farm Management Edit" state, to create and modify the data, depends on the user identity. The management of farm data and agronomic activities involves alteration of the local database. Without guaranteed connection between local farm server and the central server, data consistency cannot be insured, then, the editing is enabled only in the "Farm" mode.

Monitoring

This state is active when the application is connected to the IoT sensor network or when a data collection mission is in progress. Data flow comes from the system in real time and the user can examine values of the IoT sensor network installed on the hazelnut field. On the other case, UAV and/or UGV are working into the field and the user can monitor platforms and onboard sensors properties in real time. This real-time monitoring state is available only in "Farm" mode. In the "Global" mode, users can read values coming from sensors network, but don't confuse that with the real-time monitoring state of the application. In particular, IoT data can be read in the "Farm Management" state. Main reason to exclude real-time monitoring statuses from the "Global" mode is that there are no guarantees about connection between farm and the central server, which obviously would make impossible the real-time monitoring.

Farm Data Analysis

The application becomes an analysis console in this state. User interface shows processing results and allows to configure analysis activities in corresponding "Edit" state. This type of analysis will be performed using information stored on farm database. So, this type of analysis should be performed usually in "Farm" mode. In "Global" mode the "Farm Data Analysis" is also available, and it uses the data transferred from the farm database.

Global Data Analysis

The application becomes an analysis console in this state. The user interface shows processing results and allows to configure analysis activities in corresponding "Edit" state. This type of analysis will be performed using information stored on DSP about all the hazelnut fields managed. This type of analysis will be performed only in "Global" mode.

Infrastructure Management

This special state, existing only in "Global" mode, allows the PANTHEON infrastructure. For example, it can monitor the sensor network, analyse the system logs or manage the user accounts.

7 Application Design

This section contains the application design elements in order to set up the Software Design Description base [5]. The application domain reference data model has been defined, the interface specifications between components have been detailed and, finally, the implemented user interface windows are represented, both in terms of mock-ups, designed in the preliminary phase, and in terms of end user application screenshots.

7.1 ORM Data Model

The data model, identified in the application design phase, is shown below. At this software development cycle maturity level, the detailed planning has mainly focused on those functionalities allowing hazelnut field management.

7.1.1 Farm

Figure 16 shows the data structure used to modelling the hazelnut field.

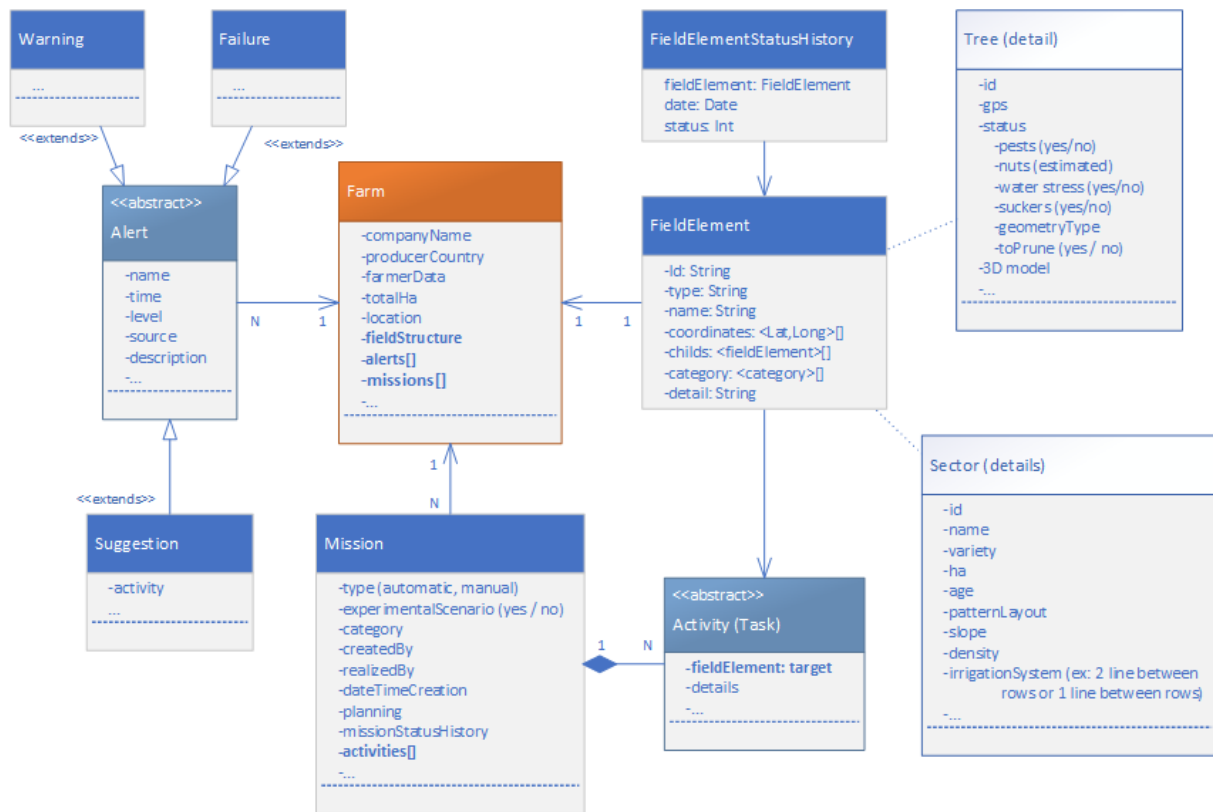


Figure 16 - Farm field model structure

The main application class is "Farm", modelling a farm involved in the hazelnut’s cultivation. For this reason, the corresponding class is highlighted in orange in the diagram. Representing the main domain object, the class appears in all the subsequent diagrams that model other application domain areas.

The farm partitioning into sectors, rows and trees is modelled using “FieldElement” class. Through the “FieldElement” class a hierarchical structure of the elements can be built. For example, in Figure 17, the

hierarchical structure modelling a farm partitioned in sectors is shown; each sector is partitioned in rows, and finally each row contains N trees.

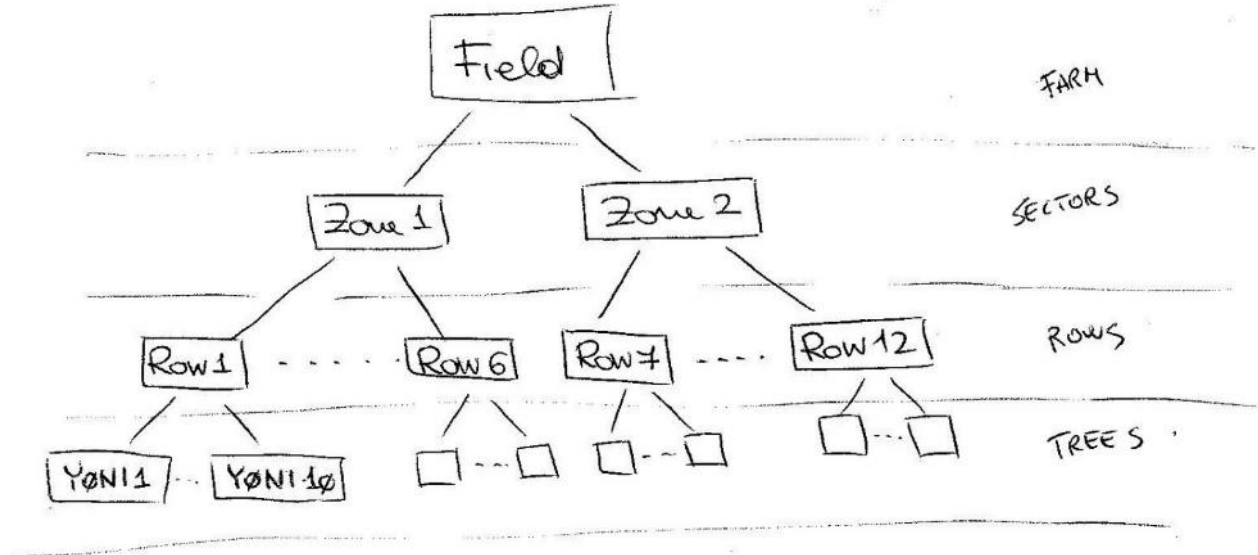


Figure 17 - Farm elements hierarchical structure

Each leaf element of the hierarchical structure represents a single tree in the field.

The "Alerts" are all the notifications that PANTHEON sub-system rise up relating to the farm. The alerts can be malfunction, warning, or suggestion for agricultural activities to be performed notifications.

The diagram also shows the "Mission" and "Activity" classes whose description will be provided in next diagrams. They have been inserted in this context to highlight the fact that each activity has a "target" and it corresponds to a "FieldElement".

7.1.2 Data Acquisition

Figure 18 shows a section of the data structure modelling the application data acquisition part.

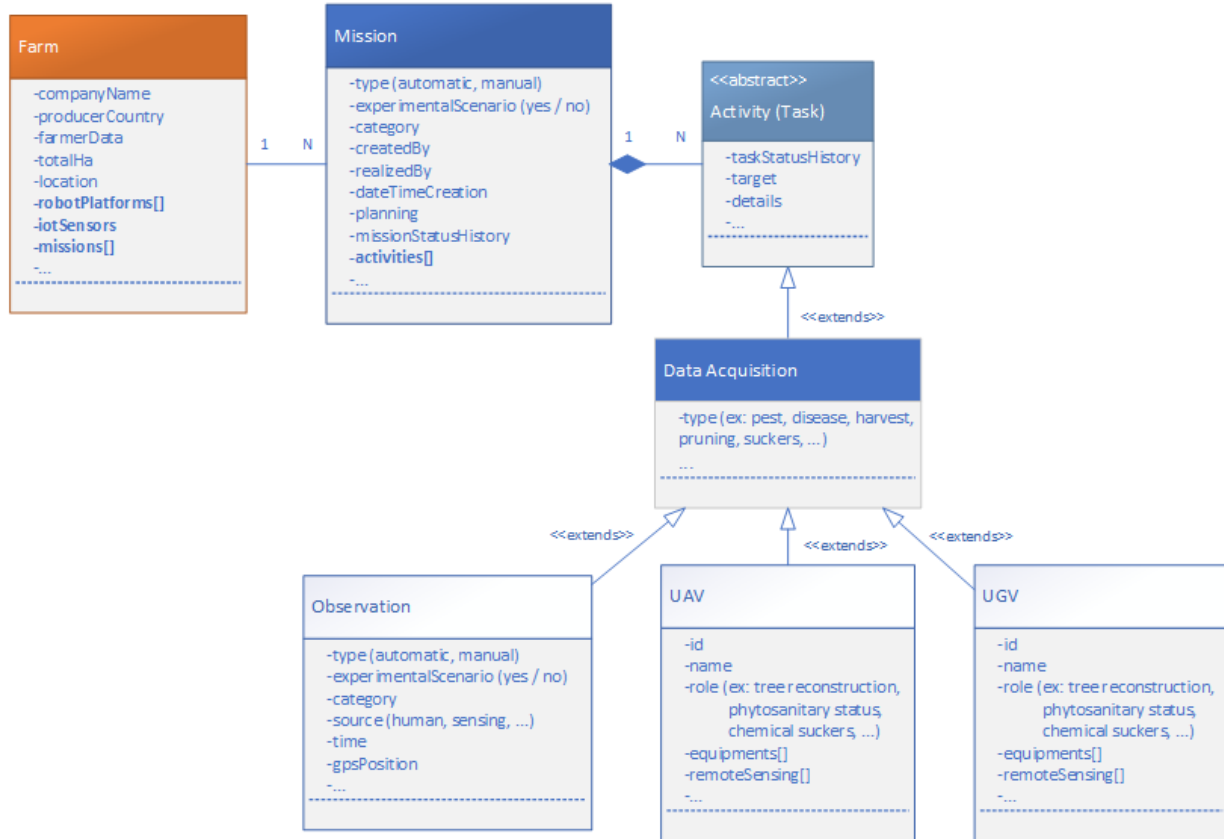


Figure 18 - Data acquisition model structure

The master class in this context is "Data Acquisition". At the end, this is an activity that the user can define and schedule as a mission itself. In this case we speak in term of data acquisition mission. A related example is the scans performed by UGV and UAV.

The manual surveys, or "Observation", are acquisition actions that are performed on the field at a specified moment by an agronomist (or any other human worker). In this case, the term "mission" is nonsense even if the acquisition has a planned date and time of when it should be performed.

7.1.3 Activity

Figure 19 shows the data structure modelling the agronomic activities management.

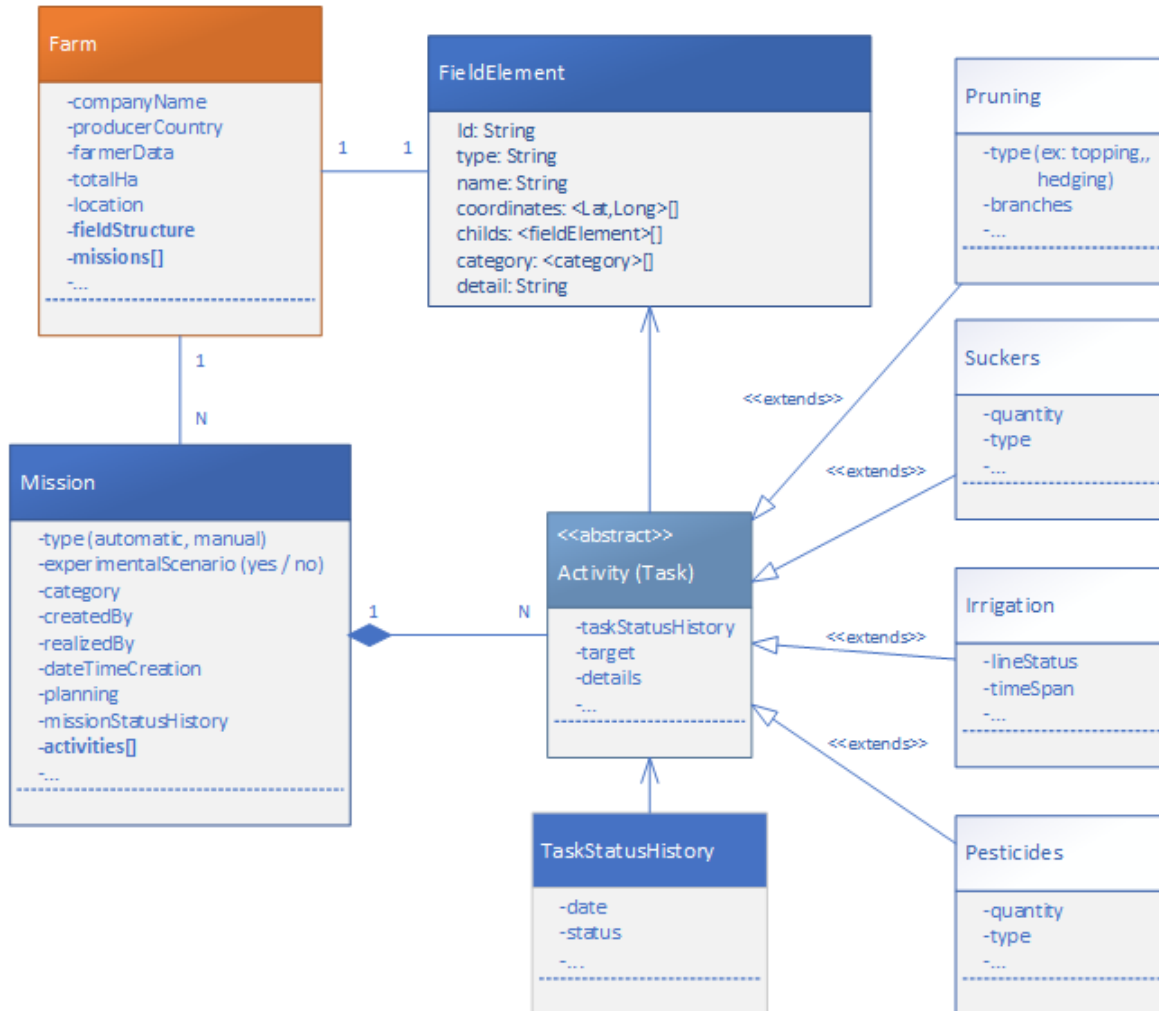


Figure 19 - Agronomic activity model structure

The master class in this context is "Activity" and represents the generic task. As previously seen, data acquisition is also a task, but in this case the diagram focuses on the agronomic activities.

The model allows the definition of any type of agronomic activity, extending the abstract "Activity" class. In PANTHEON context, the application should support the user in the pruning, sucker control, irrigation and plant health control management. For this reason, operations have been modelled through 4 corresponding classes, namely "Pruning", "Suckers", "Irrigation" and "Pesticides".

The diagram also shows the "Activity" relationships with the "Mission" and "FieldElement" classes. "Mission" represents the collection and scheduling of defined activities, while "FieldElement" is the agricultural operation target element; for example, a sector can be the target of an irrigation operation.

7.1.4 Mission

Figure 20 shows the mission management data structure modelling.

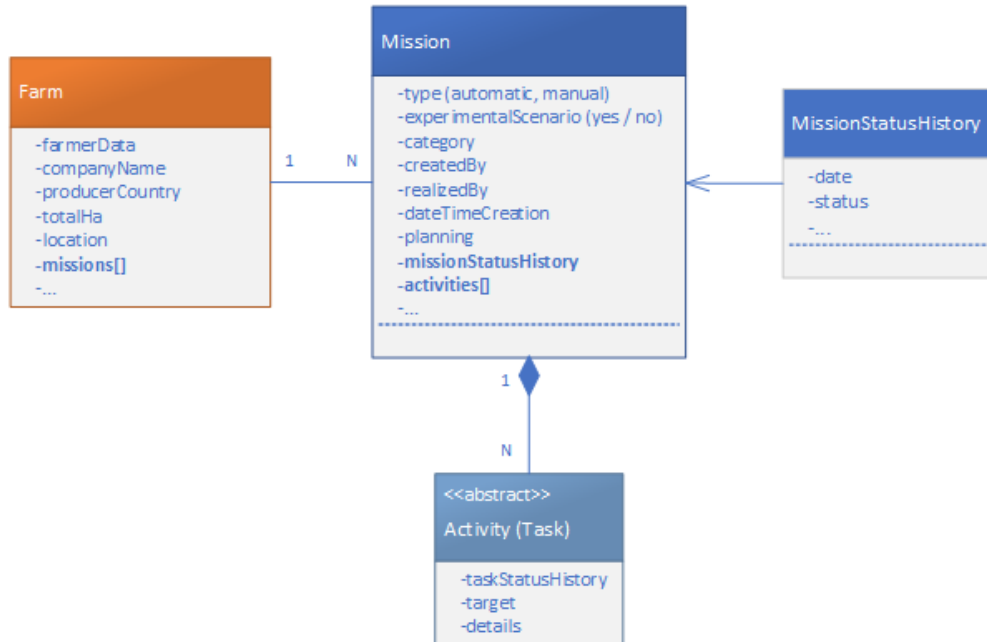


Figure 20 - Mission planning model structure

A "Mission" is a group of activities that must be carried out within a given time. Each mission has to be scheduled on the calendar and has its own life cycle, ending after all activity's execution is completed.

The activity execution sequence is not random, in fact the PANTHEON system establishes the execution order base on criteria computed by the planning algorithm. The "Mission" object should be able to get and manage the activities sequence order.

The "MissionStatusHistory" class allows to keep track of all mission state changes and to historicize the agronomic operations performed.

7.1.5 Monitoring

Figure 21 shows the data structure related to the real-time data monitoring coming from the IoT network sensors and from the aerial and terrestrial robotic platforms (UGV and UAV) that carry out field activities.

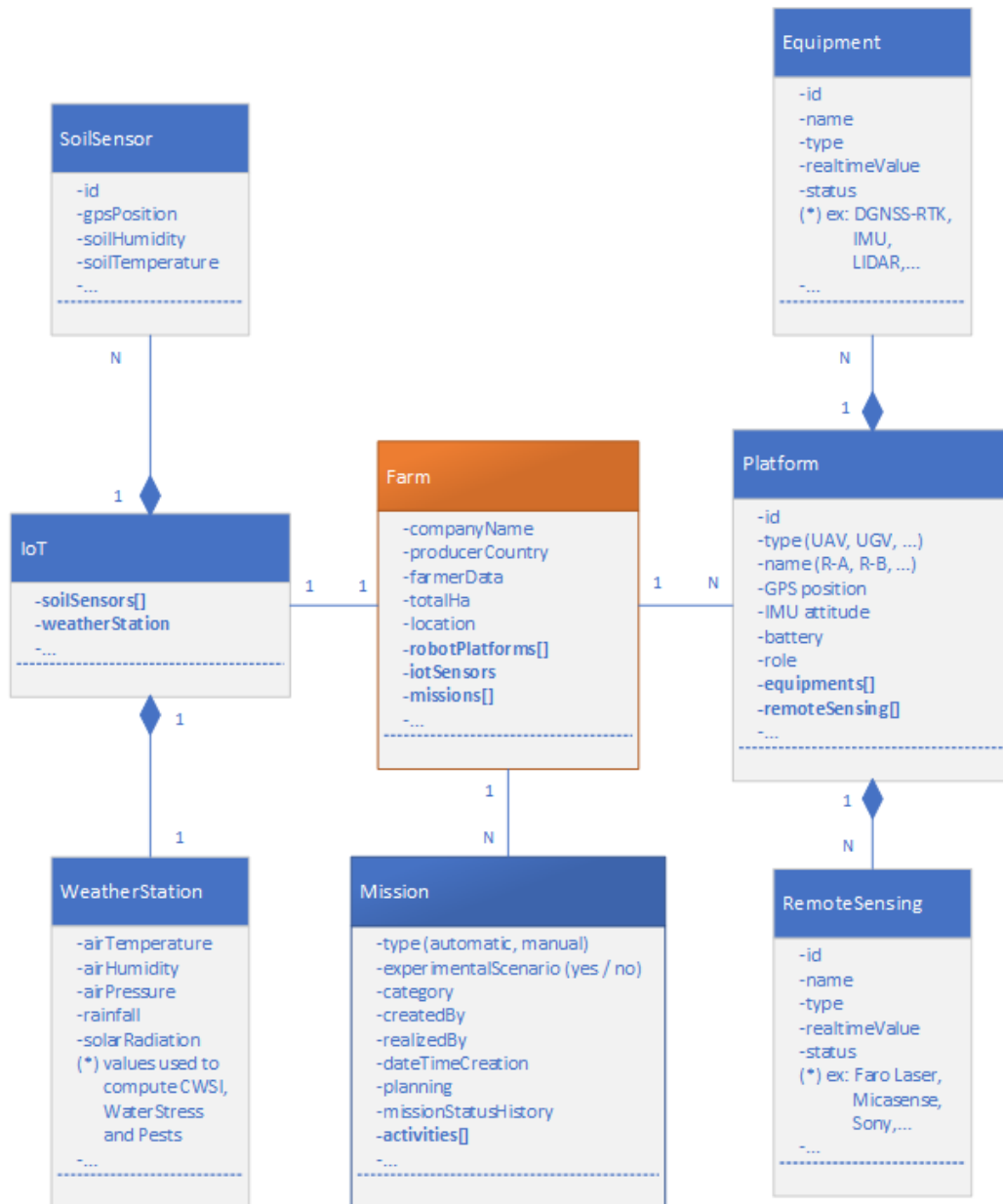


Figure 21 - Real-time data monitoring model structure

The two main classes in this representation are the "IoT" class and the "Platform" class.

"IoT" models all the static sensors installed on the hazelnut field. The sensor network is usually composed by a few weather stations, and some sensors for collecting data concerning the soil status, e.g., temperature and moisture.

"Platform" models a generic vehicle moving on the field, for example an UGV or an UAV. The class models also the main components of the robotic platforms, such as actuators and sensors for data collection. The

application processes real-time platforms positioning and sensors status to display collected data in the monitoring page.

NB: acquisitions performed by laser scanners or photo cameras are not involved in the real-time monitoring. This data, called “raw data”, will be sent and processed by the DSP layer. The application does not deal with the scan “raw data”.

7.1.6 Data Storage and Processing Centre (DSP)

Figure 22 shows the data structure related to the Data Storage and Processing centre.

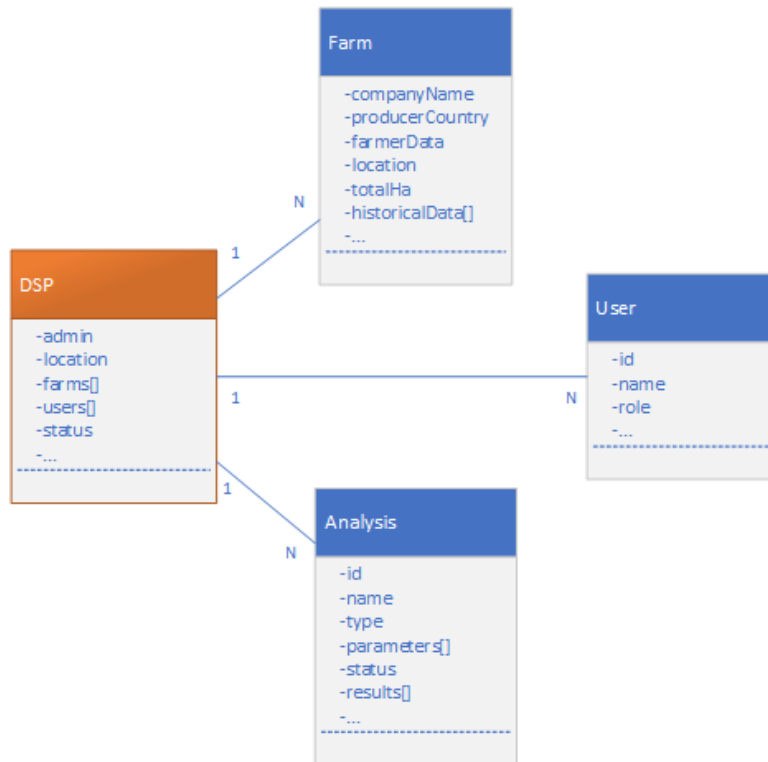


Figure 22 - Data Storage and Processing centre model structure

The diagram master class is "DSP" which contains information about the system central node. The structure allows to keep track of all the farms where PANTHEON system is installed on, manage registered users and monitor the status of the DSP infrastructure.

The "Analysis" class models any type of analysis that can be performed on the agronomic data collected from all the PANTHEON farms.

7.2 Interface Design Specifications

Back-end to front-end communication

Back-end receives requests from front-end. These requests get through web services functions that let both sides transfer data in a secure and simple way. Several versions of web service features are available on market nowadays. PANTHEON back-end will be compliant to REST protocols.

A web service is a web function that listens to new data arrival continuously and, when new requests have been sent, transmits data to the software body to let it elaborate and reply. Reply modes can be synchronous or asynchronous, meaning that, in the first option, service stays in pending status until the reply is ready to be broadcast and, after sending operation is over, comes back to listening level, while in the second, listening agent closes connection every time a new request is completed and returns ready to receive new requests. Replies will be sent when ready through a different service (sender should be polling on another service after request has been completed).

Front-end architecture requires that different services should be waiting for requests from front-end. These services might be synchronous or asynchronous depending on front-end needs.

Back-end to data storage communication

In order to send replies, back-end should interact with the data storage layer. The use of MongoDB drivers will be necessary to complete that task. The back-end environment lets developers use a specific library that is able to manage all the communication features with that specific database. So, any bidirectional interaction with the database (storing or retrieving data) can be easily managed by resorting to this library. The back-end will use this feature to manage required data and interact with selected databases through the MongoDB universe.

Back-end to notification environment

The notification management of required information to the front-end is also in charge to the back-end. It uses just a subscriber role in the ROS environment in this task. This means that when a notification that is supposed to reach the front-end layer travels through the ROS communication environment, the back-end should be able to catch it and deliver to the front-end properly. This can be accomplished by the back-end ROS toolkit for the interaction with the ROS environment, and using a web socket feature for sending data to the front-end

7.3 Prototype

7.3.1 User Interface Widget Mock-up

Application widgets and sections drafts are shown in this and the next paragraphs.

Those drafts have been created during design phase. Some components have been refined and modified during interface analysis and development to grant better usability.

Farm Map

The widget in Figure 23 shows the field map based on satellite view.

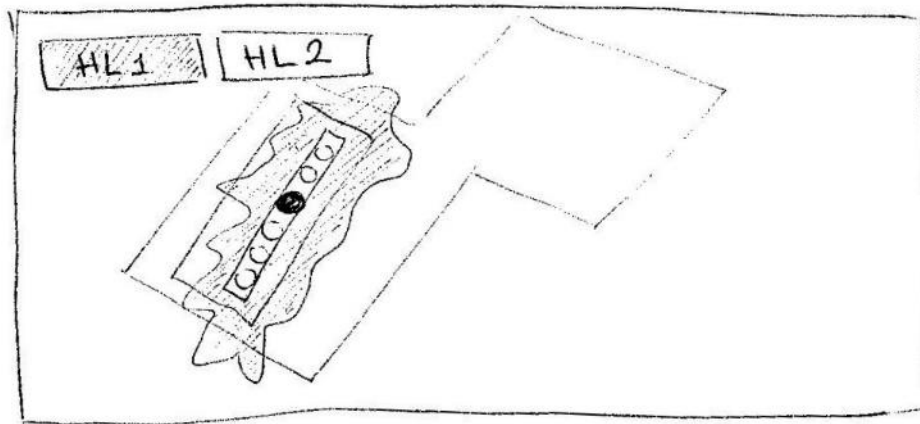


Figure 23 - Farm map widget mock-up

Field target elements, who have been hierarchically stored in field configuration, are drawn on map.

Map can be used in two ways: the interactive mode that lets the user select an activity target element or filter its search (i.e. monitoring, estimates, etc.); the non-interactive feature that lets the user view an element position within the field only.

Widget modes can be used like following:

- Interactive without target selection (Figure 24): a field area can be selected; elements within such area will be shown. That selection can be repeated down to the single tree layer.

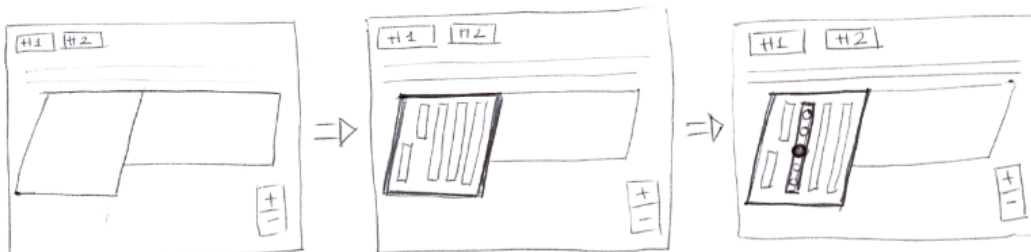


Figure 24 - Farm map widget, interactive mode

- Interactive with target selected (Figure 25): the specified target element, its owners and all the elements belonging to the same layer will be highlighted; different elements will be selectable using the interactive feature.

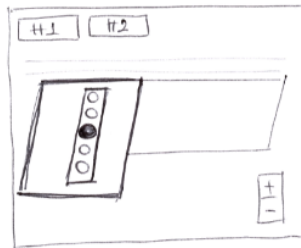


Figure 25 - Farm map widget, target selection in interactive mode

- Non-interactive with target selected (Figure 26): the specified target element and its owners only will be highlighted; no different element selection allowed.

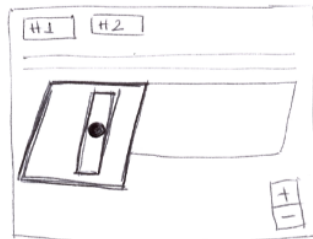


Figure 26 - Farm map widget, target selection in non-interactive mode

Different layers are available to show acquired based heatmaps (i.e.: water stress, soil temperature, fruit detection, pest damage); layers change accordingly to application section.

Info box

The widget in Figure 27 should always be associated to a target element selection widget (i.e.: farm map) and eventually with a date or date interval selection widget in page composition.

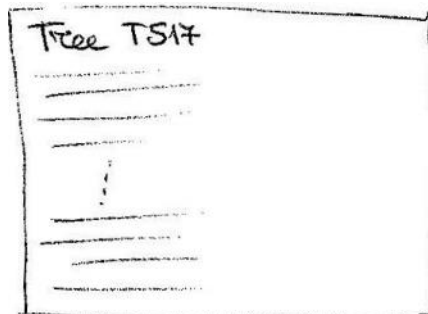


Figure 27 - Info box widget mock-up

Data based on application section and selected target will be shown in widget. A button for switching to element detail page will be available when widget is in application main pages, i.e. the tree detail page. Some information will send user to a data insert/modify form (i.e.: production effective estimation).

Date Selector

Component in Figure 28 shows a drop-down menu that displays acquisition date available.

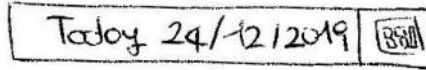


Figure 28 - Date selector widget mock-up

When a date is selected from drop-down menu, all the related components , on the same page, will show information which are related to the selected date.

On the other hand, when the user selects a "FieldElement", the "Date Selector" component shows only the "FieldElement" relevant dates.

Task list

Widget in Figure 29 shows the task list mock-up.

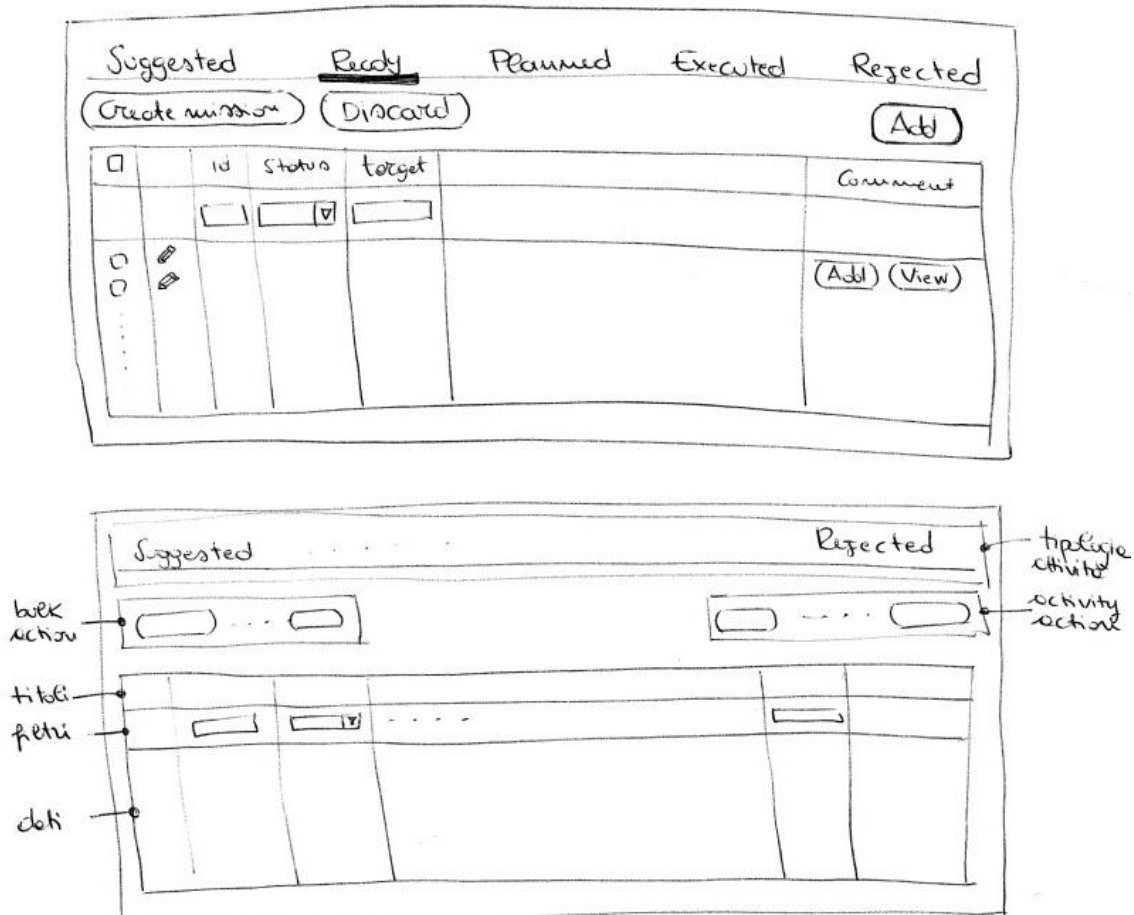


Figure 29 - Task list widget mock-up

This widget shows the task list. Tasks are grouped according to the following states:

- Suggested: a task suggested by automatic system but not yet analysed

- Ready: a task approved by user but not yet set and planned for mission
- Planned: tasks that are set in mission and planned depending to mission execution date
- Executed: completed tasks
- Rejected: tasks aborted by user

Typologies for data acquisition will be:

- UAV: ready, planned and executed
- UGV: ready, planned and executed
- Manual: executed

“Bulk action” (multirow concurrent tasks) buttons, fixed (row independent tasks) buttons, and different columns containing specific information will be shown according to selected task state.

Table contents will be filtered accordingly to other page widgets like map and date filter. Filtering rows depending on displayed columns will be available too.

Activity refinement page for detailed data analysis and modification will be available for any row.

Comment addition and comment list view will be available from any table row for any task.

Tree 3D Viewer

This component in Figure 30 shows the tree structure three-dimensionally letting user rotate, tilt and zoom for detailed analysis.

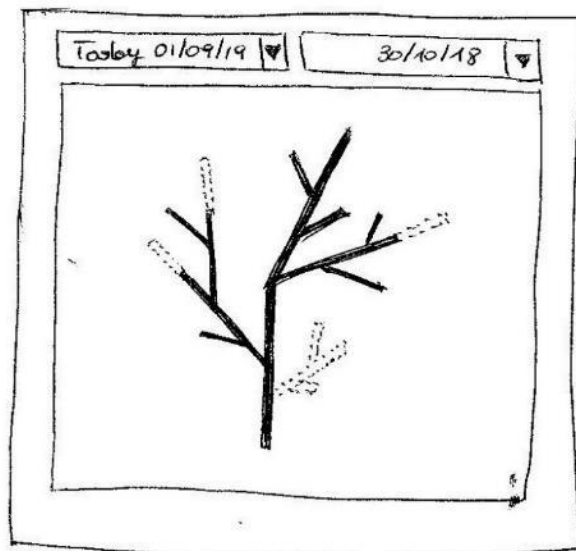


Figure 30 - Tree 3D viewer widget mock-up

Last acquired scan is shown when widget is rendered.

Widget lets user compare two scans by simultaneously overlying their images using different colors and transparencies. Data to be compared can be chosen by acquisition date drop down menus.

Tree pruning selector

This widget in Figure 31 shows a stylized tree structure and lets user select branches.

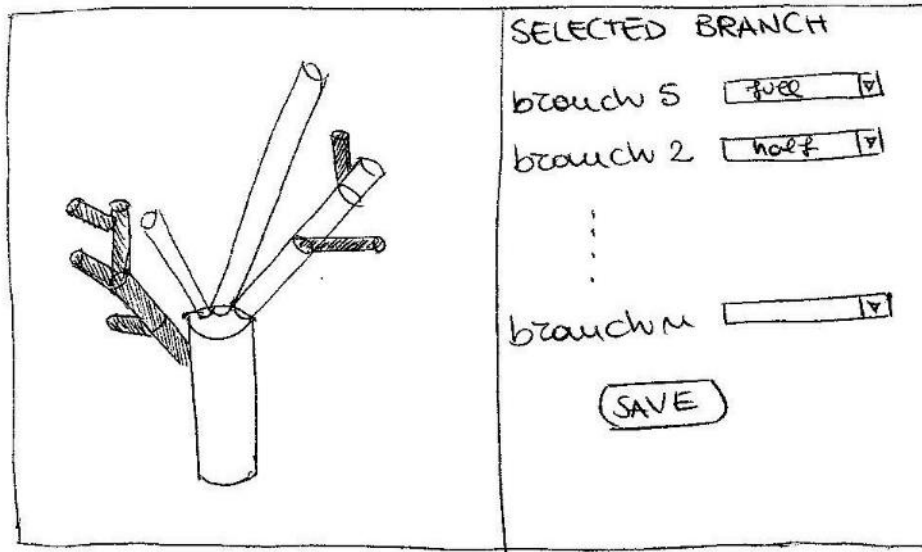


Figure 31 - Tree pruning selector widget mock-up

Tree structure view can be rolled, tilted and zoomed. Due to their status, branches will be drawn with different colours.

When branches are selected, further details for operations to come and their mode of execution (i.e.: full branch, half branch pruning, etc.) can be specified. When a branch is selected for cutting, all descendant branches will be shown transparently to indicate they will suffer the effect.

NB: in the PANTHEON context, the system marks the branches to be cut, therefore the tree model is shown to the user with branches already marked. Then, the user can verify and approve the system suggestion or edit the pruning configuration interactively.

Mission Summary

This component in Figure 32 shows a highlighted monthly calendar when missions are planned.

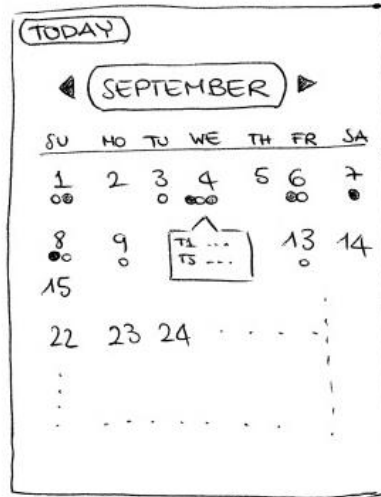


Figure 32 - Mission summary widget mock-up

A different colour indicator represents different task typologies.

Missions can include specific task execution (i.e.: irrigation, pruning, herbicide, fertilize) or data acquisition ones (UAV and UGV).

Task Summary

The component in Figure 33 shows task number. Tasks are grouped by typology and status (suggested, ready, planned).

TASK	WEEK		
	suggested	ready	mission
Irrigation	0	0	1
Fertilize	2	2	0
Pruning	0	0	1
Pesticide	15	10	0
Herbicide	1	1	1

Figure 33 - Task summary widget mock-up

A time period choice among “week”, “15 days”, “month” for summary can be selected by drop down menu.

Weather Forecast Summary

The component in Figure 34 shows the weather forecast of the hazelnut field located in Caprarola.

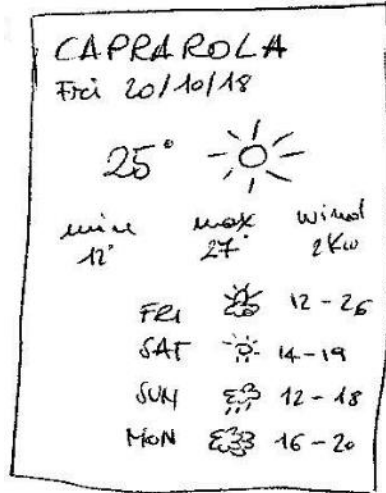


Figure 34 - Weather Forecast Summary widget mock-up

Present temperature and other parameters like maximum, minimum temperature, humidity and wind chill are displayed in this page. Maximum and minimum temperature forecast for future days is displayed in this page.

Notification Box

The Figure 35 shows notification boxes used in the Dashboard and on the application main page header.

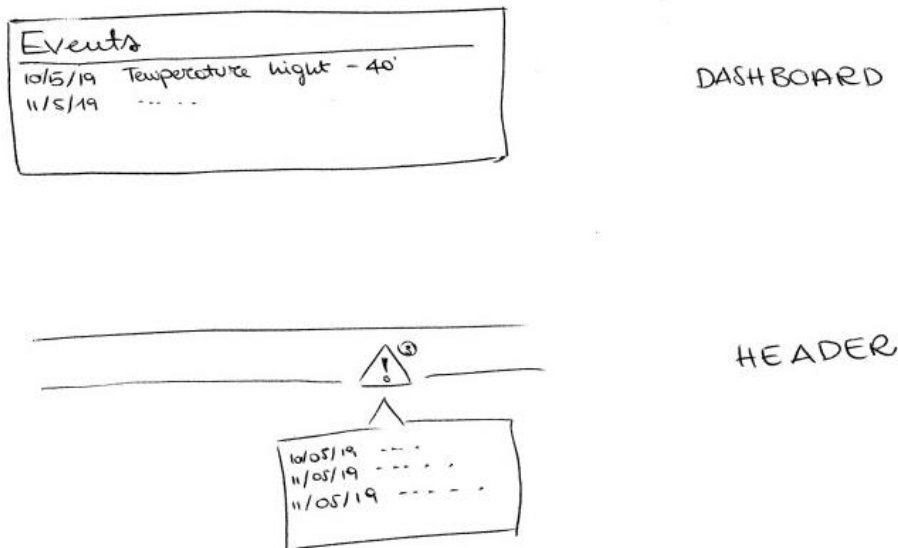


Figure 35 - Notification box widget mock-up

Relevant events retrieved from central system are displayed in this widget. This widget can be shown in extended or thick way depending on page context.

7.3.2 User Interface Page Mock-up

This paragraph shows application sections. Every page will be developed using the components described in the previous paragraph and others created for the specific section, if necessary.

Web App layout

Figure 36 shows the application main layout. Web application is based on a responsive design to better adapt to screen dimensions for any device that will use it.

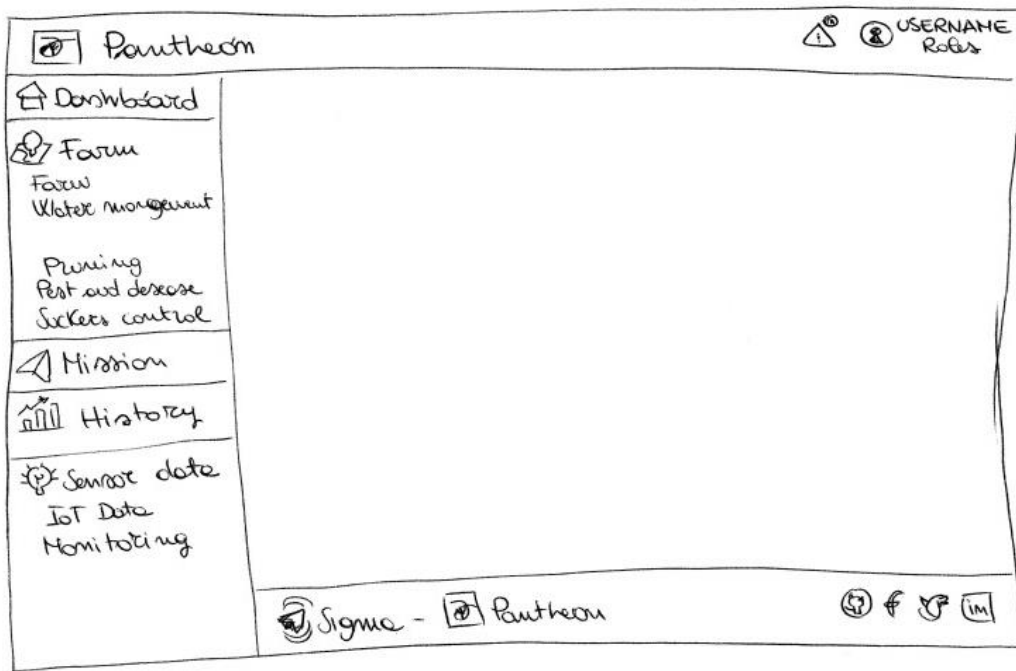


Figure 36 - Web App layout mock-up

Layout is divided in these areas:

- Header: user profile and relevant event notification (synthetic mode) widget are present in this area
- Left sidebar: navigation menus are displayed in this column. In some view typologies it can be resized or hidden
- Main area: requested page contents are displayed here
- Footer: social media icons and further information on involved firms are displayed in this section

Dashboard

The dashboard page, shown in Figure 37, is the first page the application displays after a user login has occurred.

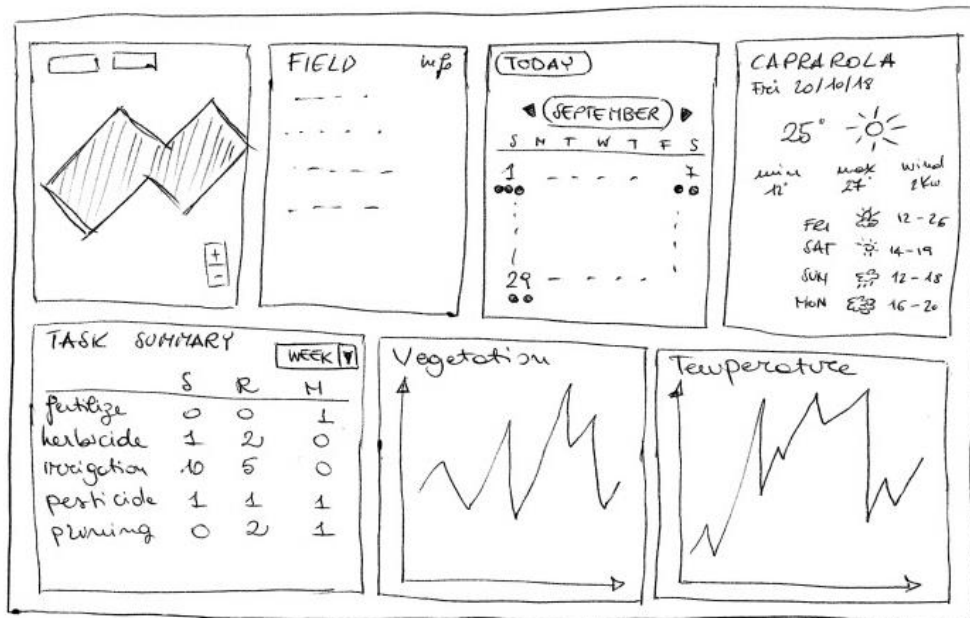


Figure 37 - Dashboard page mock-up

The summary or general info widget is shown to give the user information about all the operations that should be executed on field or to give user field main statistics.

The following widgets are displayed on page:

- Farm map, in interactive mode, showing the hazelnut field
- Info box displaying all field information and production forecasts
- Calendar summary displaying the monthly mission summary with typology detail
- Weather preview
- Activity summary
- Summary plotting temperature and tree life status

Farm

Following Figure 38 shows the farm page:

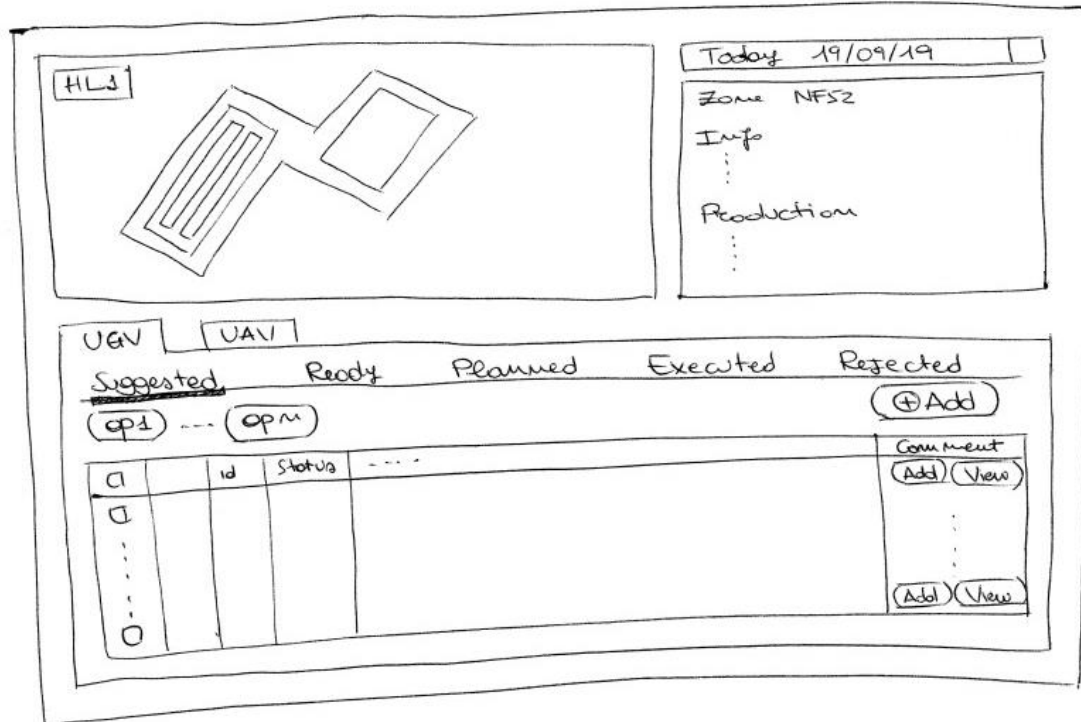


Figure 38 - Farm page mock-up

The following components are displayed in this page:

- Farm map, used in interactive mode
- Date selector, info box and task list information are updated selecting the date
- Info box: entire field data, production estimations and other statistical information are displayed at the beginning; later selected target data on map will be shown
- UAV and UGV details, showing acquisition data task lists are displayed in the panel below. Data are grouped by acquisition device typology

An input window letting user store effective production data can be enabled in the Info box, in the production data section. "Data Acquisition" typology data can be added from task list. Relative missions (UAV and UGV) can also be created. Already completed manual acquisition tasks can be added from task list too.

Farm - Agronomic Activities

In Figure 39 shows the agronomic activities generic page. This page is valid for pruning, sucker control, irrigation, Pest & Disease activities.

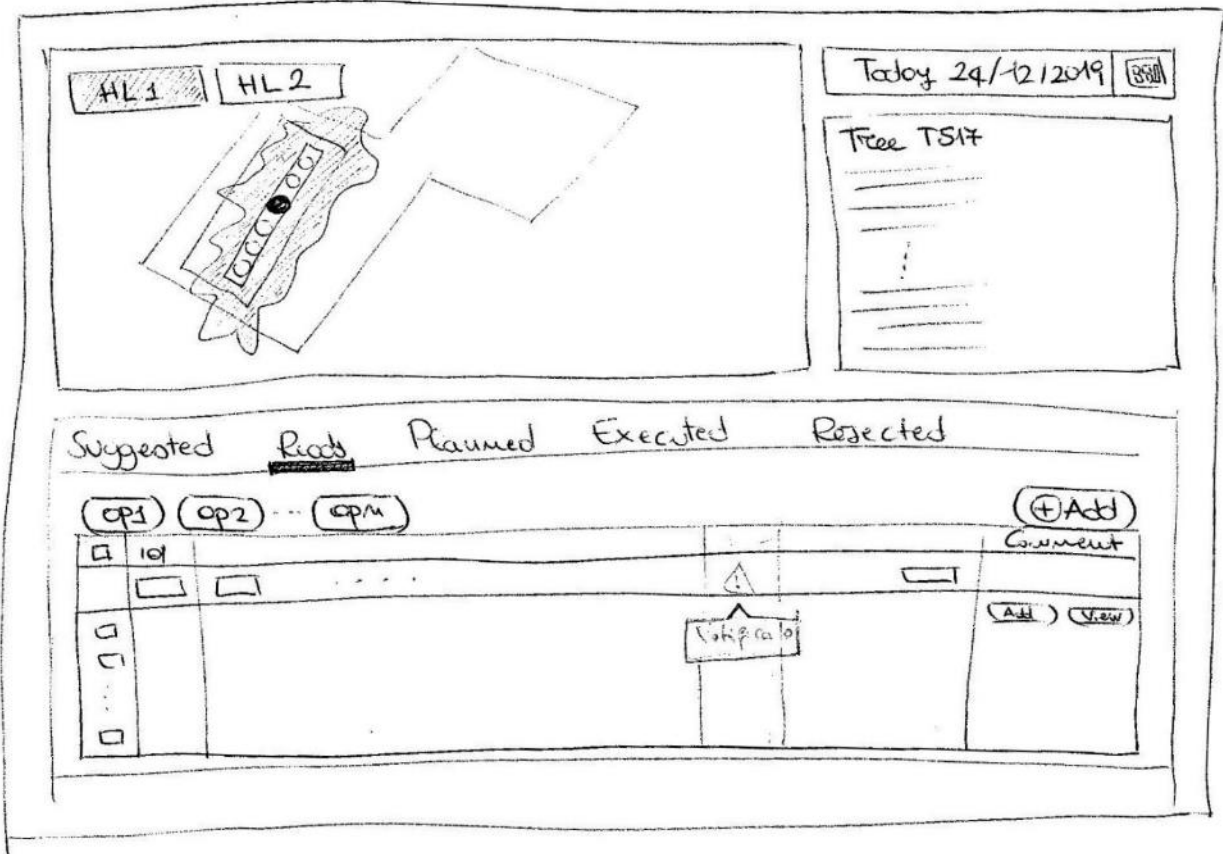


Figure 39 - Farm activity list page mock-up

The following components are displayed on page:

- Farm map, used in interactive mode
- Date selector, selecting a date, information in Info box and task list are updated
- Info box, where element details are specified
- Task list displaying only present web app section tasks

Every section has specific heatmaps that are available on map and that can be displayed or hidden.

Farm - Production

The page, shown in Figure 40, displays historical production data of the previous years and lets user add actual information about the harvest performed on current year grouped by different quality details (weight, waste, rotten, damaged, etc.).

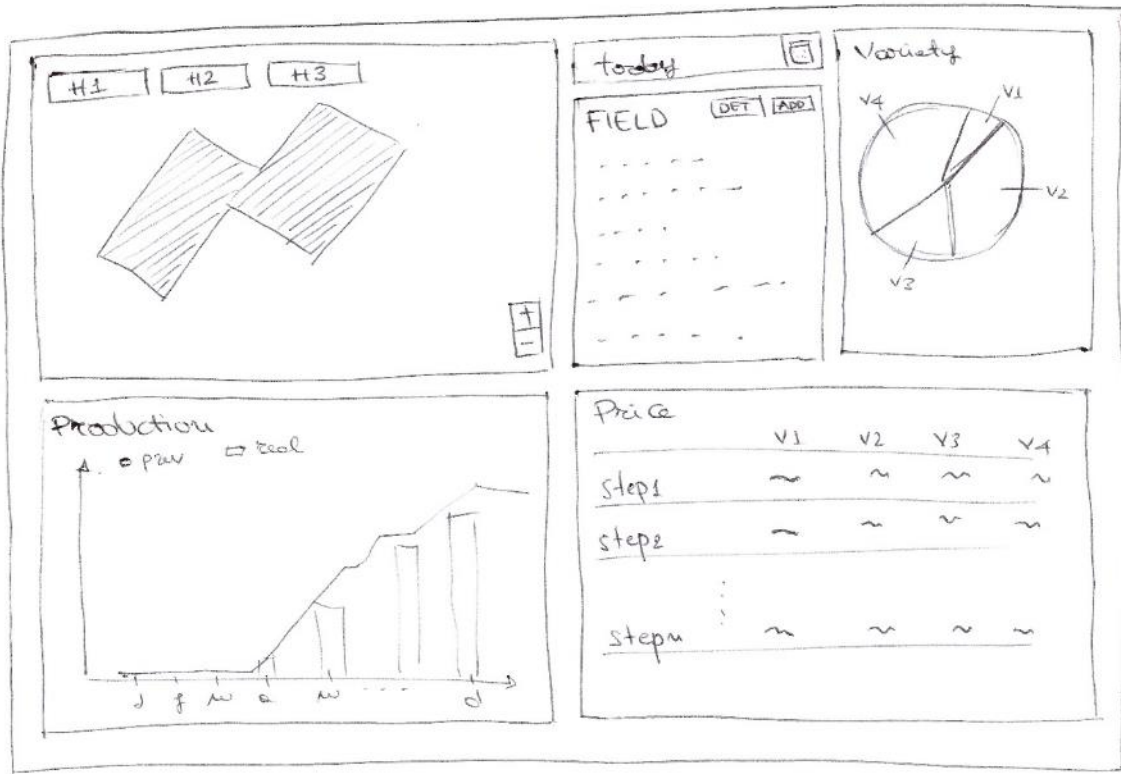


Figure 40 - Farm production page mock-up

The page displays the calculated price as function of input data and price range tables. Damages due to animals or wind can be inserted too.

Mission

The following Figure 41 shows the agronomic operations mission page.

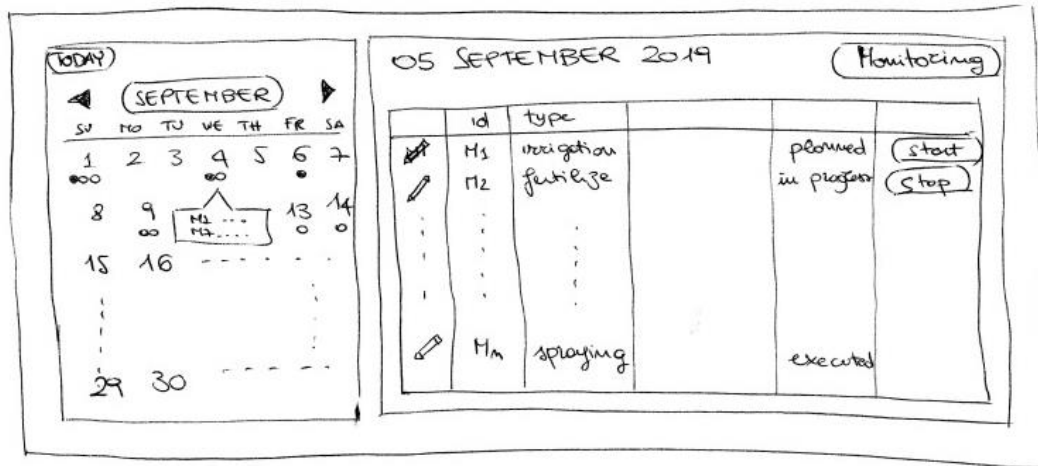


Figure 41 - Mission page mock-up

The following components are displayed on page:

- Summary calendar highlighting dates when missions are planned using tooltips and colored dots.
- Link to the task monitor page, it is active only when “in progress” missions are available
- Table containing planned mission list for the selected month on calendar; depending on task status, different actions will be available (i.e.: start and stop)

“Start mission” and “Stop mission” buttons will activate a dialog box where necessary information to link to mission will be inserted.

History

The following Figure 42 shows the historical data management page.

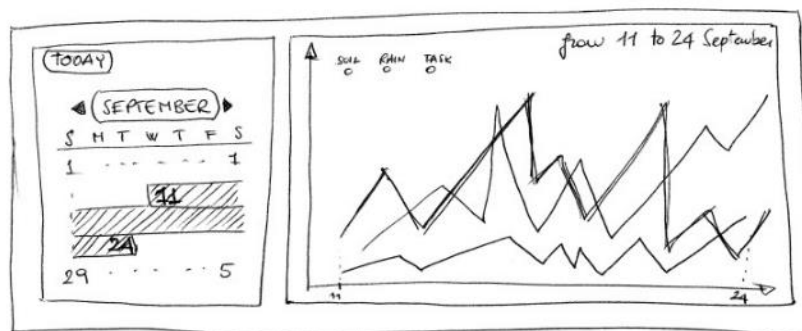


Figure 42 - History page mock-up

The following components are displayed on page:

- A calendar widget to select a time interval

- A graph plotting different data (temperature, task, etc.) in selected time lapse

Data concerning the last 30 days are displayed when page opens.

IoT Data

The following Figure 43 shows the IoT sensor network data collected page.

72

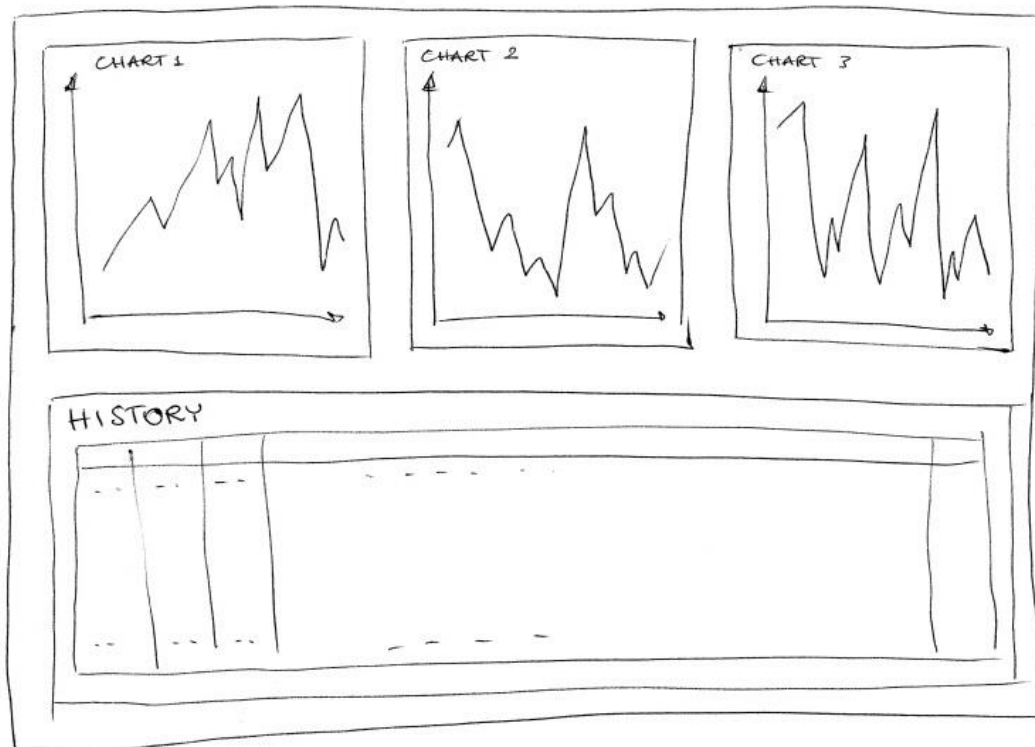


Figure 43 - IoT Data page mock-up

The page displays the following two component types:

- The first section depicts real-time graphs for data acquired from static sensors
- The second section (history) shows historical data acquired by such sensors

Monitoring

The following Figure 44 shows the farm mission monitoring page.

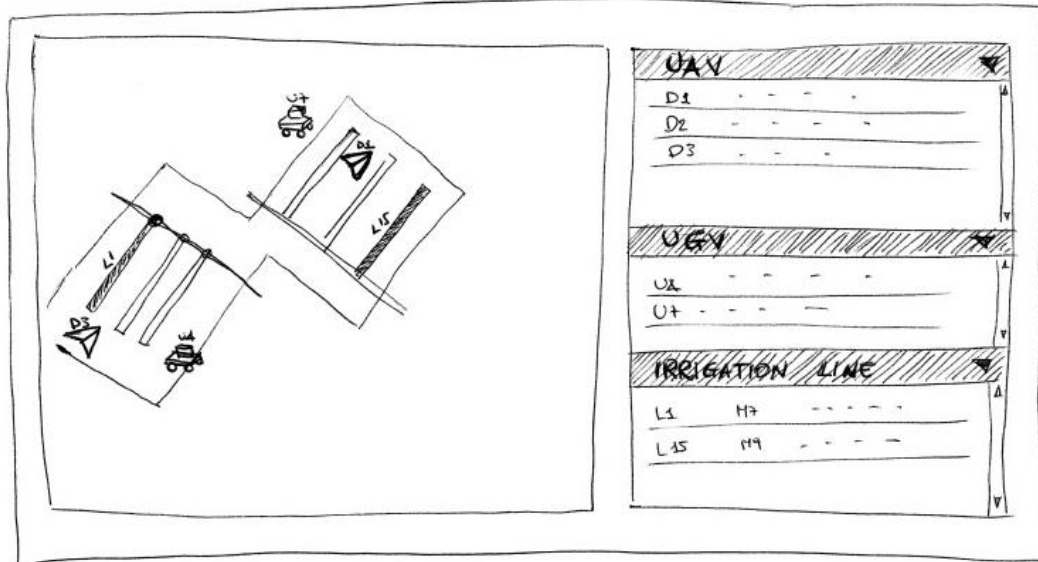


Figure 44 - Monitoring page mock-up

The following components are displayed on page:

- Farm map, in non-interactive mode, displaying mission target and robotic platforms position
- A retractable menu displaying devices executing missions, grouped by platform type. Further specific details, e.g., position, attitude, pesticide level, are also shown and updated in real-time.

Task Detail

The following Figure 45 shows the task detail page.

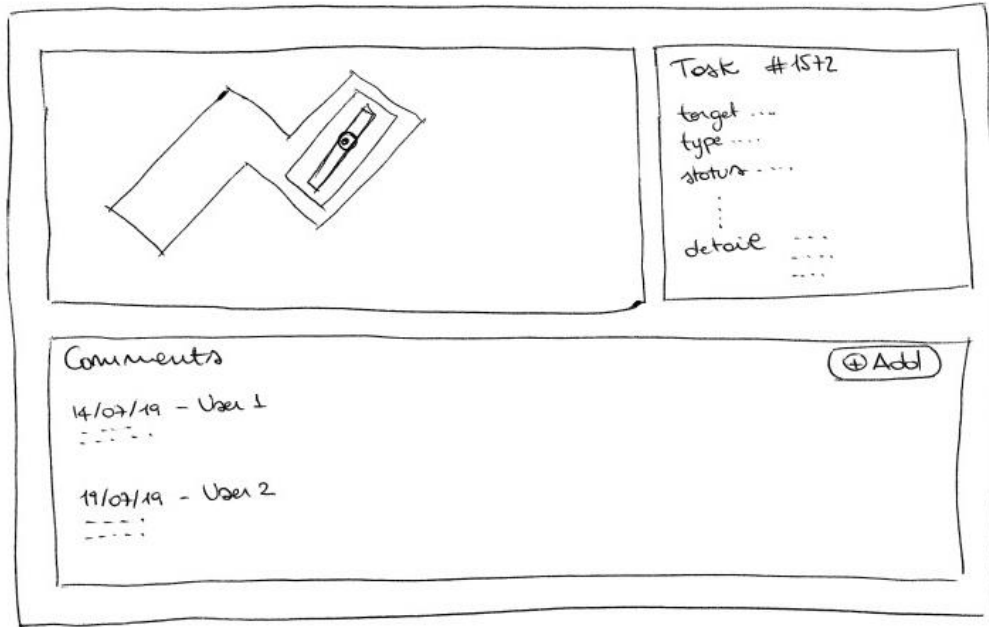


Figure 45 - Task detail page mock-up

The following components are displayed on page:

- Farm map, used in non-interactive mode with selected target
- Info box, where task details (like type, status, approval date, mission date, etc.) are specified
- A panel about comments where all task comments are displayed. New comments can be added

When task is pruning type, the widget “Tree selector” (with modification feature disabled) will also be shown to highlight which branches have been selected.

Farm Element Detail

Detail pages, for field elements like sector, row or tree, are shown in Figure 46 and Figure 47.

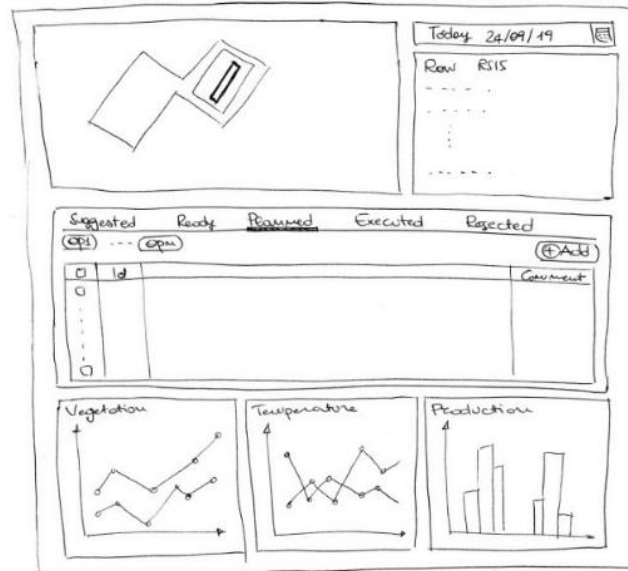


Figure 46 - Element detail page (sector or row)

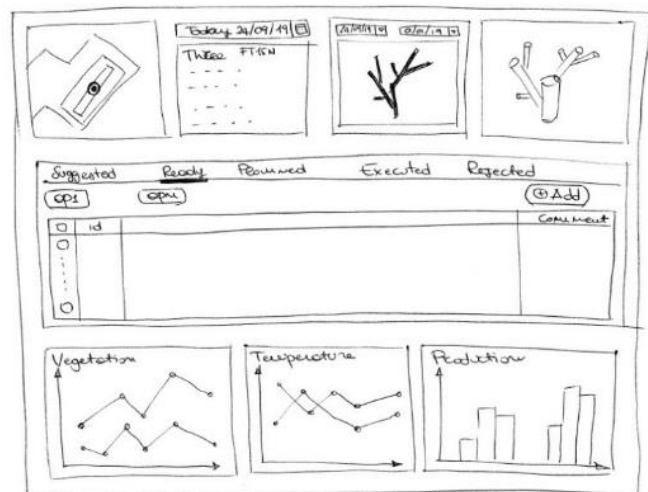


Figure 47 - Element detail page (tree)

The following components are displayed on page:

- Farm map, used in non-interactive mode with selected target
- Date selector, Info box information and graphs are updated by selecting the date
- Info box, where element details are specified
- Task list representing all selected target tasks
- Graphs representing some parameters details, like vegetation, production and temperature

Three-dimension structure display will also be available for tree details.

7.3.3 User Interface Dialog Mock-up

Add Task

When “Farm” map is used in non-interactive mode, the selected target element component is displayed in this dialog, shown in Figure 48. The component is useful to highlight what will be the activity target.

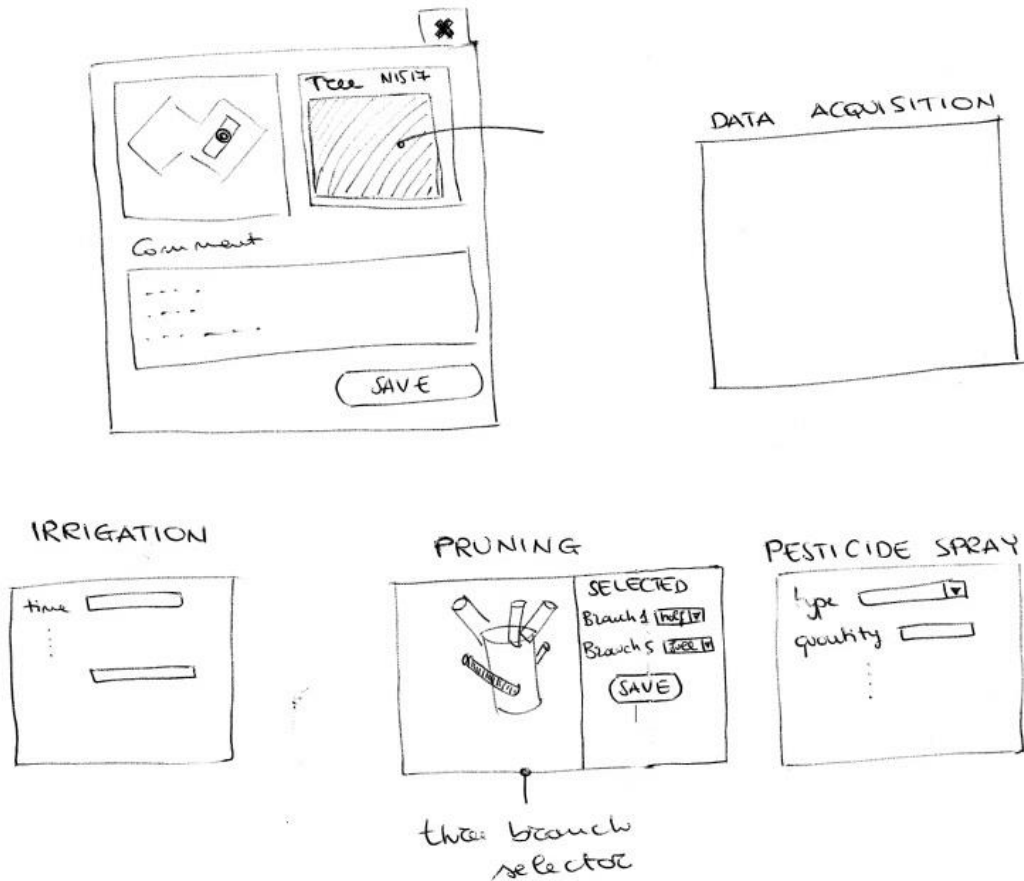


Figure 48 - Add Task dialog mock-up

Task typology will come from site section and can be:

- Irrigation
- Pruning
- Fertilize
- Pesticide spraying
- UAV data acquisition
- UGV data acquisition

Any activity typology shows a different box that will let the user specify details to send to devices or to human resources that will execute the activity:

- Irrigation: water quantity/irrigation length, etc.
- Pruning: pruning branch selection and cut type by the “Tree pruning selector” component

- Fertilize: fertilizer type, quantity, etc.
- Pesticide spraying: pesticide type, quantity, etc.
- UAV/UGV data acquisition: work area, waypoints, etc.

Create Mission

Dialog box, shown in Figure 49, is opened by bulk button in the task list ready tab, after selecting various tasks.

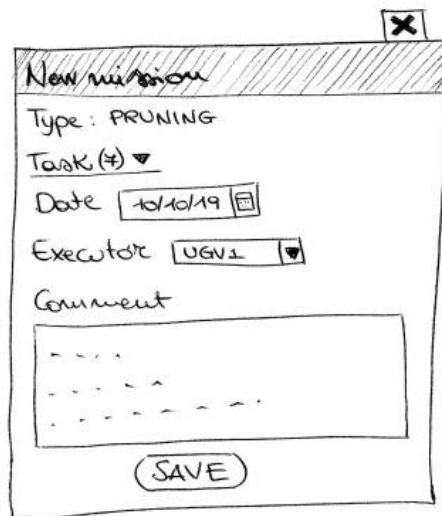


Figure 49 - New Mission dialog mock-up

Task type (irrigation/fertilize/pruning/pesticide/herbicide/data acquisition) comes automatically from web app section where mission is being created. Planned date should be inserted to plan mission. Device that will execute the mission can be selected. Device can be modified or inserted even later.

Start Mission

This dialog, shown in Figure 50, recaps Mission ID Code and Typology.

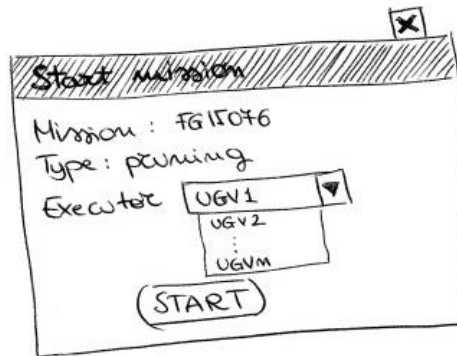


Figure 50 - Start Mission dialog mock-up

The dialog shows a dropdown menu to modify/insert mission to be executed by the robotic platforms, mandatory in this section, and lets the mission start. Note that, for the duration of the mission this will be shown as “in progress”.

Stop Mission

This dialog, shown in Figure 51, permits to close the mission manually and allows the user to insert comments.

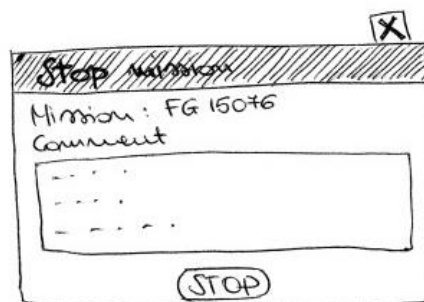


Figure 51 - Stop Mission dialog mock-up

7.3.4 User Interface Look and Feel

Web App Layout

Application layout is responsive and adapts itself to desktop (Figure 52), tablet (Figure 53) and smartphone (Figure 54) devices screen dimension.

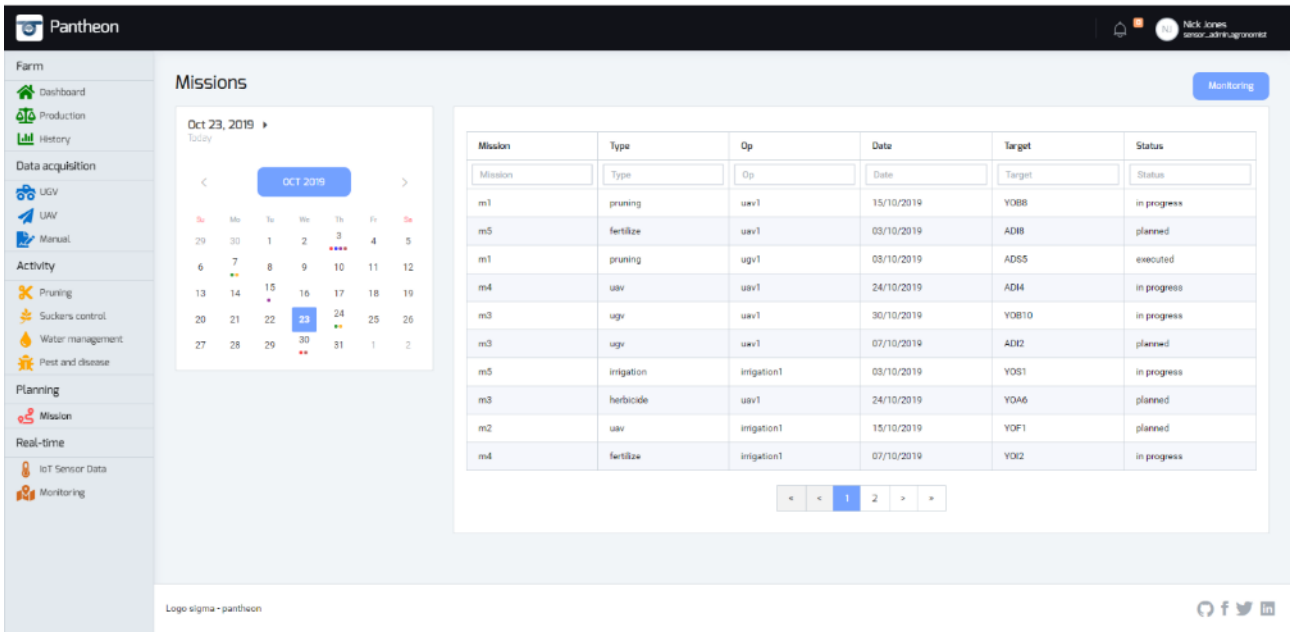


Figure 52 - Web App, desktop layout screenshot

When device screen dimensions are narrow, side menu is compressed and shows only icons or is hidden and becomes accessible on by specific icon, to better view quality.

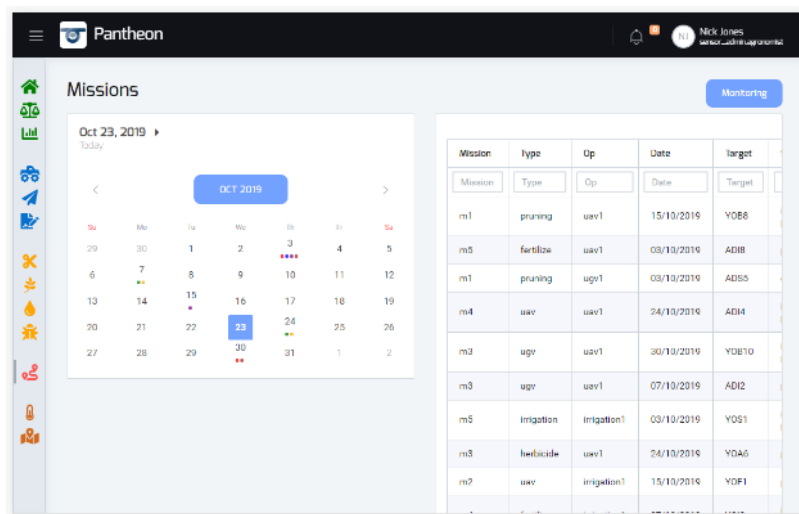


Figure 53 - Web App, tablet layout screenshot

Similarly, page widget boxes can occupy a different page space percentage standing side by side or one under the other.



Figure 54 - Web App, smartphone layout screenshot

Navigation menu shows icons with different colors for any group, so that they can be more easily identified even in compressed menu mode.

Dashboard

Dashboard, shown in Figure 55, displays data belonging to all application sections and groups them in synthetic way.

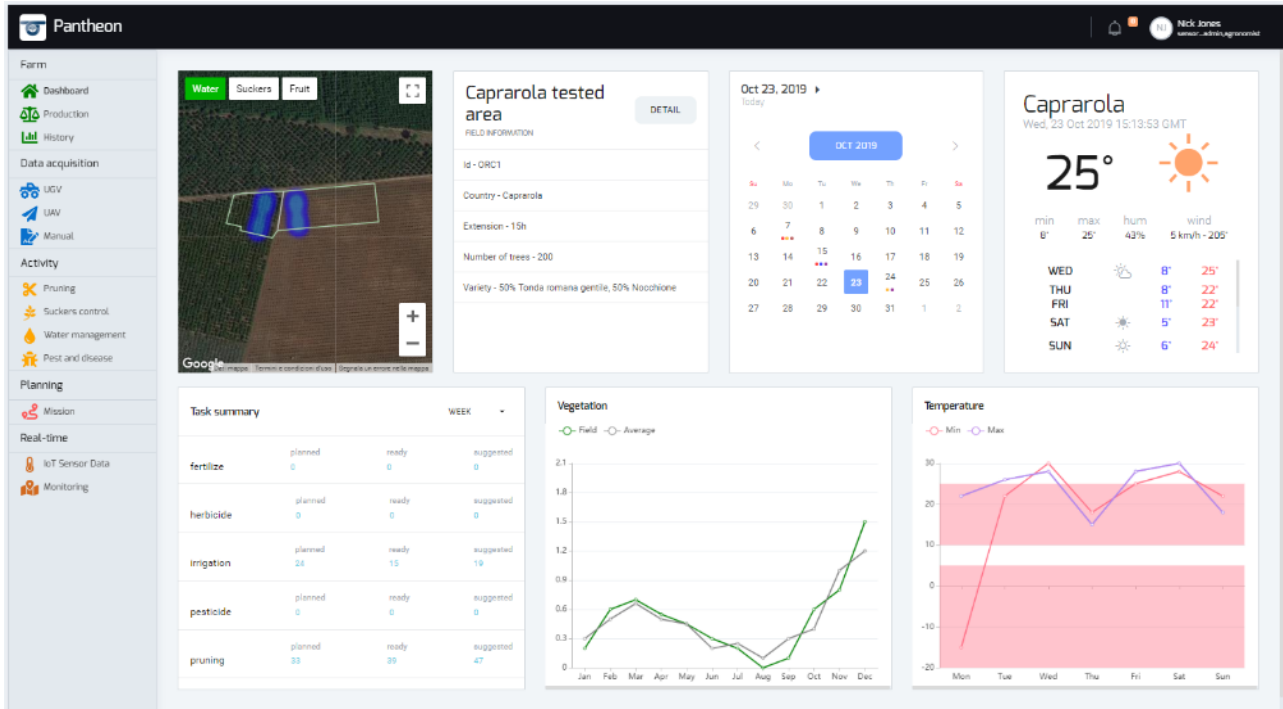


Figure 55 - Dashboard page screenshot

“Farm map” component is set in interactive mode up, so that user can select a field element and view its main information in the infobox. In this prototype, map widget should use Google Maps features. Detailed and historical data can be viewed by clicking on the “detail” button.

“Task summary” and “Mission summary” widgets summarizing what tasks and missions are present in the system and in what state are used.

Present and next days forecast weather conditions can be viewed through the “Weather preview” widget. The component, in this prototype, retrieves weather data by a third part service as “Yahoo weather”. Graphs on temperature and vegetative status for the entire field can be also be viewed.

Production

The page, shown in Figure 56, displays effective and estimated production data.

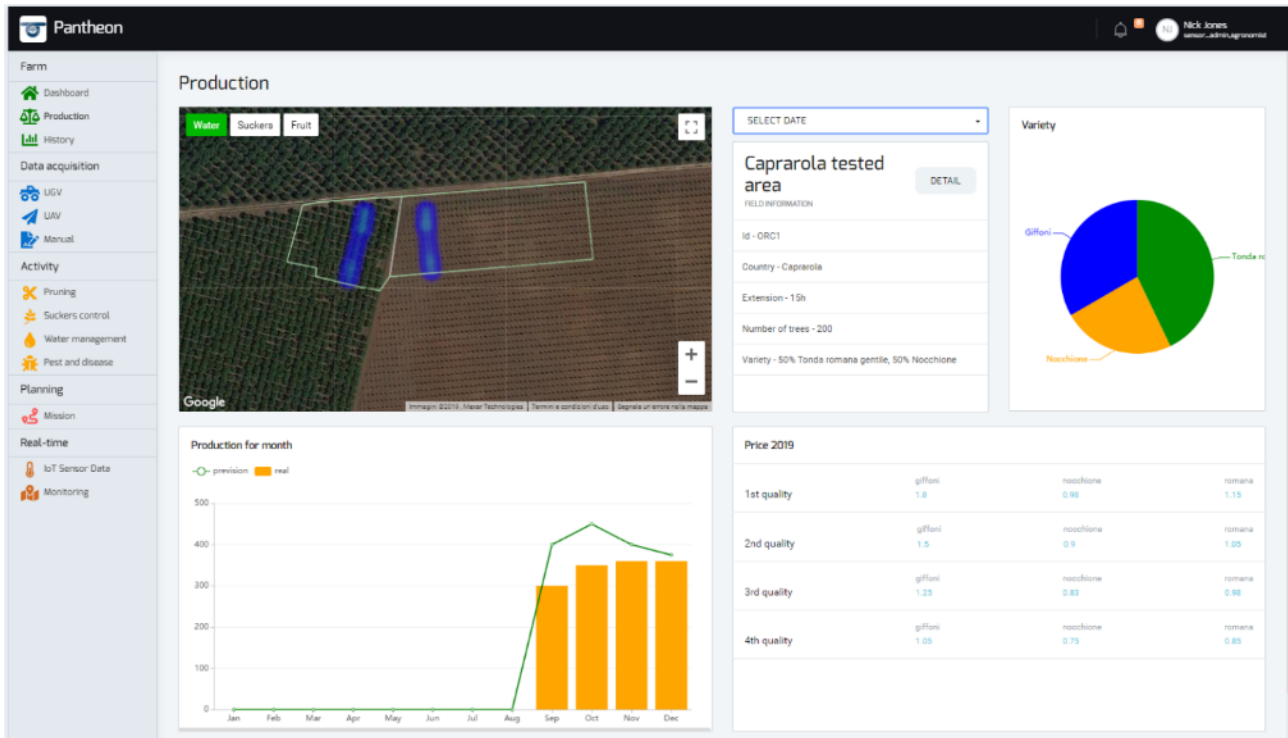


Figure 56 - Production page screenshot

Other widget data can be filtered through “Farm” map and “Date” selector components. They can limit data to a specific time period and to a specific field element.

Production percentages for various infield hazelnut varieties are shown in a pie chart. Current year sell prices tables divided in varieties and qualities are available for users.

History

This page, shown in Figure 57, displays historical acquired data and compares with previous years data through graphs on prices, rain amounts, farm vegetative state and temperatures.

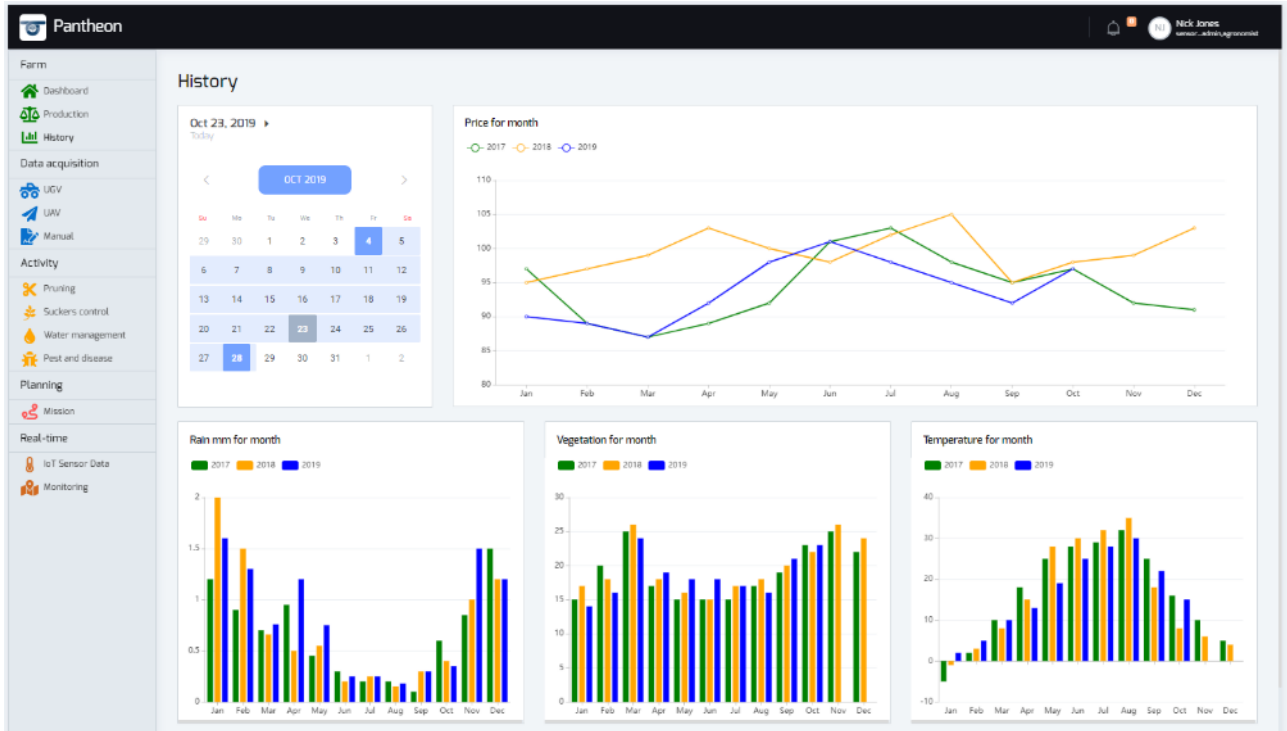


Figure 57 - History page screenshot

Page shows historical acquired data and compares with previous years data through graphs on prices, rain amounts, farm vegetative state and temperatures. Time interval can be filtered through the calendar widget to show required data on graphs.

UAV/UGV Data Acquisition

In all application sections about data acquisitions and tasks, “Farm” map and “Date” selector components are used as a filter to show data used in other page widgets. This section page structure, shown in Figure 58, is always the same. Only reference data and shown task status change.

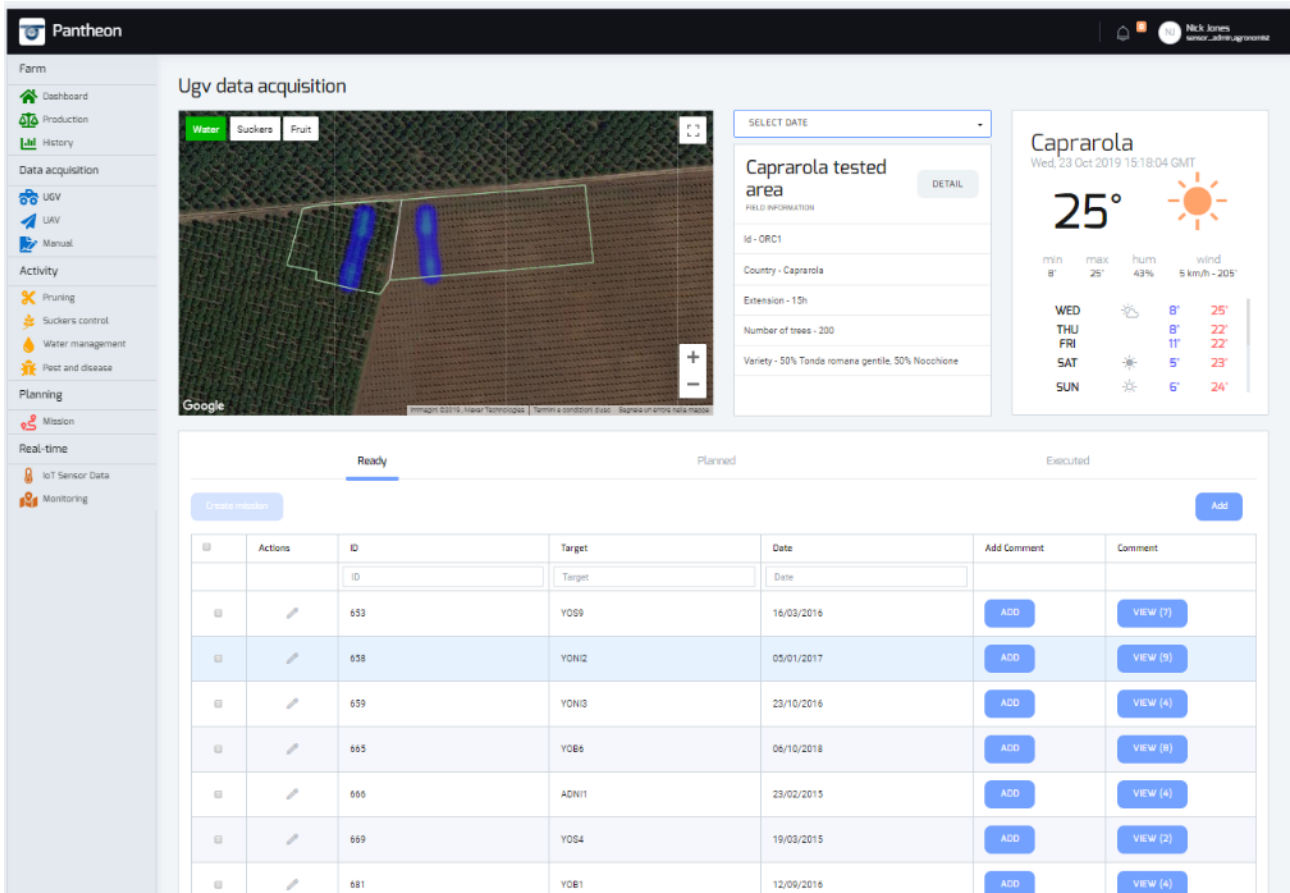


Figure 58 - UAV/UGV Data Acquisition page screenshot

For data acquisition performed by UAV and UGV, tasks can only be in the following status values:

- Ready: data acquisition task inserted by user; acquisition data mission will be created using these tasks
- Planned: tasks already set in a mission and planned
- Executed: already executed tasks

Manual Data Acquisition

This page, shown in Figure 59, displays data added manually by farm operators into the system.

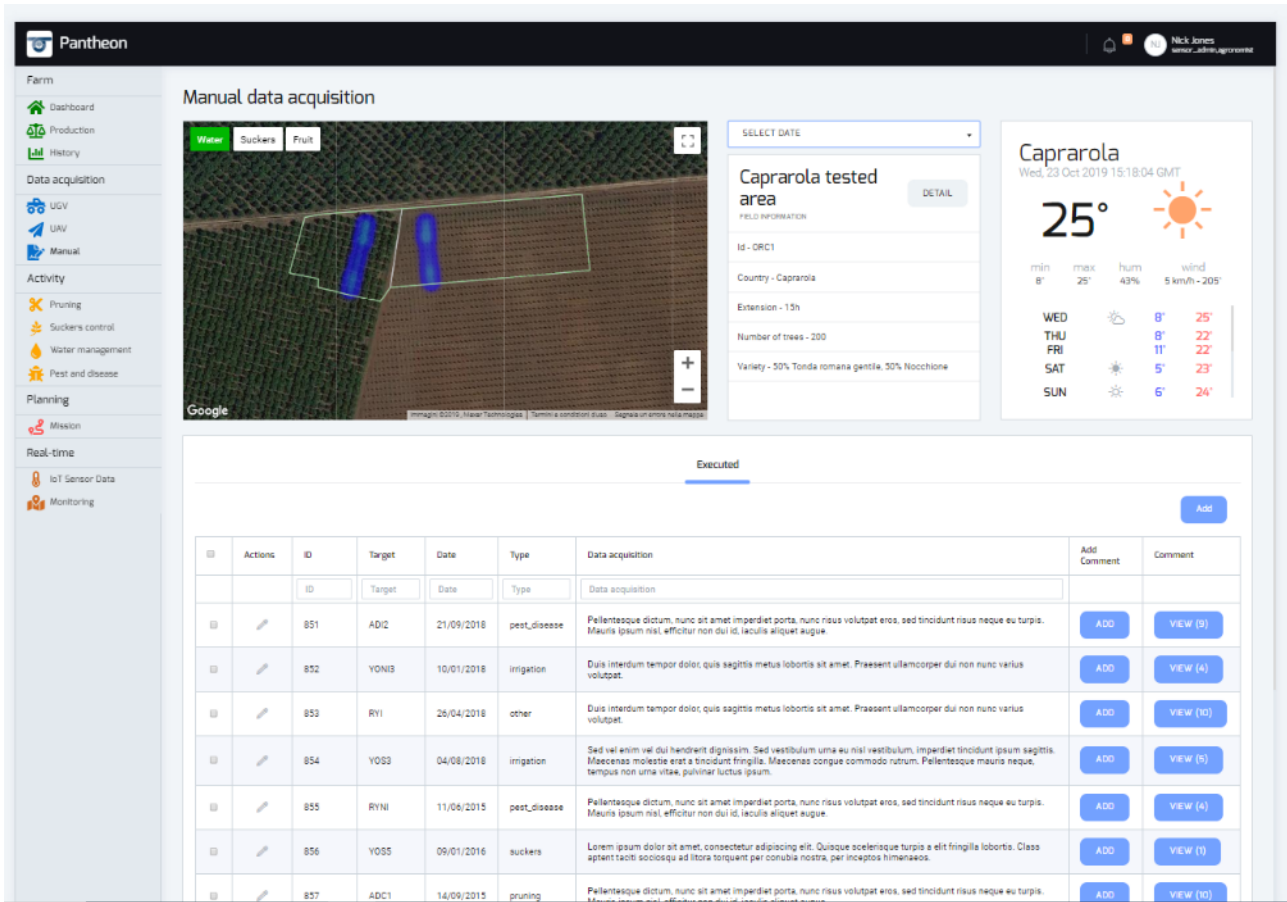


Figure 59 - Manual Acquisition page screenshot

A manual data acquisition activity represents an observation the agronomist carries out on the field. For this reason, this kind of activity can be only in "executed" state, corresponding to the event "observation inserted into the system"..

For this task, it is mandatory to insert the type of observation performed. A text field where to describe in details the issue should also be filled; photos input for further acquisition datum can also be available.

Pruning

This page, shown in Figure 60, displays data about hazelnut field pruning activities.

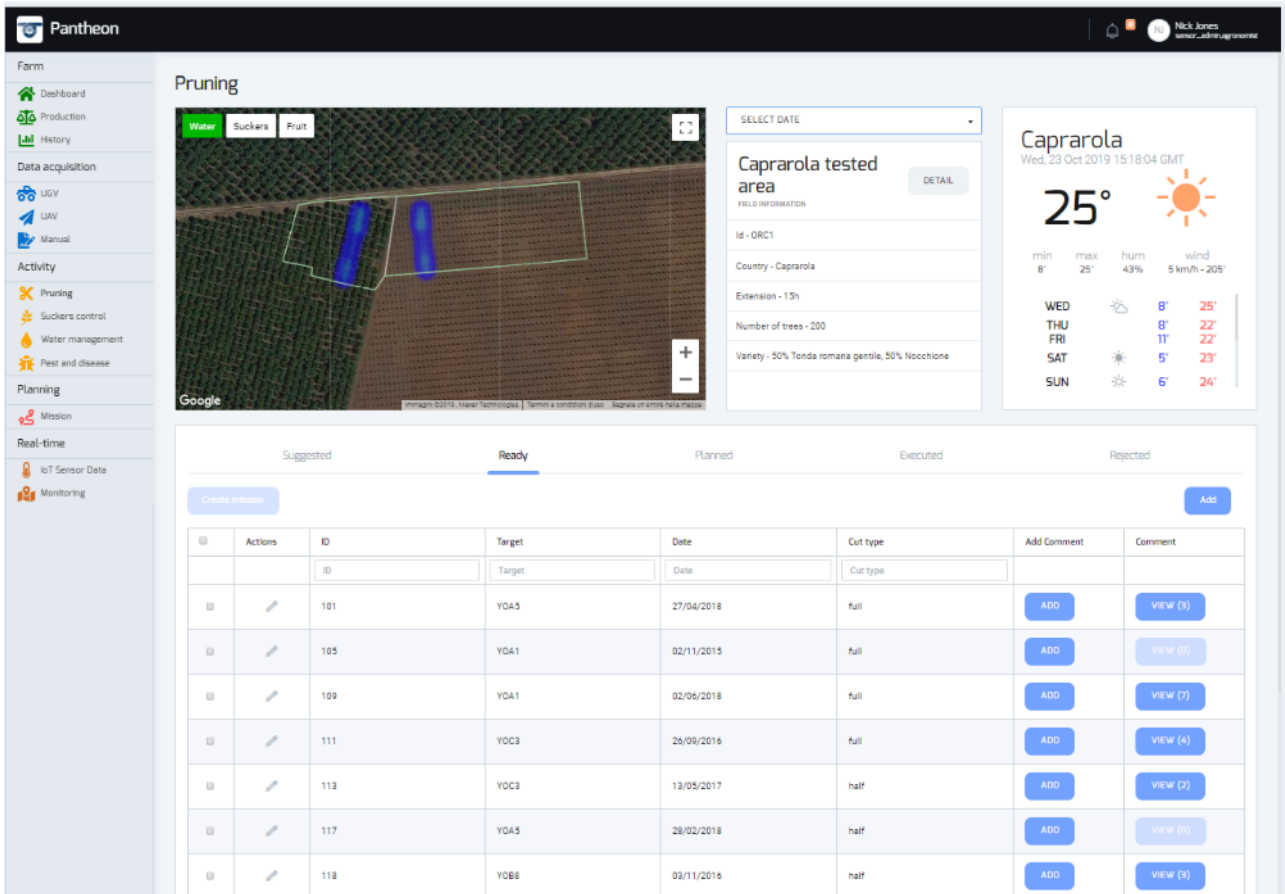


Figure 60 - Pruning page screenshot

All the tasks, shown in the activity section, can be in the following status values:

- **Suggested:** data analysis system suggested task after acquired sensor data elaboration. These tasks can be approved or rejected
- **Ready:** tasks that have already been analysed and approved by a system user
- **Planned:** tasks already set in a mission and planned
- **Executed:** already executed tasks
- **Rejected:** suggested tasks that were not considered valuable and so rejected by user

In pruning task, suggestions on cut type to execute will be available in addition to standard information. Branch to be cut (by the “tree pruning selector” widget) and cut type can be specified by the adding task dialog.

Sucker Control

This page, shown in Figure 61, displays data about hazelnut field sucker control activities.

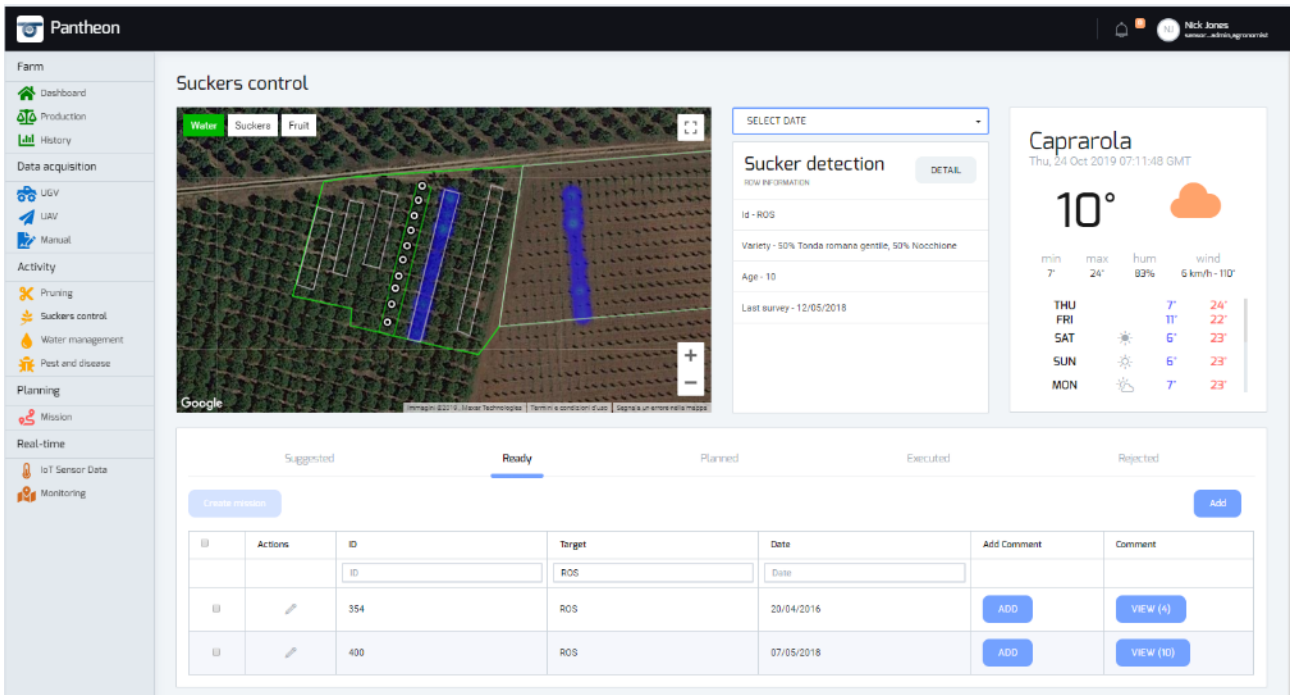
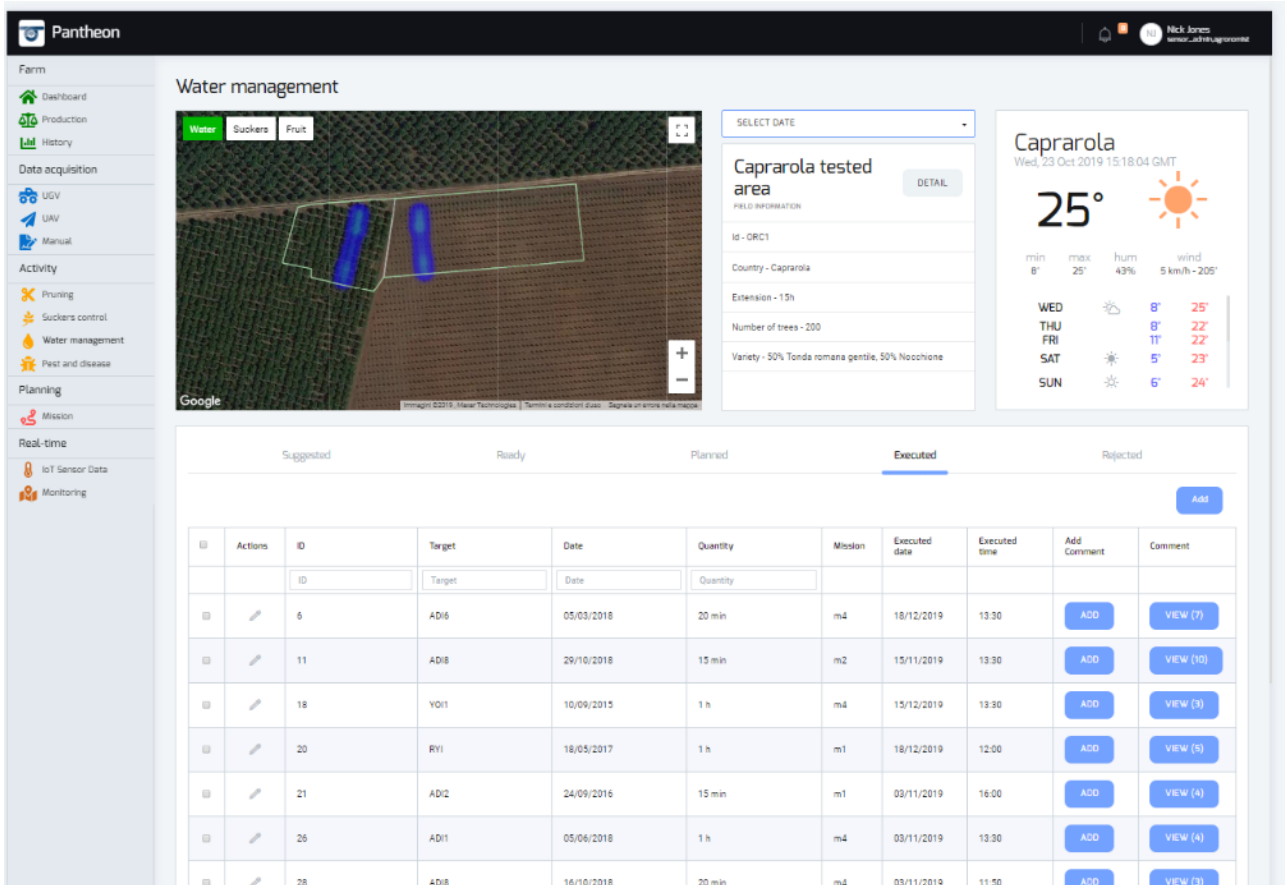


Figure 61 - Sucker Control page screenshot

In the sucker controls section tasks, product and quantity to be dispensed for sucker treatment can be specified in addition to standard information. Fields for product choice and dispensed quantity will be available in the add task dialog box.

Water Management

This page, shown in Figure 62, displays data about hazelnut field water management activities.



Water management										
SELECT DATE										
Caprarola tested area										
FIELD INFORMATION										
Id - ORC1										
Country - Caprarola										
Extension - 15h										
Number of trees - 200										
Variety - 50% Tonda romana gentile, 50% Nocchione										
Caprarola										
Wed, 23 Oct 2019 15:18:04 GMT										
25°										
min 8° max 25° hum 43% wind 5 km/h - 205°										
WED 8° 25°										
THU 8° 22°										
FRI 11° 22°										
SAT 5° 23°										
SUN 6° 24°										

Water management										
Suggested										
Ready										
Planned										
Executed										
Rejected										
Add										
ID	Actions	ID	Target	Date	Quantity	Mission	Executed date	Executed time	Add Comment	Comment
		ID	Target	Date	Quantity					
6		ADi6	ADi6	05/03/2018	20 min	m4	18/12/2019	13:30	ADD	VIEW (7)
11		ADi8	ADi8	29/10/2018	15 min	m2	15/11/2019	13:30	ADD	VIEW (10)
18		YDi1	YDi1	10/09/2015	1 h	m4	15/12/2019	13:30	ADD	VIEW (3)
20		RY1	RY1	18/05/2017	1 h	m1	18/12/2019	12:00	ADD	VIEW (5)
21		ADi2	ADi2	24/09/2016	15 min	m1	02/11/2019	16:00	ADD	VIEW (4)
26		ADi1	ADi1	05/06/2018	1 h	m4	03/11/2019	13:30	ADD	VIEW (4)
28		ADi8	ADi8	16/10/2018	20 min	m4	03/11/2019	11:50	ADD	VIEW (3)

Figure 62 - Water Management page screenshot

Irrigation facility activation time selection will be available in the irrigation task section in addition to standard information. This parameter will be specified in the add task dialog box.

Pest and Disease

This page, shown in Figure 63, displays data about hazelnut field pest and disease management activities.

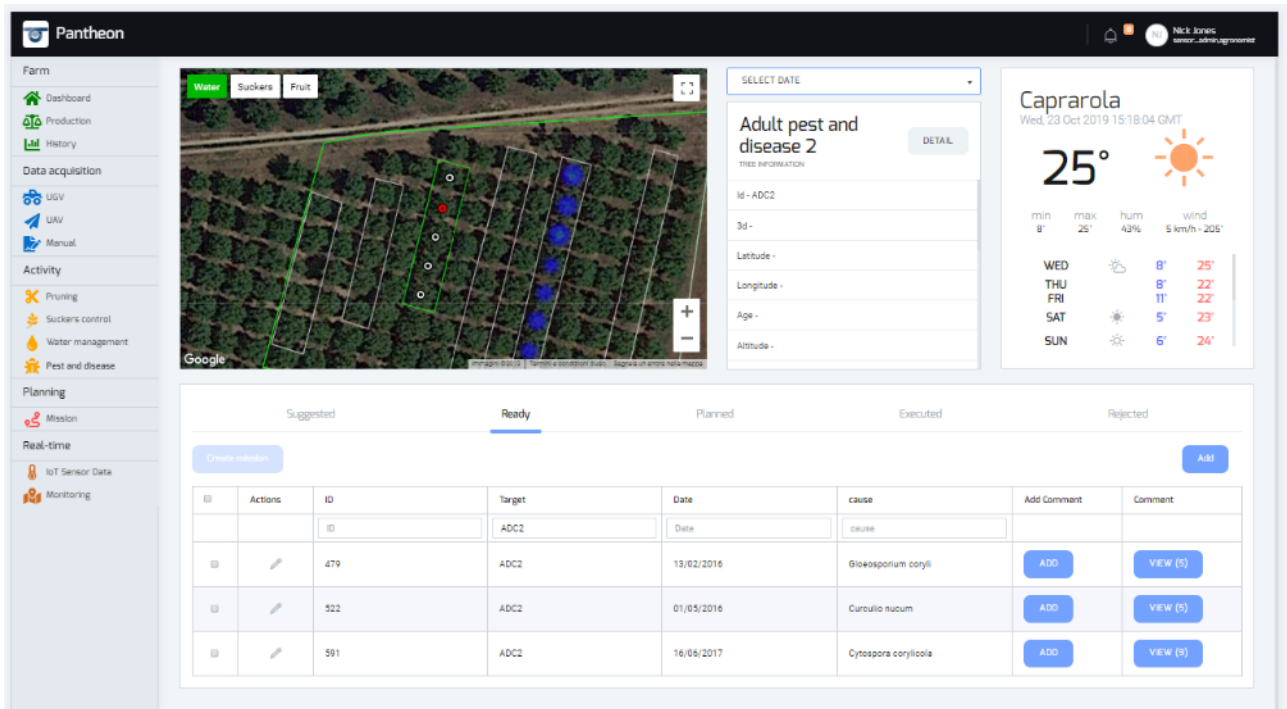


Figure 63 - Pest and Disease page screenshot

The pest species (or the issue detected), the sprayed product and quantity should be available in the pest and disease section in addition to standard information. Such information can also be specified in a new task.

Missions

This page, shown in Figure 64, displays data about hazelnut field missions planned.

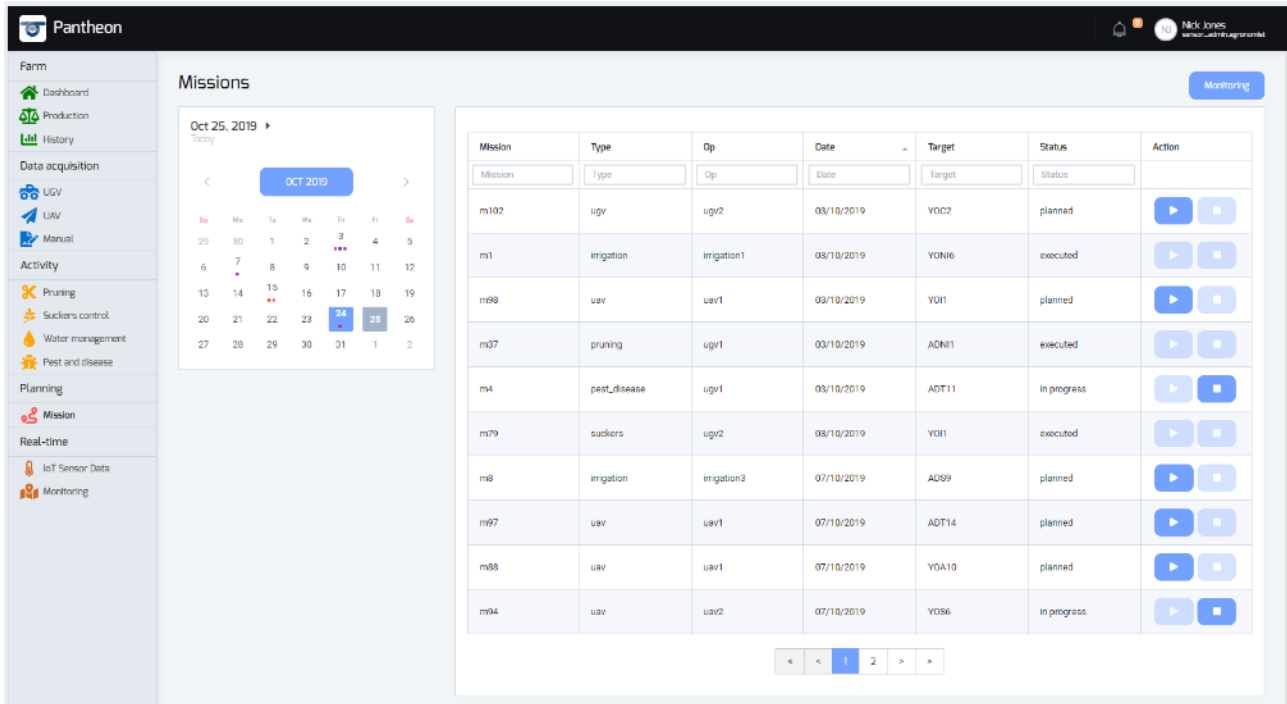


Figure 64 - Mission page screenshot

In the mission page, all mission stored in the system can be viewed both in synthetic way through a calendar and in detailed way through a list. Current month data is shown when the page is called, then time period can be changed by using the calendar. Data can also be filtered through table filters.

Further operations like “Start mission” and “Stop mission” will also be available according to mission status.

IoT Sensor Data

This page, shown in Figure 65, displays real-time data coming from fixed sensors installed on the hazelnut field.

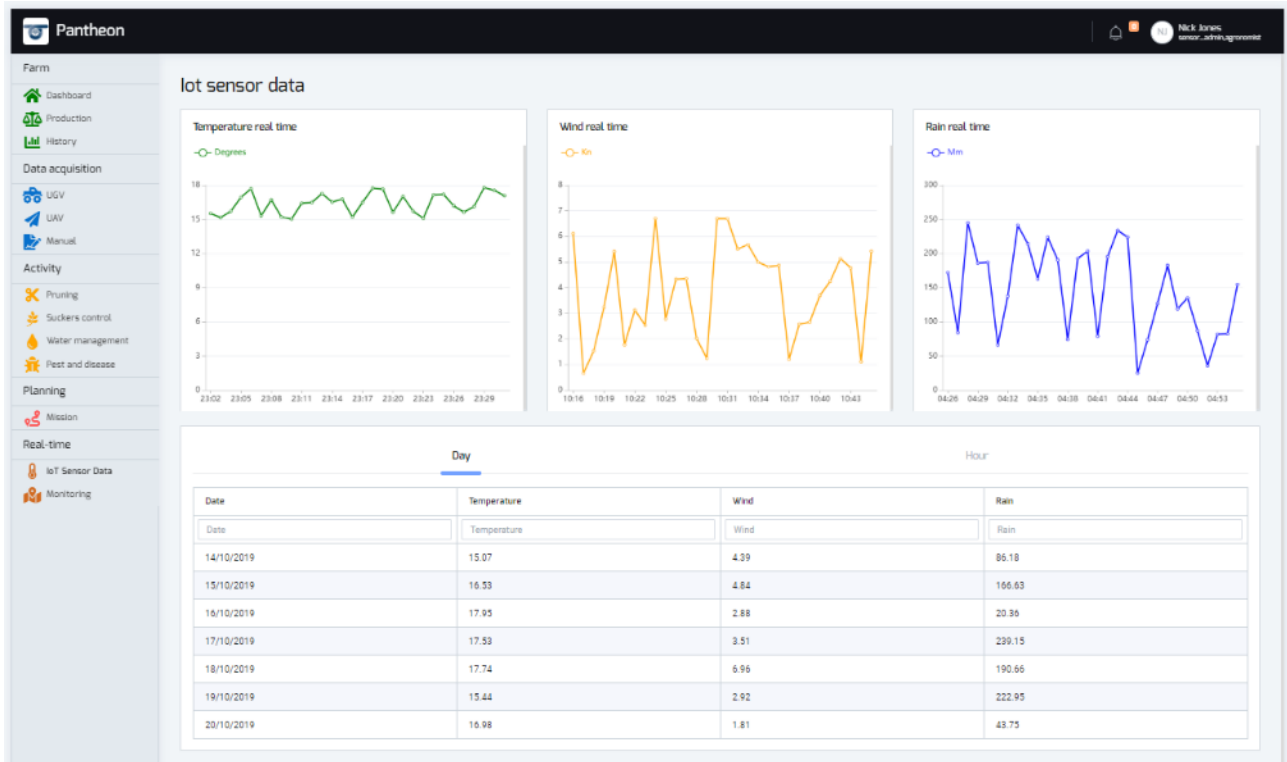


Figure 65 - IoT Sensor Data page screenshot

IoT sensors send data to central system that updates graphs in real time with a one-minute refresh rate. Data acquired in the last seven days grouped hourly or daily are also displayed.

Monitoring

The monitoring page, shown in Figure 66, displays data about autonomous platform operating on the field, providing map live and mounted sensors status.

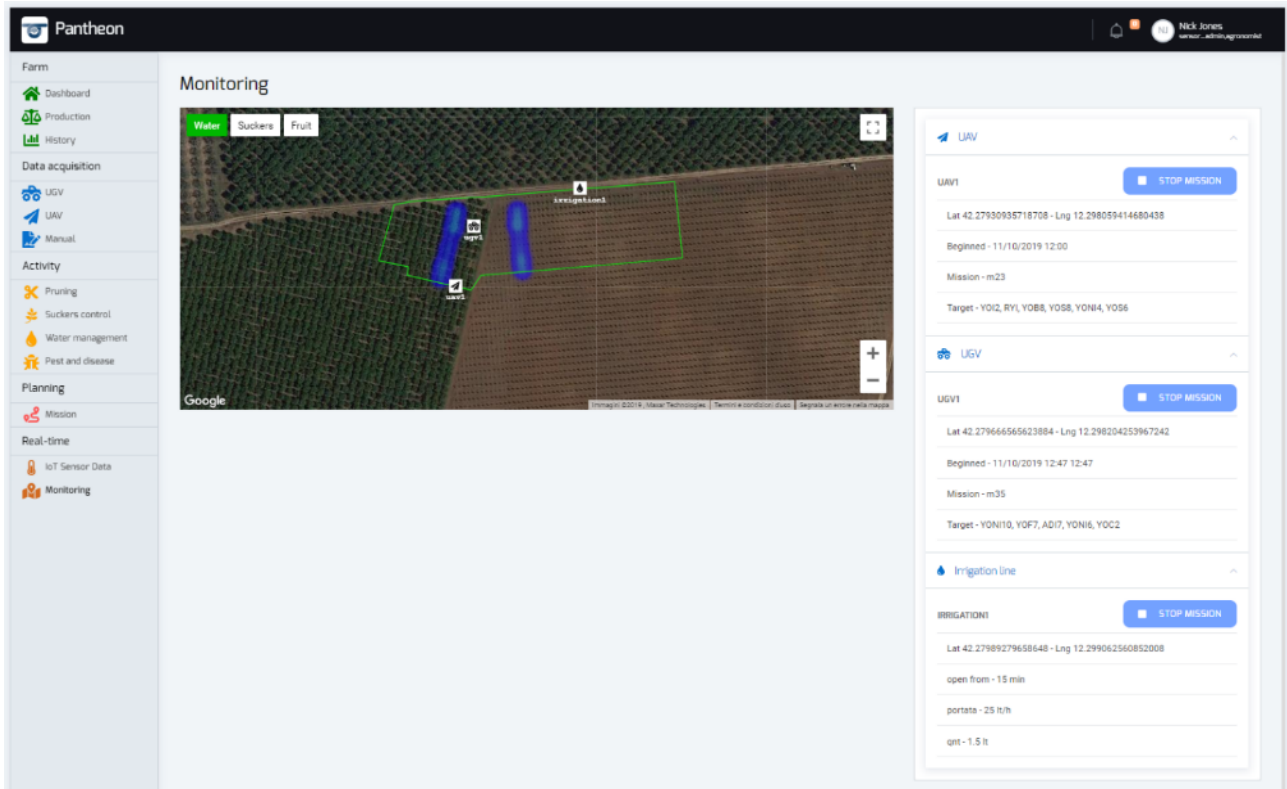


Figure 66 - Monitoring page screenshot

In information section data involving running mission, working device and execution typology are displayed. Used product will be also specified for sucker control and pest control missions. Static data like facility type and water maximum flow will also be shown together with real time data in irrigation operations.

Element Detail Data

On the element detail page, all the information regarding a "FieldElement" object is collected. That information can vary from the whole field to a single tree level. Following Figure 67 shows the tree element detail page.

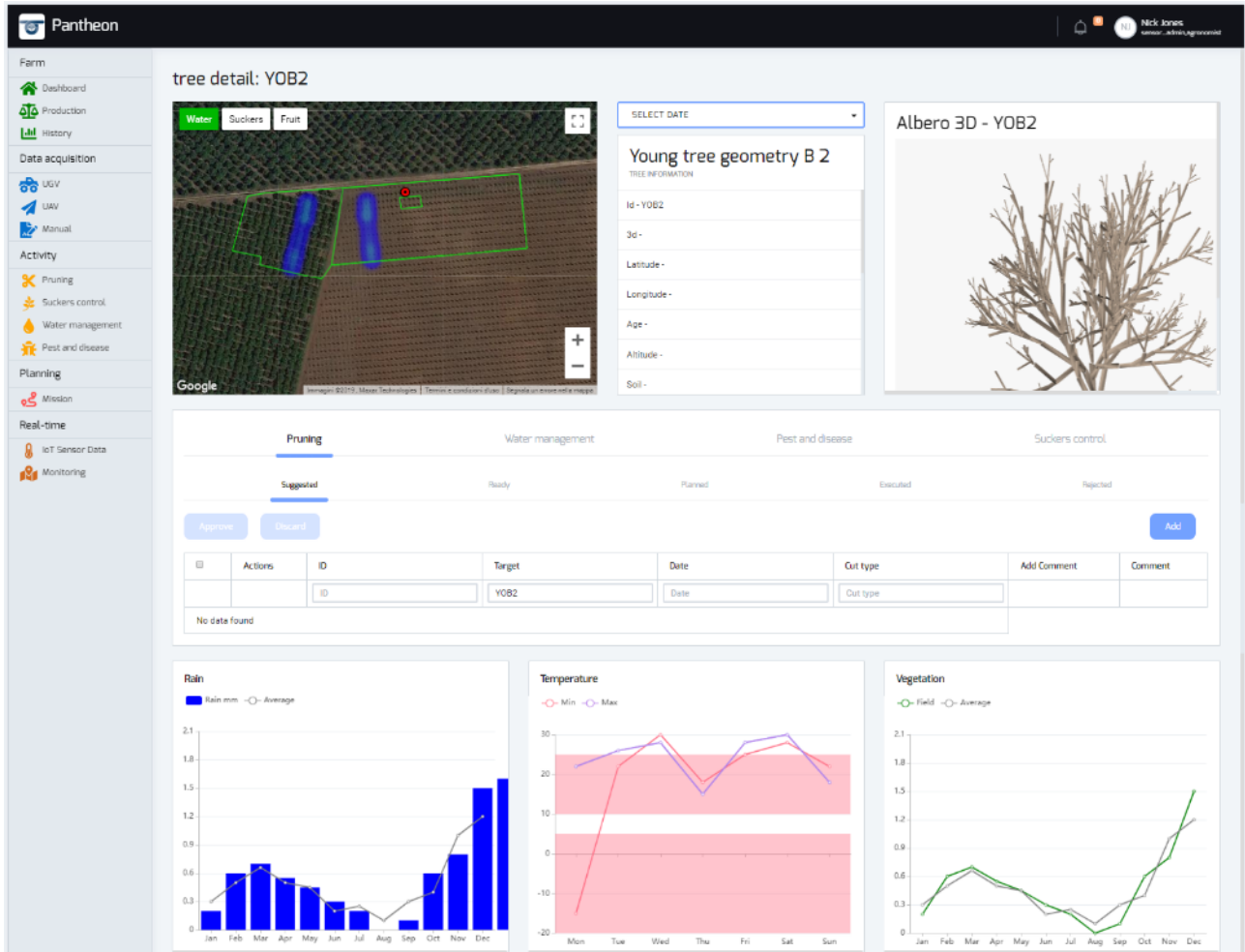


Figure 67 - Element (tree) detail page screenshot

Farm map is used in non-interactive way to show information about farm element positioning; in this context, the "Info box" component, obviously, does not provides any link to additional "detail" pages..

If single tree details have been selected three-dimensional scanned tree representation analysis widget will also be present. Previous scans can be viewed acting on the date choice menu. All the activities, concerning the selected element, are shown in the task table grouped by type. The page also shows the trend graphs of water stress index, temperatures and vegetative state about the analysed element.

When the analysed element is the single tree, a feature to compare two scans will be available. Requested scans will be selected by choosing among available dates. The page will then display any tree referring executed tasks during the selected time lapse and the overlying three-dimensional representation.

Task Detail

In the task detail page, shown in Figure 68, all data concerning a specific task will be available.

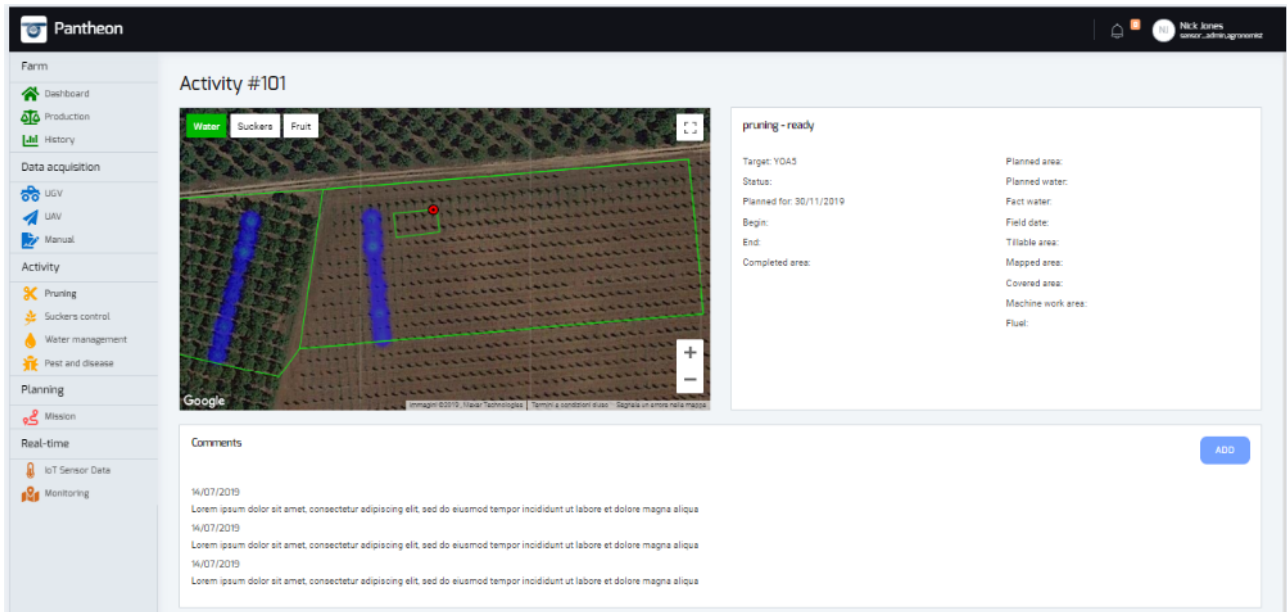


Figure 68 - Task Detail page screenshot

“Farm” map component will be in non-interactive mode to graphically highlight the task target and its position within the field. In detail section, task status is displayed. Related operations can be available depending on the status, like approving or rejecting a suggested operation.

Further detailed information is also specified, like if task has already been set in a mission, when it is planned, its possible start and end data, the fertilizing chosen product, the sprayed product, the product quantity or the amount of irrigation delivered. A comment panel where to see all the notes written by other users or to create new ones is displayed on page. This allows the user to trace information, task feedbacks and possible detected issues.

7.3.5 Back-end Design

Back-end feature

The back-end feature is the layer in charge of receiving requests from the front-end layer and, sending replies back to the same front-end after the request has been elaborated. Back-end must communicate with front-end, storage layer and notification level to accomplish the task. Back-end structure must be smart and thin in order to be able to answer to request with the least time delay possible.

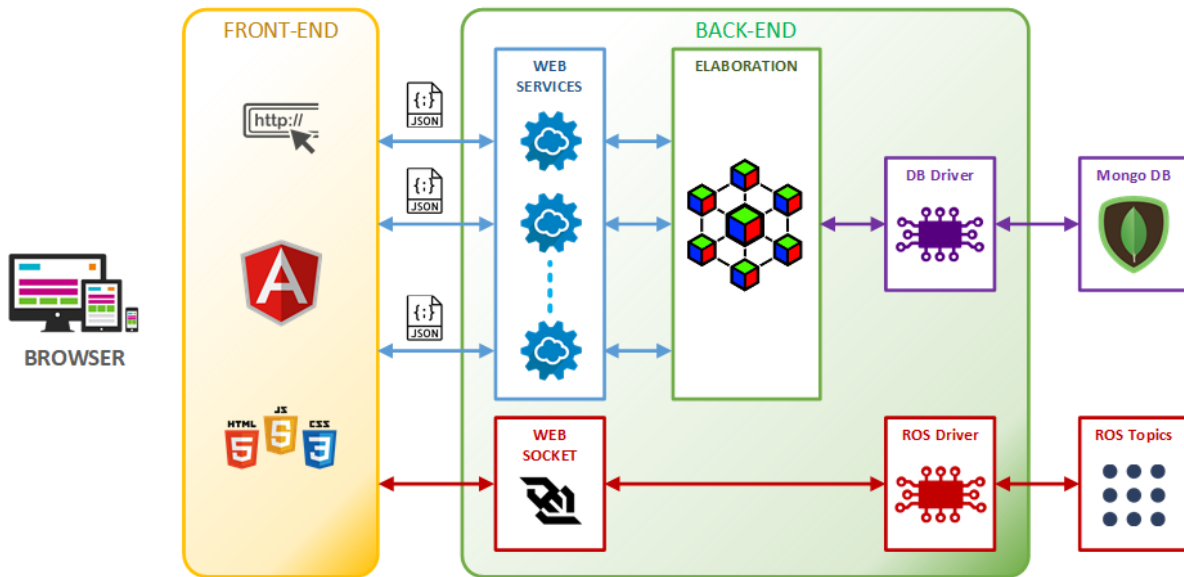


Figure 69 - Back-end architecture design

Back-end positioning in overall structure

Back-end is architecturally a background feature. It must be able to exchange data with higher (front-end) and lower (data storage, messaging features and so on) levels, so its position in the feature ranking is in a middle layer. It has no direct contact and information exchange both with users and with on-field devices, but it is in charge of coordinating the data flow between those two ends of the architectural ladder.

Back-end software development environment

In order to minimize software development and maximize software efficiency the use of microservices development environment is recommended. Interaction with front-end can be guaranteed by internal web services features, communication with data storage by MongoDB library addon, communication with the ROS environment by the ROS toolkit and notification delivery to the front-end by the web socket feature

Back-end internal operative structure

The back-end is composed, basically, by 4 blocks:

1. a listening web service task,
2. a sending web service task,
3. an elaboration structure,
4. a notification task.



Listening and sending web services can act separately when in asynchronous mode, or together when in synchronous feature. They will receive data in JSON format. Data should include the request type and all the details for the data elaboration and reply creation.

The request type will enable the specific function in the elaboration structure. Data will be requested from storage layer and elaborated there. Reply JSON file will also be created in this function. After file creation, the JSON will be transferred to the sending web service for delivery and request closing.

A continuous polling on ROS notification environment will always be active for detecting new messages for front-end. When a new notification is acquired, the function enables a web socket transmission to the front-end, retrieving the polling status afterwards.

8 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, S. Lamprecht, R. Retzlaff, T. Udelhoven, R. B. Nico, E. Garone, V. Cristofori, M. Contarini, M. Paolucci, C. Silvestri, S. Speranza, E. Graziani, R. Stelliferi, R. F. Carpio, J. Maiolini, R. Torlone, G. Ulivi and A. Gasparri, "PANTHEON: SCADA for Precision Agriculture," in *Handbook of Real-Time Computing*, Springer, 2019.
- [3] V. Cristofori, S. Speranza and C. Silvestri, "Developing hazelnuts as a sustainable and industrial crop," in *Achieving sustainable cultivation of tree nuts*, Burleigh Dodds, 2019.
- [4] I. Computer Society, "IEEE Recommended Practice for Software Requirements Specifications," IEEE, 2009.
- [5] I. Computer Society, "IEEE Recommended Practice for Software Design Descriptions," IEEE, 2009.