D4.2 - Decision Enablers, Advisory Support Tools and DEMETER Stakeholder Open Collaboration Space

Dissemination Level: Public
Submission Date: 03rd July 2020

Contents

List of Figures

List of Tables

1 Executive Summary

DEMETER aims to lead the Digital Transformation of the European Agrifood sector based on the rapid adoption of advanced technologies, such as Internet of Things, Artificial Intelligence, Big Data, Decision Support, Benchmarking, Earth Observation, etc., to increase performance in multiple aspects of farming operations, as well as to assure the viability and sustainability of the sector in the long term. It aims to put these digital technologies at the service of farmers using a human-in-the-loop approach that constantly focuses on mixing human knowledge and expertise with digital information. DEMETER focuses on interoperability as the main digital enabler, extending the coverage of interoperability across data, platforms, services, applications and online intelligence, as well as human knowledge, and the implementation of interoperability by connecting farmers and advisors with providers of ICT solutions and machinery.

To enable the achievement of the aforementioned objectives, and to promote the targeted technological, business, adoption and socio-economic impacts, DEMETER will design and develop a targeted decision support system (DSS) that will enable the delivery of tailored advisory services to the agricultural sector. This DSS will combine the data analytics from WP2 with AI-based expert system, machine learning and benchmarking techniques to provide precision decision support to the users. This deliverable describes the AI-based analytic functions, benchmarking techniques and performance monitoring tools that serve as core building blocks of the DEMETER DSS for addressing the needs of the pilots. While delivering the AI building block services, full-service lifecycle, including analysis, development, testing, etc., will be carried out to cover each step of the DSS development. The pilots have provided a minimum set of data needed to calculate the indicators. Along with the benchmark tools DEMETER will be providing a reporting tool to the pilots and farmers, enabling them to improve the data collection activities to obtain a more specific benchmark report. These benchmark tools will implement a set of flexible rules to define the optimal list of farms with similar condition according with size, environmental, and economic conditions, type of farming activities. The selected farmers will be used to provide to a specific benchmark value of the indicators.

Starting from the definition of the different components that the DEMETER DSS will be developing to cover the needs from the pilots, several advancements are proposed. The DSS components will be evolved from their initial brief description reported in D4.1 to a more complete description by containing links to WP2 and WP3 (by means of the related Generic Enablers), a draft development plan and an initial API. Alongside these aspects, responsible and interested partners are also proposed.

The other two important points under the scope of this document are the data visualization module, with which users will be presented with intuitive meaning thanks to Knowage and the implementation of the SOCS, where initial steps will be taken to describe this space for collaborating, sharing of best practices, and participating in the co-creation processes. This collaboration space makes a farmer's needs visible to advisors and developers.

2 Acronyms and Glossary

Acronyms	
AGPL	Affero General Public License
AI	Artificial Intelligence
AIM	Agricultural Information Model
API	Application Programming Interface
ARIMA	Autoregressive Integrated Moving Average
BBCH	Bayer, BASF, Ciba-Geigy, and Hoechst
CAN	Controller Area Network

CKAN	Comprehensive Knowledge Archive Network
COCO	Common Objects in Context
CORS	Cross-Origin Resource Sharing
CRUD	Create, Read, Update, Delete operations
CSV	Comma-Separated Values
CWSI	Crop Water Stress Index
DOY	Day of the Year
DSM	Digital Surface Model
DSS	Decision Support System
DTM	Digital Terrain Model
DWH	Datawarehouse
ECMWF	European Centre for Medium-Range Weather Forecasts
ET_0	Reference Evapotranspiration
ETL	Extract-Transform-Load
ETS	Exponential Smoothing Algorithm
FADN	Farm Accountancy Data Network
FAPAR	Fraction of Absorbed Photosynthetically Active Radiation
FaST	Farm Sustainability Tool
FC	Field Capacity
GDD	Growing Degree Day
GEE	Google Earth Engine
GIS	Geographic information system
GPS	Global Positioning System
GSP	Gross Saleable Production
IACS	Integrated Administration and Control System
ICT	Information and Communications Technology
JPY	Java Python Library
JRI	Java R Interface
JSON	JavaScript Object Notation
JSON-LD	JSON Linked Data
JSONP	JSON with Padding
KPI	Key performance indicators
LPIS	Land Parcel Information System
LSU	Livestock Standard Unit
MAA	Multi-Actor Approach
MFCC	Mel-Frequency Cepstral coefficients
MFFB	Mel-Frequency Filter Bank
MHR	Medical History Review
ML	Machine Learning
MSI	MultiSpectral Instrument
MVP	Minimum Viable Product
NDVI	Normalized Difference Vegetation Index
NPTS	Non-Parametric Time Series
NUTS	Nomenclature of Territorial Units for Statistics
OEM	Original Equipment Manufacturer
Pascal VOC	Pascal Visual Object Classes
PEMS	Portable Emission Measurement Systems
PLF	Precision Livestock Farming
PWP	Permanent wilting point

QBE	Query by Example
REST	Representational state transfer
RGB	Red-Green-Blue
RH	Relative Humidity
RMSE	Root Mean Square Error
ROI	Return of Investment
RS	Remote Sensing
SAR	Specific Absorption Rate
SDK	Software Development Kit
SOA	Service Oriented Architecture
SOCS	Stakeholders Open Collaboration Space
SSO	Single Sign On
SVM	Support Vector Machines
UAV	Unmanned Aerial Vehicle
UI	User Interface
UUID	Universally Unique Identifier
WP	Water Pressure
XLS	Excel Spreadsheet

Glossary	
DEMETER Engagement Hub (DEH)	The **DEH is a registry where suppliers can describe their resources** which become accessible by developers to be used through the "co-creation process" to build new solutions. **DEH is a space dedicated to developers** since its focus is on enabling users (acting as DEMETER Providers) to register their offered resources (components, devices, services, data sources, platforms, etc.), and users (acting as DEMETER Consumers) to browse it to discover suitable resources matching their requirements (search API or tags).
Agricultural Interoperability Space (AIS)	The **AIS enables the secure interoperability and integration of different agriculture platforms, technology components and datasets, utilizing a brokerage environment based on producers/consumers**. **AIS is a space dedicated to developers** since its focus is on delivering a full set of interoperability mechanisms to actually deploy the solutions (which consists of resources such as components, devices, services, data sources, platforms, etc. that are accessible for deployment) through the DEMETER enabled interoperability mechanisms.

3 List of Authors and Reviewers

Organisation	Author
ICE	Oscar Garcia (Editor)
	John Beattie
	Ross Campbell
ENG	Antonio Caruso
	Francesca Cantore
TRAGSA	Pablo Gallego
SIVECO	Mihai Angelihou
VICOMTECH	Izar Azpiroz

Organisation	Author
Lesprojekt	Jaroslav Smejkal
AGRICOLUS	Diego Guidotti
DNET	Nenad Gligoric
	Senka Gajinov
SINTEF	Arne J. Berre
	Bjørn Marius von Zernichow
MIMIRO	Harald Volden
Landbrukets	Erland Kjesbu
ROTECHNOLOGY	Lorenzo Bortoloni
ATOS	Sergio Salmerón
	Tomás Pariente
UMU	Manuel Mora
JD	Andreas Schroeder
VITO	Bart Beusen
	Isabelle Piccard

Organisation	Reviewer
INESC Porto	Filipe Neves dos Santos
OGC	Martin Klopfer

4 Introduction

This deliverable summarizes the initial results of various WP4 tasks related to **Decision Enablers, Advisory Support Tools and DEMETER Stakeholder Open Collaboration Space**.

Aims of these tasks were:

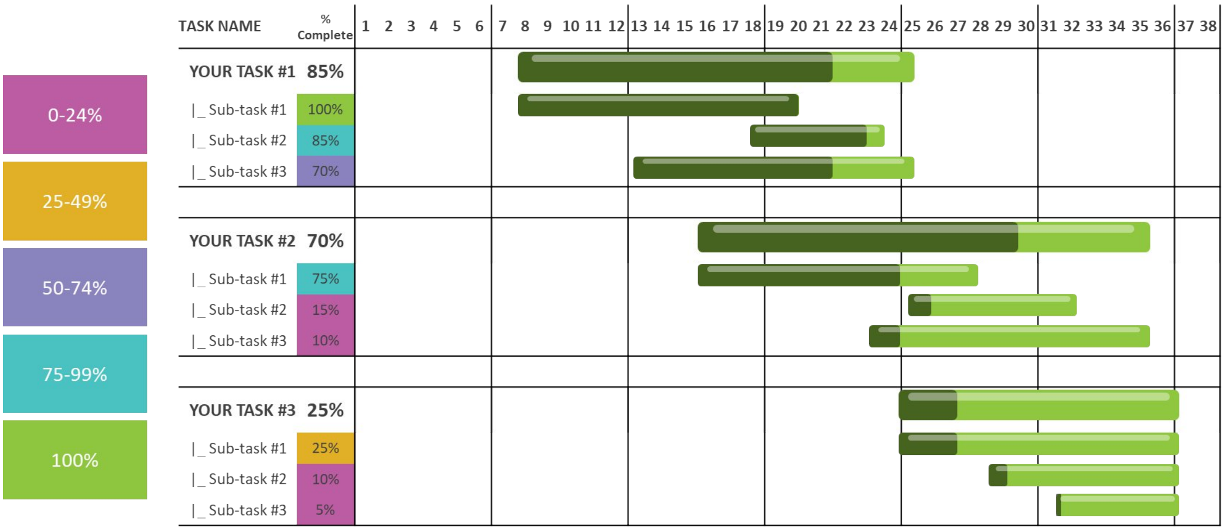
- to develop a self-service dashboard framework capable of integrating a number of service frontends, transformation operators and visualisation widgets,
- to develop generic enablers as open components that accelerate the development of sector or pilot specific decision support systems, and
- to develop cloud-based collaboration tools that enable and support collaboration among different stakeholders in a pilot domain.

The document is structured as follows.

Section 5 details the different components that the DEMETER DSS will be developing to cover the needs from the pilots. Starting from the initial list of components provided in D4.1, where a brief description of the functions and the input/output structure of the components of the DSS to be integrated in the frame of DEMETER was reported, their description has been improved by adding the following elements:

- Related Generic Enablers: List of components from WP2 (e.g. Ontology mappings), WP3 (logins, security, etc.).
- Status: Whether the component already exists, or it is under development, to be produced by DEMETER, or it exists but needs upgrades/modifications to match DEMETER needs.
- Development Plan: An initial development plan to develop the functionalities of a given component.
- Responsible Partners (name and organisation): The initial responsible person in charge of a given component.
- Partners involved: The list of partners involved in the development of a given component.
- API: The API that will be used to send/retrieve the data and/or access functionalities of a given component.

The development plan follows the template below in which a semaphore-like characterisation has been put in place to ensure that a tracking of the tasks is being kept based on a project months' timeline.



Section 6 shows the main advances done for the data visualization module, in which users will be presented with intuitive means to interactively explore and analyse data, enabling them to effectively identify interesting patterns, infer correlations and causalities, and supports sense-making activities. The technology selected to meet this need in DEMETER is represented by Knowage ([1]), which is an open source suite developed by ENG that combines traditional data and big data sources into valuable and meaningful information.

Section 7 provides a description of the Stakeholders Open Collaboration Space **(SOCS),** which is a space **dedicated to all stakeholders** (farmers, advisors, and suppliers) where they can **collaborate, share best practices, and participate to co-creation processes**. This collaboration space makes a farmer need visible to advisors and developers and conveys the information coming from the farmers as input to select the most suitable resources registered in DEH to be used to build the optimal solution. The SOCS will act as a facility intended to support knowledge sharing, lessons learnt and co-creation of solutions, based on human to human interaction, communication, and technology information.

5 Decision Support Enablers and Advisory Support Tools

After the activities reported in D4.1, the following is the classification adopted by WP4 to the different pilots. It is worth noting that the list of components from D4.1, though pretty complete, has been updated resulting in two components being removed and one component being added:

- Removed component(s):
  - **(old) Component E.3 – Pesticide Monitoring**: None of the related pilots (1.1, 3.3 and 5.3) had a strong requirement on the development of this component.
  - **(old) Component H.2 – Product Preference**: The related pilot (5.2) interested in this component has withdrawn their interest in this component. As a side effect, (old) component H.3 has been renumbered to (new) H.2.
- Added component(s):
  - **(new) Component H.3 – Field Book and FaST**: Since farmers must create a field book where they must record some information about their farms, including information related to phytosanitary treatments, and currently this field book is being done on paper, we have agreed that a component is deemed necessary to facilitate this tracking activity which is going to be interesting for all pilots.

The table below details the different components classified corresponding to their areas and the lead partner:

Type of Area	Area	Lead Partner	Component
Crop Farming	A - Crop Growth, Status and Yield	VITO	A.1 Plant Yield Estimation
			A.2 Plant Phenology Estimation
			A.3 Plant Stress Detection
			A.4 Detect Crop Type
			A.5 Estimate Beehive
	B - Irrigation Management	UMU	B.1 Water Balance Model
			B.2 Data Fusion for Irrigation

	C - Nutrition Management	SIVECO	C.1 Nitrogen Balance Model
			C.2 Nutrient Monitor
	D - Machinery and Field Operations	JD	D.1 Emission
			D.2 Field Operation
			D.3 Variable Rate
	E - Pest and Disease Management	ICE	E.1 Computer Vision-based Counting Module
			E.2 Estimate temperature-related pest events
Livestock Farming	F - Animal Yield	ENG	F.1 Estimate Milk Production
			F.2 Poultry Feeding
	G - Animal Welfare	MIMIRO	G.1 Estimate Animal Welfare Condition
			G.2 Stress Recognition: Support Vector Machine for Poultry Stress detection
General decisions about farm management	H - Traceability	ROTECH	H.1 Traceability
			H.2 Transport Condition
			H.3 Field Book and FaST
	I - Benchmarking	AGRICOLUS	I.0 Indicator Engine for Benchmarking Purpose
			I.1 Generic Farm Comparison
			I.2 Neighbour Benchmarking
			I.3 Technology Benchmarking

Table 1: DEMETER Decision Support Components

The next sections describe the functionalities and the different elements of every single component of the DEMETER DSS. These elements are the input/output structure and the related Generic Enablers, which are addressed in WP2 (e.g. Ontology mappings), WP3 (logins, security, etc). For each component, the status is described, looking at whether it already exists, is currently in development, to be produced by DEMETER, or actually exists but needs upgrades/modifications to match DEMETER needs. In addition, the development plan and the partners involved in its development are described together with the generic API and the pilots that will have access to this component.

5.1 DSS AREA: A - Crop Growth, Status and Yield

5.1.1 Component A.1 Plant Yield Estimation

What: Data-driven Machine learning (ML) algorithms to predict yields, using as input a combination of crop type and crop variety, remote sensing input (e.g. NDVI), soil data, meteorological data and future meteo predictions (up to the end of the growing season). A ML model needs target data to be trained, in this case historical yield data will be needed to train the machine learning models.

Crops have different optimal climate, water and nutrient requirements, and sensitivity to extreme weather, invasive pests, and other impacts. Farmers adjust planting and harvesting times, available water, and nutrients by irrigating and adding fertilizer. Yield predictions typically must be modelled using data that is often incomplete, asynchronous, and collected at different spatial and temporal scales with different levels of uncertainty. Variability due to climate, soil, pests, and management decisions contribute to further structural and functional complexity of managed ecosystems.

Remote Sensing Multi-temporal Imagery data can be either spatially coarse but temporally dense, e.g. Proba-V at 300m resolution and daily frequency, or spatially finer but temporally sparser, e.g. Sentinel-1 at 10m spatial resolution and 5-daily interval for central Europe, or monthly drone flights at 2cm resolution. Historical time series data may provide a view on what crop growth curve we may expect, while any deviation from the optimal curve will signal potential problems for crop growth (e.g. water stress).

There is a large variability between DEMETER pilots on what they expect from such a ML yield prediction module. For some crops, yield predictions require data for the complete growing season up to the point of prediction, while for other crops yield predictions may require only data of the day of the prediction. If we regard every data input as a timeseries, the number of timesteps in the timeseries data may be 1 or more. In the case that more than 1 timestep is required, the temporal resolution of the data that is used as input to the ML yield prediction model, may also vary significantly, going from hourly data to monthly data. It is important to note here that the higher the temporal resolution of the input timeseries, the more parameters there are in the ML model to be trained, and the more training data will be needed to make the model robust and avoid overfitting.

A machine learning model should thus be flexible in terms of input data. The AI based tools may be based on a collection of general architectures, that are adaptable in terms of the exact input they require. Examples of possible algorithms are support vector machines, decision trees, random forest, and neural networks. Several (untrained) ML architectures may be provided, that are agnostic to the spatial resolution of its inputs, and that are configurable in the temporal resolution of the inputs: a specific neural network architecture may be used to train a model on input data that has weekly temporal resolution, while it may also be used to train a model that deals with daily values for each of its inputs, although this will mean that more training data will be needed as the size of the neural network grows with larger input size.

If the yield prediction is only needed near the end of the growing season, then for some crops the last available NDVI image may contain enough information for the yield prediction. For providing irrigation advice in the middle of the season, some crops will require knowledge of the effect of irrigation quantities on the final yield.

For the case of orchards and vineyards, yield prediction may be done by counting the fruit some weeks before harvest, using deep learning object detection algorithms on 2D images. This methodology for yield prediction is more directly related to the AI components from DEMETER WP2, and will not be part of the methods proposed in this section.

Input: all input data may be regarded as timeseries with the number of timesteps >=1, and with different temporal and spatial resolutions. The spatial resolution of the prediction will be defined by the spatial resolution of the ground truth data that is available on yields.

Input Data Arable Crops	
Field coordinates and geometry	Mandatory
Crop info	Crop type: Mandatory Crop variety-phenotype: Optional
Planting data	Default: planting date entered by farmer
Remote Sensing Multi-temporal	• Default: Sentinel-2 spectral information and derived

Imagery data	biophysical parameters • Optional refinements o Interpolation of missing RS data using data fusion (e.g. CropSAR enhanced NDVI timeseries – VITO)
Meteorological data	• Default: Data from Satellite (ECMWF ERA5, Copernicus) • Optional (data provided by user) o National or regional webservice o local meteo station in the neighbourhood o local meteo station in the field • subset of air temperature, air relative humidity, rain, wind speed, wind direction, solar radiation
Irrigation data	Mandatory input from farmer (if applied) • water quantity
Soil data	• Default: soil type • Optional: In-field soil sensors: soil moisture
Prediction Input data	
Predicted meteorological data	• prediction data provided by user o average air temperature, rainfall
Future Irrigation schedules	• provided by the user o water quantity per day
Target Data (training data)	
Historical crop yield data	Should be provided by the user • Crop yield per field/management zone (ton/ha)

Table 2: Component A1 Input Data

Output:

Output Data	
Predicted yield	Same resolution as training data (historical yields)
Optional: sensitivity analysis	Which input parameters contribute most to the value of the predicted yield (input for Decision Support system)

Table 3: Component A1 Output Data

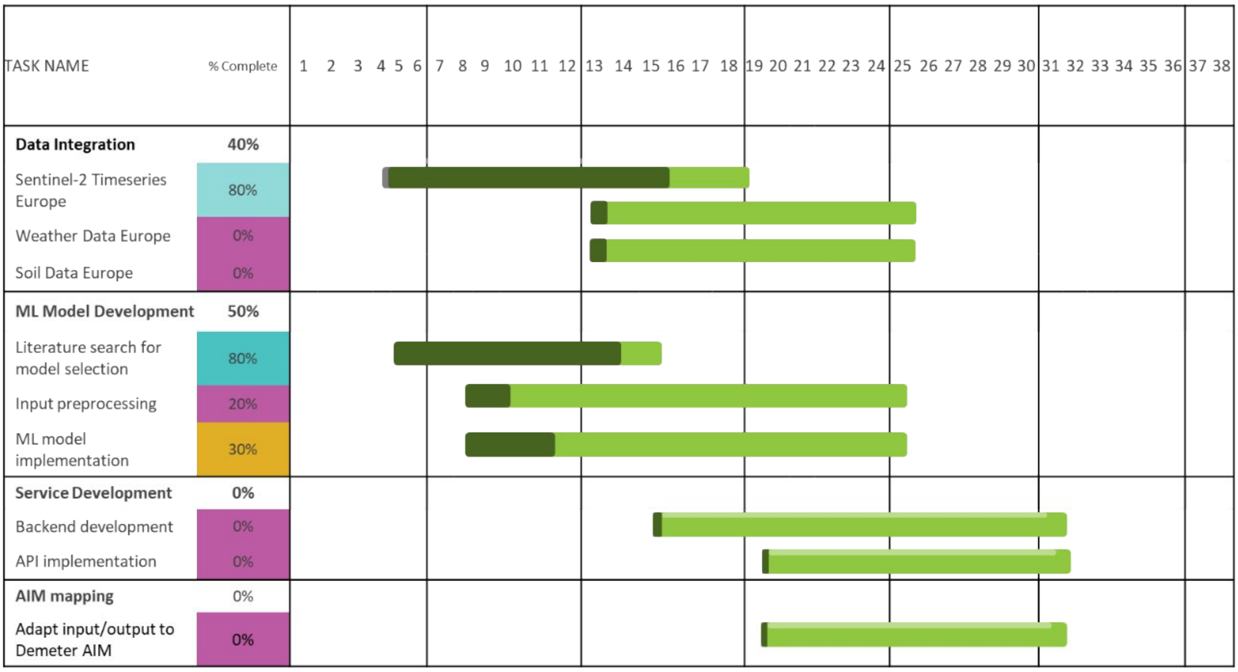
Related Generic Enablers: The ML model needs standardized and secure ways to access data:

- Semantic interoperability
- Standardized interfaces for data retrieval from various sources:
    - Data security
    - Data ownership
    - Access rights

Status: To be produced by DEMETER.

Development Plan: Development will start with examples of ML for pilots where input data and target data (training data) are available: pilots 1.3, 3.4 and 1.4. It might be possible that pilots 1.1 & 1.2 may have training data available, but this is still in-situ:

- Implement data download methods for sentinel-2, weather data and soil information.
- Implement a set of machine learning algorithms from which the user can choose.
- Develop the backend services to make the service available through REST API.
- Adapt data inputs to accept data in the AIM format that will be developed in DEMETER.



TASK NAME	% Complete
Data Integration	**40%**
Sentinel-2 Timeseries Europe	80%
Weather Data Europe	0%
Soil Data Europe	0%
ML Model Development	**50%**
Literature search for model selection	80%
Input preprocessing	20%
ML model implementation	30%
Service Development	**0%**
Backend development	0%
API implementation	0%
AIM mapping	0%
Adapt input/output to Demeter AIM	0%

Responsible Partner (name and organisation): Bart Beusen (VITO)

Partners involved: VITO, OdinS, ELGO, SIVECO

API:

<u>Calls</u>

- *POST /yield_prediction/{architecture_uuid}/train: train this model architecture on a given training input feature set; return an identifier of the trained model.*
- *POST /yield_prediction/{model_uuid}/predict: predict yield for a given prediction input feature set using the specified model; return the predicted yield value (ton/ha).*
- *GET /yield_prediction/trained_models: receive a list of trained models that are available to this user.*
- *GET /yield_prediction/model_architectures: receive a list of ML model architectures that are available to this user for training on the user's training data, together with the configuration parameters per architecture.*

<u>Objects</u>

- "training input features" is an object with the following variables per field:
  - field geometry (polygon)
  - planting date
  - harvest date
  - irrigation data per day ({date, quantity})
  - weather data per day ({date, min temp, max temp, rainfall})
  - crop yield (ton/ha)
  - crop type
  - crop variety
  - soil type
  - soil moisture measurement {(date, value)}
  - ML Model configuration parameters

- "prediction input features" is an object with the following variable per field:
  - field geometry (polygon)
  - planting dates
  - predicted harvest date
  - future irrigation scheme per field ({date, quantity})
  - future weather scenario ({date, min temp, max temp, rainfall})
- Trained models: A list of trained models available to the user:
  - UUID
  - Crop type
  - Crop variety
- Model architecture: A list of available ML architectures (e.g. Support Vector Machine, Multilayer Perceptron, Random Forest):
  - Name
  - Description
  - Configuration parameters

Pilots involved: 1.1 & 1.2, 1.3, 1.4, 3.4

5.1.2 Component A.2 Plant Phenology Estimation

What: A component that estimates the date of a phenology stage of a crop. For the baseline model, ambient temperature is the main factor in plant development. Ambient temperature is transformed into GDD and the date is transformed into day-of-year (DOY). DOY is the number of days elapsed since the first day of the current year. The calculation of DOY from a date is straightforward. On the other hand, GDD are a measure of heat accumulation defined as the number of temperature degrees above a base temperature, the base temperature being the temperature below which plant growth is zero.

The main goal is to provide a more generous weather feature set together with a more efficient plant phenology prediction model. Currently, these measurements are default weather data that are available via Copernicus or more general platforms (e.g. Google Earth Engine). Classification and Regression distinct Machine Learning have been compared, and decision-tree based models seems to provide the most accurate results.

Input: All input data should be, at least, daily measured.

Input Data Arable Crops	
Field coordinates and geometry	Mandatory
Meteorological data	• Default: Data from Satellite (ECMWF ERA5, Copernicus) • Optional (data access provided by user) o National or regional webservice o Local meteo station in the neighbourhood o Local meteo station in the field • Subset of daily measurements: air temperature (minimum and maximum), air relative humidity, rain, wind speed, wind direction, solar radiation. • Mandatory: Daily minimum and maximum temperature for growing degree day calculation, starting from the first day of the year.
Time data	DOY: Day of the year

Prediction Input data	
Predicted meteorological data	Default: ERA-5 Optional: prediction data access provided by user • Subset of daily measurements: air temperature (minimum and maximum), air relative humidity, rain, wind speed, wind direction, solar radiation.
Target Data (training data)	
Historical olive phenology data	Provided by AGRICOLUS, according to BBCH scale.

Table 4: Component A2 Input Data

Output: An estimation of the date at which the main phenological phases occur.

Output Data	
Predicted yield	Same resolution as training data (historical yields)

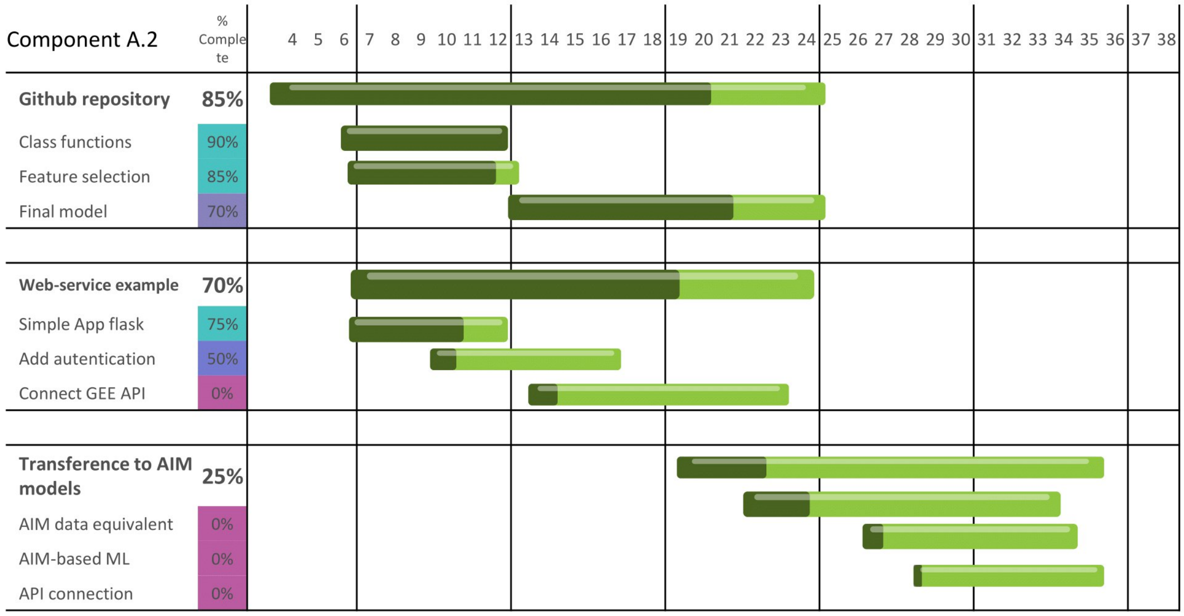
Table 5: Component A2 Input Data

Related Generic Enablers: Core Generic Enablers, Data & Knowledge Enablers, Decision Support and Performance Monitoring Enablers.

Status: In development. Current basic models exist, but more complete data-set predictions (Copernicus ERA5) need to be validated. Feature selection section is not consistent. In addition, the existing data sets and models need to be adapted to AIM format to match DEMETER integration requirements.

Development Plan:

- Transference to GitHub repository.
- Add authentication and authentication wrappers to the web service example.
- Replace the simple model by more complex one.
- Connect the web service to GEE to demand weather data.
- Transference to AIM data and ML models.



Responsible Partner (name and organisation): Izar Azpiroz (VICOM)

Partners involved: Vicomtech, AGRICOLUS

API: Google Earth Engine or Copernicus data extraction engine. It should have the following interactions with Orion or another General Context Manager:

Calls

- *POST /pheno_component/{crop_location_code}/create: create a space for predicting phenology for a specific crop or area.*
- *POST /pheno_component/{crop_location_code}/train: receive an array of the "training input features" to train the models; the training is incremental, e.g. the features will be added to the training set; return an object with the training result.*
- *POST /pheno_component/{crop_location_code}/predict: receive an array of the "predict input features" and return an array of "prediction result".*

Objects

- "training input features" is an object with the following variables
    - date: date of the field observation
    - lat/lon of the trap
    - pheno_phase: A numeric status describing the phenologic state.
    - weather: an array of weather data; each element contains (at least for the simplest model)
        - date: date of observation
        - tmin: minimum daily temperature
        - tmax: maximum daily temperature
- "training result"
    - confusion_matrix or rmse
- "predict input features":
    - contains the same input variables as "training input features" except the pheno_phase value
- "predict result":
    - date: date of the field observation
    - lat/lon of the trap
    - estimated_pheno_phase: value of the estimated phenological phase

Pilots involved: 3.1

5.1.3 Component A.3 Plant Stress Detection

What: Plant stress can come from many causes such as: burns (very high temperatures), lack of water, lack of nutrients in the soil, etc. In this case, as a modern method of determining plant problems, the composition of the NDVI vegetation plan can be used.

Input: Farm data (geographic coordinates), crop traits, used inputs, used hybrid, seeding density, soil texture, soil moisture, agrometeorological data (temperature, rainfall, leaf wetness), in-situ determination (such as Chlorophyll meter), satellite images (Proba – V, Sentinel, Landsat). NDVI index correlated with the in-situ measurements for obtaining a viable modern method to use based on satellite images. The transmission of data from the field will be done through a custom application available to farmers.

European Union
European Regional
Development Fund

Field coordinates and geometry	Mandatory
Historical crop data	Crop inputs: Mandatory Seeding density: Mandatory
Soil data	Elevation: Mandatory
System input data	
Remote Sensing Multi-temporal Imagery data	• Default: Landsat7, Sentinel-2 spectral information • Proba-V NDVI
Meteorological data	• Default: Data from Satellite (ECMWF ERA5, Copernicus, Darksky) • National or regional data and predictions • National or regional historical data • Local meteo station in the field (soil moisture)
In-situ determination	• Chlorophyll meter • N-Tester

Table 6: Component A3 Input Data

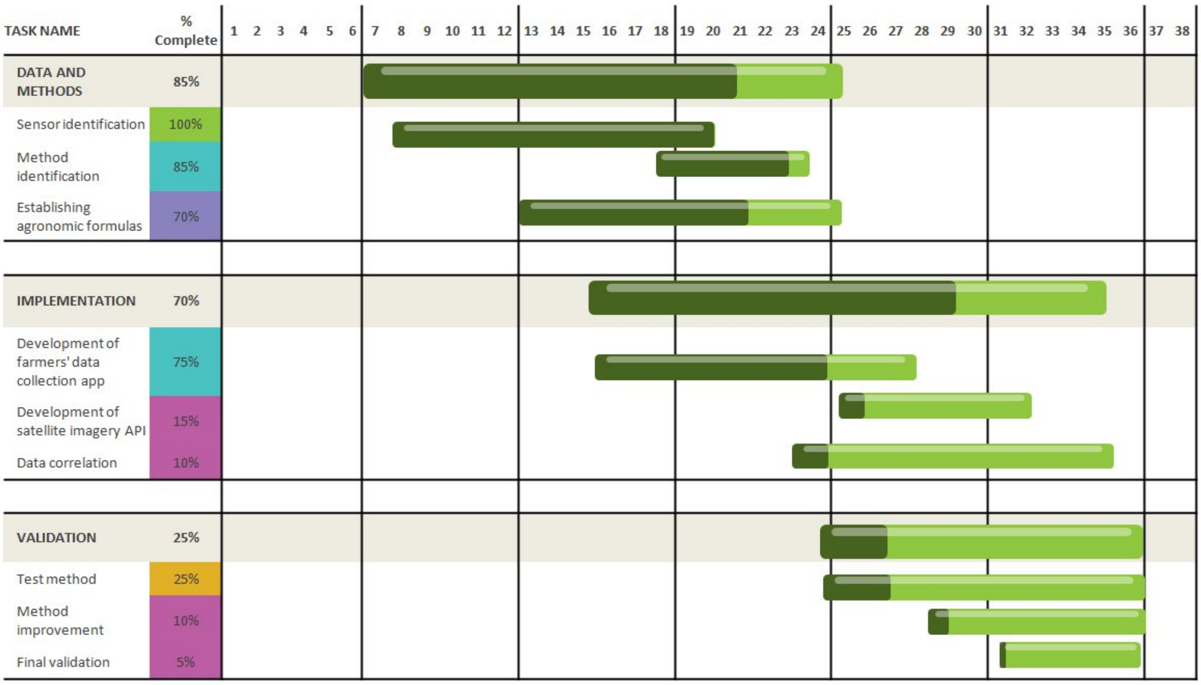
Output: Deliverables to farmers will consist of recommendations for the distribution of important material for saving crops or even replanting certain areas of the plot.

Recommendations can also be in the form of warnings based on agrometeorological sensors or cartographic input distribution plans.

Output Data	
Pest alert	Possible pest alert
Meteorological alerts	Extreme meteorological alerts
Treatment recommendations	Recommended treatments
Planting recommendations	Recommended planting instructions or replanting

Table 7: Component A3 Output Data

Related Generic Enablers: Security Enabler, Semantic Enabler
Status: To be produced by DEMETER
Development Plan: In the first phase, the problems appeared during the whole vegetation period of the corn culture will be identified, then data will be collected from the ground and will be correlated with the satellite images; the final result will be based on the new legend of aerial images interpretation, in accordance with the data collected by the ground.



Responsible Partner (name and organisation): Mihai Angelihou (SIVECO)

Partners involved: SIVECO, APPR
API:

Calls

- *POST /crops/stress_detection: recommends treatent and planting information. Also returns possible pest alerts and extreme weather alerts.*

Objects

- field_geometry: polygon (mandatory).
- crop_inputs: predefined list of values- hybrid, insecticide, herbicide, fertilizers (mandatory).
- seeding_density: information regarding the seeding density (mandatory).
- elevation: numeric (optional).

Pilots involved: 1.4

5.1.4 Component A.4 Detect Crop Type

What: Detect different crop types on the fields. The most conventional way of performing satellite-based crop classifications is the use of optical imagery, which started with the Landsat missions. Gradually the spatial, temporal, and spectral resolution has increased since, constantly improving the classification results. In June 2015, Sentinel-2 was launched as part of the Copernicus Sentinel-2 mission. This mission consists of a constellation of two identical satellites, Sentinel-2A and Sentinel-2B. The revisit frequency of one satellite is 10 days, resulting in a revisit frequency of 5 days for the constellation. Sentinel-2 carries a MultiSpectral Instrument (MSI) with 13 bands in the visible, near infrared and shortwave infrared part of the spectrum. These state-of-the-art specifications have been used in recent research on crop classification.

A major drawback of optical sensors remains clouds that cannot be penetrated by optical radiation. Clouds and cloud shadows therefore lead to gaps in optical imagery, and missing data in optical time series. For classification and monitoring purposes, this drawback can significantly affect performance.

Possible strategies to overcome the disadvantages of clouds and shadows are multi-sensor combinations, especially strategies that exploit different parts of the electromagnetic spectrum. With the advent of the Sentinel-1 mission, SAR data now has almost global data availability at no charge and regular time intervals. The revisit frequency is 6 days now that both the Sentinel-1A as well as the Sentinel-1B satellites are operational. Considering overlap and combining ascending and descending orbits, this leads to a Sentinel-1 observation every ~2 days in Europe. Contrary to optical imagery that provides insights into biophysical processes of vegetation, SAR imagery reflects the structural components of vegetation and the underlying surface, resulting in a complementary data source that can be used for crop classification and monitoring purposes.

Input:

Data input crop mapping	
Field coordinates and geometry	Mandatory
Optical Remote Sensing Multi-temporal data	Default: Sentinel-2 spectral information and derived biophysical parameters Optional refinements • Interpolation of missing RD data using data fusion (e.g. CropSAR enhanced NDVI timeseries – VITO)
SAR Remote Sensing Multi-temporal data	Sentinel-1 SAR data
Target Data (training data)	
Crop type	Crop type per field

Table 8: Component A4 Input Data

Target data (training data): To train a crop mapping algorithm, the user needs to provide field geometries and associated crop types. The Integrated Administration and Control System (IACS) is the most important system for the management and control of payments to farmers made by the Member States in application of the European Common Agricultural Policy. The Land Parcel Information System (LPIS) is the spatial database that permits spatial and alphanumeric queries and data retrieval in function of the subsidy application and administrative cross-checks. In Belgium this is arranged at the regional level, resulting in two similar datasets for the Flemish and the Walloon regions. From these datasets, crop cover per agricultural parcel can be extracted. Similar initiatives may exist in other European countries.

Output: Crop type class (number of classes and crop types to be detected, are user-defined)

Related Generic Enablers:

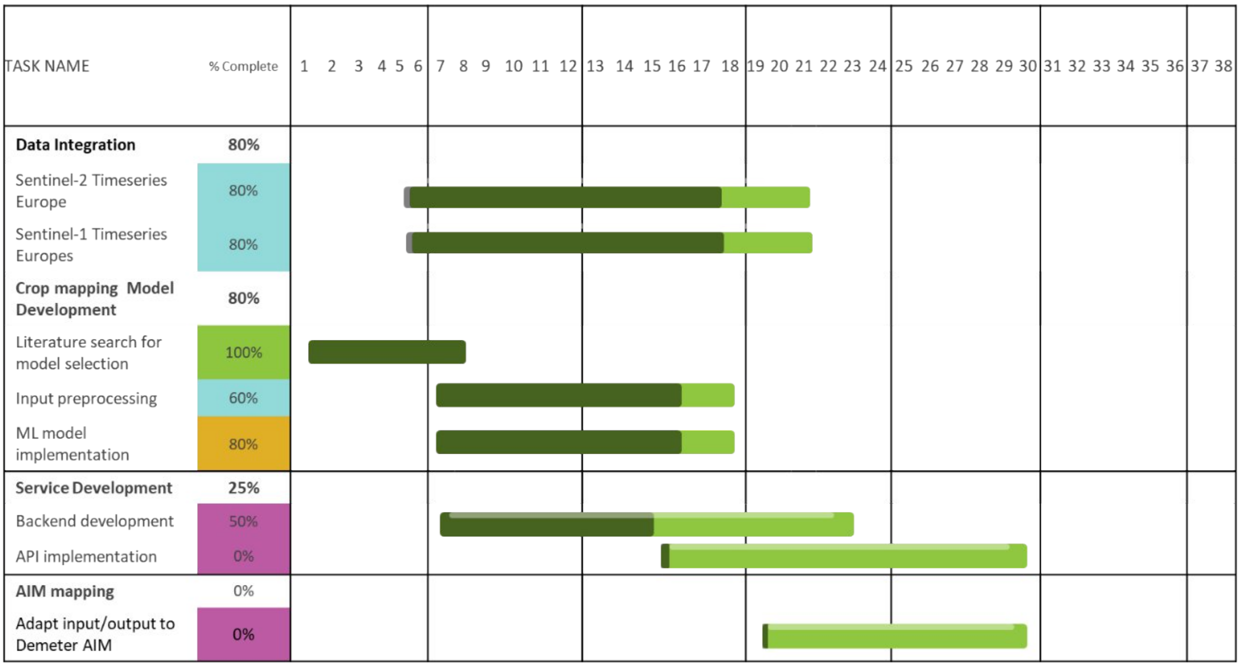
- Semantic interoperability.
- Standardized interfaces for data retrieval from various sources:
    - Data security.
    - Data ownership.
    - Access rights.

Status: first ML model implementation ready for the case of Belgium but needs upgrades/ modifications to be applicable European wide, and to match DEMETER needs.

Development Plan:

- Implement data download methods for Sentinel-1 and sentinel-2.
- Implement a set of machine learning algorithms from which the user can choose.

- Develop the backend services to make the service available through REST API.
- Adapt data inputs to accept data in the AIM format that will be developed in DEMETER.



TASK NAME	% Complete
Data Integration	**80%**
Sentinel-2 Timeseries Europe	80%
Sentinel-1 Timeseries Europes	80%
Crop mapping Model Development	**80%**
Literature search for model selection	100%
Input preprocessing	60%
ML model implementation	80%
Service Development	**25%**
Backend development	50%
API implementation	0%
AIM mapping	0%
Adapt input/output to Demeter AIM	0%

Responsible Partner (name and organisation): Bart Beusen (VITO)

Partners involved: VITO

API:

Calls

- *POST /croptype_detection/{architecture_uuid}/train: train this model architecture on a given training input feature set; return an identifier of the trained model.*
- *POST /croptype_detection/{model_uuid}/predict: predict crop types for a given prediction input feature set using the specified model; return the predicted crop type.*
- *GET /croptype_detection/trained_models: receive a list of trained models that are available to this user.*
- *GET /croptype_detection/model_architectures: receive a list of ML model architectures that are available to this user for training on the user's training data, together with the configuration parameters per architecture.*

Objects

- “training input features” is an object with the following variables per field:
  - field geometry (polygon)
  - Date: start of season
  - Date: end of season
  - crop type
  - crop variety
  - ML Model configuration parameters (including nr of classes)
- “prediction input features” is an object with the following variables per field for which the crop type needs to be detected
  - field geometry (polygon)
  - Date: start of season

    - Date: end of season
- Trained models
    - A list of trained models available to the user
        - UUID
        - Classes: crop types that can be detected by the model
- Model architecture
    - A list of available ML architectures that are available for training
        - Name
        - Description
        - Configuration parameters

Pilots involved: 3.4

5.1.5 Component A.5 Estimate Beehive

What: A component that estimates the number of beehives required to pollinate a particular field.

Input: The component will require size and location data of the field to be queried. This may be in the form of geocoordinates representing the field boundaries from which an area can be calculated or an area (m^2) and the central geocoordinate of the field. Data relating to the field contents will also be required in the form of the Crop Type present (e.g. maize or wheat), Crop Status (maturity) (e.g. 72%), Crop Coverage (e.g. 94% Coverage) and the pesticide that may be present on the fields crops (e.g. Biscaya). Additionally, meteorological data will be required in the form of Temperature, Wind and Precipitation Levels which can be queried against the available public meteorological data calculated from the geocoordinates provided or from in field sensors such as air flow sensors.

Input	Type	Notes
Date	Date	Format: yyyy-MM-dd
field_area	Float	Units: m^2
field_coordinates	Float Array	GeoJson Format
crop_type	String	The crop planted
crop_status	Float	Units: %
crop_coverage	Float	Units: %
temperature	Float	Units: Celsius
wind	Float	Units: Km/h
precipitation	Float	Units: mm

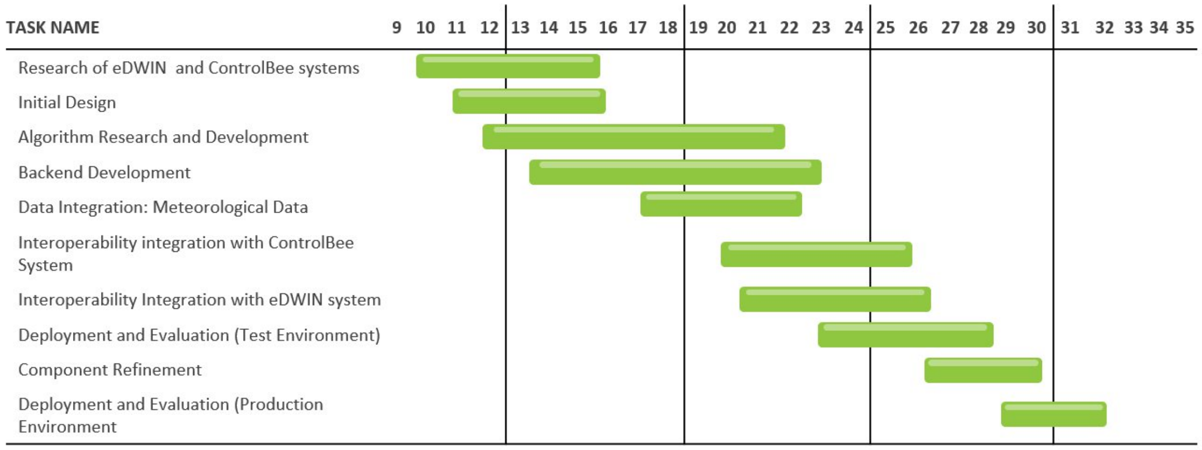
Table 9: Component A5 Input Data

Output: The component will output an estimate of the number of beehives required to fully pollinate a queried field. E.g. 4.3 Hives Required.

Related Generic Enablers: Security Protection, Data Integration, Data Management, Data Fusion.

Status: To be produced by DEMETER.

Development Plan:



Responsible Partner (name and organisation): Ross Campbell (ICE).

Partners involved: Poznan Supercomputing and Networking Centre (PSNC), Information Catalyst for Enterprise (ICE).

API:

<u>Calls</u>

- *POST /pollination_requirements/create_field: Create a virtual representation of a field with Field Size & Location and Field Contents received as input object. Returns the field object, including a created UUID of the object.*
- *PUT /pollination_requirements/{field_uuid}/update_field: update the variables of a previously created field, such as crop type and status.*
- *POST /pollination_requirements/{field_uuid}/predict: receive UUID of field previously created and query field data against publicly available meteorological data for this field / area. An estimate of the number of bees / hives required to pollinate the field is returned.*

<u>Objects</u>

- "field" is an object with the following variables:
  - date: date of the field observation
  - field_area: A numeric stating the size of the field in m^2
  - field_coordinates: Either 1 central coordinate can be provided if the field area attribute is not null or an array of field boundary coordinates can be provided (>= 3) from which the field_area can be calculated. Coordinates will be required in GeoJson Format.
  - crop_type: the crop planted in the field
  - crop_status: A numeric stating the maturity of the field as a percentage e.g. 34%
  - crop_coverage: A number static the crop coverage of the field e.g. 92%
  - pesticides: A string array of the pesticides in use within the field
- "Create field response" is the response received when using the create filed call.
- "result" is the response received when using the predict call:
  - Estimated Pollination Requirements: 4.3 beehives

Pilots involved: 5.3

5.2 *DSS AREA: B - Irrigation Management*

5.2.1 Component B.1 Water Balance Model

This component describes a crop water balance model that estimates crop soil moisture and the irrigation requirements of crops.

AGRICOLUS (pilot 3.1 DSS to support olive growers) and DNET have already developed a crop-soil water balance model. This component is therefore a data model which can be adopted by other pilots dealing with irrigation management.

What: A component that estimates crop-soil moisture/salinity and crop daily irrigation needs during the season. This component can be divided then in three sub-components which implement different models that also use data/information fusion.

These sub-components are specialised in the following models: (1) a model that estimates crop-soil moisture for actual crop in a parcel, (2) a model that estimates crop-soil salinity for actual crop in a parcel, and (3) a model that estimates irrigation requirements for actual crop in a parcel.

Input:

Input Data: Crop-Soil Moisture model	
Field data	Coordinates and geometry
Local sensing: crop-soil data	Soil type and texture. In-field crop-soil moisture sensors data
Remote sensing: imagery data (optional)	UAV/Sentinel-2 satellite spectral information for chlorophyll-sensitive vegetation indices.
Dates (optional)	First and end dates, in local time, for a time series result. If no dates are provided, the last and most recent available data is then considered and only one data element is returned.
Input Data: Crop-Soil Salinity model	
Field data	Coordinates and geometry
Local sensing: crop-soil data	Soil type and texture In-field crop-soil salinity sensors data
Remote sensing: imagery data (optional)	UAV/Sentinel-2 satellite spectral information for chlorophyll-sensitive vegetation indices.
Dates (optional)	First and end dates, in local time, for a time series result. If no dates are provided, the last and most recent available data is then considered and only one data element is returned.
Input Data: Irrigation	
Field data	Coordinates and geometry
Crop info	Crop Kc, traits, date of sowing/transplanting
Local sensing: crop-soil data	Soil type and texture, FC, PWP. In-field crop-soil salinity sensors data In-field crop-soil moisture sensors data
Local sensing: crop-plants data	In crop-plants sensors: Dendrometers, flow of sap data. In crop-field: evapotranspiration (eddy covariance) data.
Estimated data:	Crop-soil moisture estimation data Crop-soil salinity estimation data Crop-plant water status estimation data
Remote sensing: imagery data (data)	UAV/Sentinel-2 satellite spectral information for chlorophyll-sensitive vegetation indices.
Weather data	Data from Weather Services or in-field weather stations.

	Air temperature, air RH, wind, sun radiation, leaf wetness. Weather prevision for temperature and rain.

Table 10: Component B1 Input Data

Output:

Output Data: Crop-Soil Moisture model	
Time series with moisture values	If there are crop-soil moisture sensors available in the parcel. Array of values: • d: local time date of the data. • v: moisture value.
Time series with field images	If there are images available for the parcel with information related to soil moisture values. Array of images: • type: index (RGB, infrared, etc.) represented in the image. • d: local time date of the image. • img: image data with colours corresponding to the information index type.
Output Data: Crop-Soil Salinity model	
Time series with salinity values	If there are crop-soil salinity sensors available in the parcel. Array of values: • d: local time date of the data. • v: salinity value.
Time series with field images	If there are images available for the parcel with information related to soil salinity values. Array of images: • type: index (RGB, infrared, etc.) represented in the image. • d: local time date of the image. • img: image data with colours corresponding to the information index type.
Output Data: Irrigation	
Date	Local time for the irrigation requirement estimation
irrigate	'true' if irrigation must go, 'false' if it is not needed or must stop.
volume	Estimation of the volume of water needed for the irrigation
images (optional)	If there are images available for the parcel with information related to field plants water status. Array of images: • type: index (RGB, infrared, etc.) represented in the image. • d: local time date of the image. • img: image data with colours corresponding to the information index type.

Table 11: Component B1 Output Data

Related Generic Enablers:

- WP2: IAM data model.
- WP3: interoperability and security.
- WP4: Adaptive Visualisations for Dashboards.

Status: In development. Needs upgrades/modifications to match DEMETER needs and some components to be produced by DEMETER (i.e. Adaptive Visualisations for Dashboards). All models have two status, one for an initial basic model (parametric, agronomic-formulas) and a second for a more advanced one (integrating imagery, ML).

Development Plan: A common development plan can be seen within component B.2.

Responsible Partner (name and organisation): Manuel Mora (UMU).

Partners involved:

- Sub-component B.1.1 (Moisture): AGRICOLUS, INESCTEC, INIAV, OdinS, TRAGSA.
- Sub-component B.1.2 (Salinity): AGRICOLUS, ELGO, INESCTEC, INIAV, OdinS, TRAGSA.
- Sub-component B.1.3 (Irrigation): AGRICOLUS, ELGO, INESCTEC, INIAV, OdinS, TRAGSA.

API:

Calls

All the information (sensors, geometry, field data, crop, etc.) is linked to every parcel in the system and so only the parcel ID should be necessary in an API function call to access it.

- Sub-component B.1.1 (Moisture):
    - *POST /analisis_agro/soil_moisture/{query_obj}: This API will retrieve in a time series, with one or more data, the estimating crop-soil moisture values, and optionally coloured images of the parcel for sectored moisture values. If no dates for a time series are provided in the call, the last and most recent available data is then considered.*
- Sub-component B.1.2 (Salinity):
    - *POST /analisis_agro/soil_salinity/{query_obj}: This API will retrieve in a time series, with one or more data, the estimating crop-soil salinity values, and optionally coloured images of the parcel for sectored salinity values. If no dates for a time series are provided in the call, the last and most recent available data is then considered.*
- Sub-component B.1.3 (Irrigation):
    - *POST /analisis_agro/irrigation/{query_obj}: retrieving needed available information of a parcel, it estimates irrigation requirement as values and optionally as a coloured image of the parcel for sectored irrigation needs.*

Objects

- Sub-component B.1.1 (Moisture):
    - query_obj: information for API functions call:
        - parcel_id: [alphanumeric], identification of a parcel in the system.
        - ground_level: [numeric], the height/level of the sensors (i.e. multi-level/height probes).
        - images: [Boolean], set ‘true’ to also include time series with images as a result.
        - date_from: [alphanumeric], local time ‘from-date’ to get data time series.
        - date_to: [alphanumeric], local time ‘to-date’ to get data time series.
    - result_obj_ts: returned result as a time series of one or more elements:
        - v_series: if available, in-field sensors moisture information [{d: [alphanumeric], local time date of the value; v: [numeric], moisture}]
        - i_series: if available, field images with representative information of soil moisture in the parcel. [{type: [alphanumeric], id of the index represented in the image; d: [alphanumeric], local time date of the image; img: [raw], image data with colours corresponding to the different moisture values}]
- Sub-component B.1.2 (Salinity):

  - query_obj: information for the API call:
    - parcel_id: [alphanumeric], global identification of a parcel in the system.
    - ground_level: [numeric], the height/level of the sensors (i.e. multi-level/height probes).
    - images: [Boolean], set 'true' to also include time series with images as a result.
    - date_from: [alphanumeric], local time 'from-date' to get data time series.
    - date_to: [alphanumeric], local time 'to-date' to get data time series.
  - result_obj_ts: returned result:
    - v_series: if available, in-field sensors salinity information [{d: [alphanumeric], local time date of the value; v: [numeric], salinity}].
    - image_series: if available, field images with representative information of soil salinity in the parcel [{type: [alphanumeric], id of the index represented in the image; d: [alphanumeric], local time date of the image; img: [raw], image data with colours corresponding to the different salinity values}].
- Sub-component B.1.3 (Irrigation):
  - query_obj: information for both API functions calls:
    - parcel_id: [alphanumeric], global identification of a parcel in the system.
    - ground_level: [numeric], the height/level of the sensors (i.e. multi-level/height probes).
    - images: [Boolean], true to also include coloured images for sectored irrigation needs.
  - return_obj: returned result:
    - d: [alphanumeric], local time date.
    - irrigate: [Boolean], true if irrigation must go, false if it is not needed or must stop.
    - v: [numeric], if available, estimation of the volume of water needed for the irrigation.
    - images: if available, {type: [alphanumeric], id of the index represented in the image; d: [alphanumeric], local time date of the image; img: [raw], image data with colours corresponding to different sectors irrigation needs}].

Pilots involved:

- Sub-component B.1.1 (Moisture): 1.1 & 1.2 (TRAGSA & OdinS), 3.1 (AGRICOLUS), 3.2 (INESCTEC, INIAV).
- Sub-component B.1.2 (Salinity): 1.1 & 1.2 (TRAGSA & OdinS), 1.3 (ELGO), 3.1 (AGRICOLUS), 3.2 (INESCTEC, INIAV).
- Sub-component B.1.3 (Irrigation): 1.1 & 1.2 (TRAGSA & OdinS), 1.3(ELGO), 3.1 (AGRICOLUS), 3.2 (INESCTEC, INIAV).

5.2.2 Component B.2 Data Fusion for Irrigation

A promising approach in crop water management is the integration of several technologies, providing all the important information needed to support farmers in daily decision making on irrigation management, combining data to evaluate the current status of the crop and assess the irrigation needs.

What: A component that combines data from different sources (water balance model), IoT sensors and remotely sensed data to estimate current soil moisture and plant water status. This component can then be divided in two sub-components. The first one, about estimating soil moisture, is developed in component B.1 and, as already mentioned, it also uses data/information fusion. Irrigation DSS can be composed then as an information fusion final solution with all sub-components defined in components B.1 and B.2. The second sub-component is a model that estimates plant water status. Here we consider two factors: the crop water pressure and the crop water stress index (CWSI). Data and information fusion are used as different sources are needed for the model.

Input:

Input Data: Plant Water Status	
Field data	Coordinates and geometry
Crop info	Crop ET_0, Kc
Local sensing: crop-soil data	Soil type, texture In-field crop-soil salinity sensors data In-field crop-soil moisture sensors data
Local sensing: crop-plants data	In plants sensors: Dendrometers, flow of sap data. In field: evapotranspiration (eddy covariance) data.
Remote sensing: imagery data (optional)	UAV/Sentinel-2 satellite spectral information for chlorophyll-sensitive vegetation indices.
Weather data	Data from Weather Services or in-field weather stations. Air temperature, air RH, wind, sun radiation, leaf wetness.

Table 12: Component B2.2 (Plant Water Status) Input Data

Output:

Output Data: Irrigation	
Date	Local time for the values
wp	Estimation of the plant water pressure.
cwsi	Estimation of the plant water stress index (CWSI).
images (optional)	If there are images available for the parcel with information related to field plants water status. Images: (in the same date but may be of different types) • type: index (RGB, infrared, etc.) represented in the image. • d: local time date of the image. • img: image data with colours corresponding to the information index type.

Table 13: Component B2.2 (Plant Water Status) Output Data

Related Generic Enablers:

- WP2: IAM data model.
- WP3: Interoperability and security.
- WP4: Adaptive Visualisations for Dashboards

Status: In development. Needs upgrades/modifications to match DEMETER needs and some components to be produced by DEMETER (i.e. Adaptive Visualisations for Dashboards). All models have two status, one for an initial basic model (parametric, agronomic formulas) and a second for a more advanced one (integrating imagery, ML).

API:

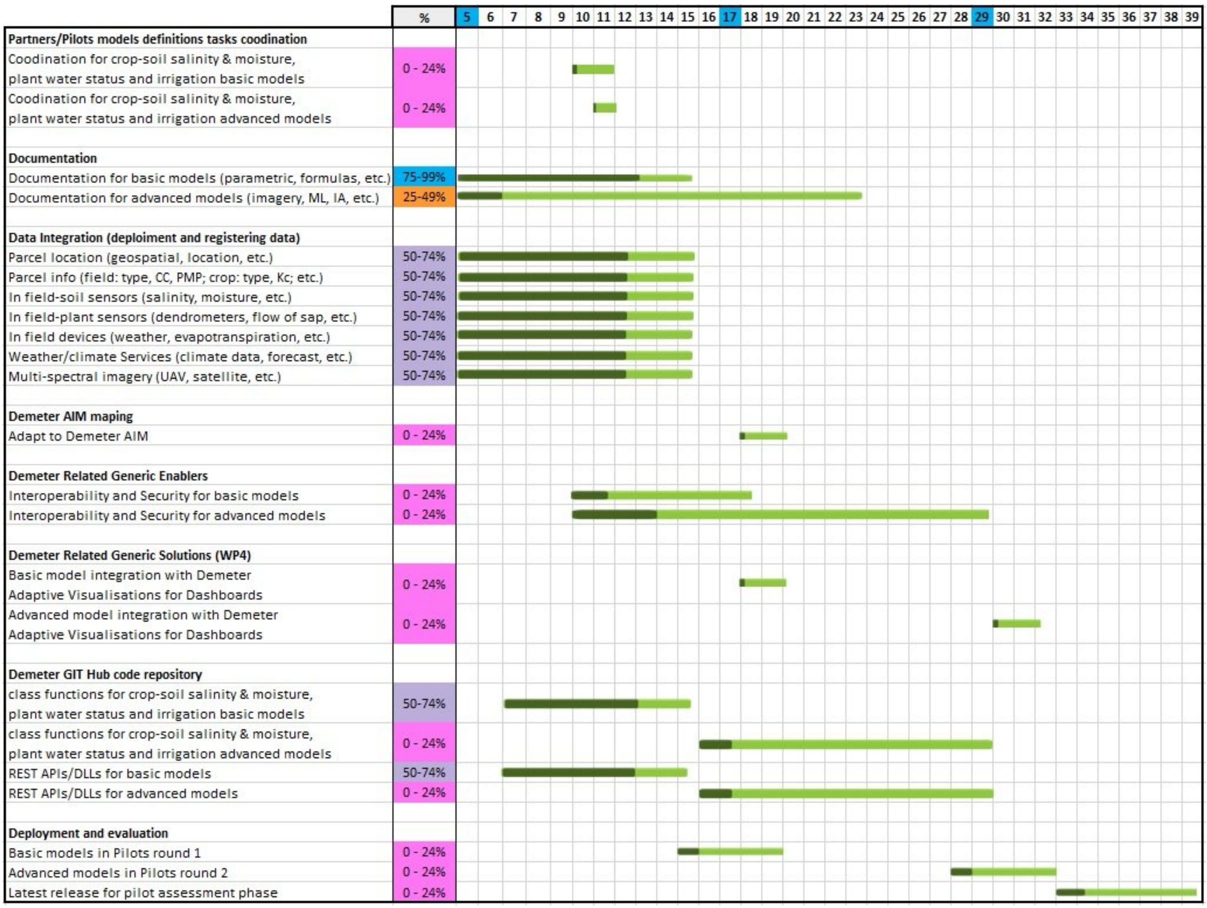
<u>Calls:</u>

- *POST /analisis_agro/plant_status/{query_obj}: retrieves information about a parcel to estimate plants water status as water pressure and water stress index (CWSI). Also, coloured images of the parcel are optionally obtained showing this information.*

<u>Objects:</u>

- query_obj: information for the API function call:
  - parcel_id: [alphanumeric], global identification of a parcel in the system.
  - image: [Boolean], true to also include coloured images for crop plants water status.
- return_obj: returned result:
  - wp: [numeric], value of the estimation of the plant water pressure.
  - ws: [numeric], value of the estimation of the plant CWSI.
  - images: if available, [{type: [alphanumeric], id of the index represented in the image; d: [alphanumeric], local time date of the image; img: [raw], image data with colours corresponding to the plant water status index}].

Development Plan: Common development plan integrating the sub-components from B.1 and B.2.



	%
Partners/Pilots models definitions tasks coodination	
Coodination for crop-soil salinity & moisture, plant water status and irrigation basic models	0 - 24%
Coodination for crop-soil salinity & moisture, plant water status and irrigation advanced models	0 - 24%
Documentation	
Documentation for basic models (parametric, formulas, etc.)	75-99%
Documentation for advanced models (imagery, ML, IA, etc.)	25-49%
Data Integration (deploiment and registering data)	
Parcel location (geospatial, location, etc.)	50-74%
Parcel info (field: type, CC, PMP; crop: type, Kc; etc.)	50-74%
In field-soil sensors (salinity, moisture, etc.)	50-74%
In field-plant sensors (dendrometers, flow of sap, etc.)	50-74%
In field devices (weather, evapotranspiration, etc.)	50-74%
Weather/climate Services (climate data, forecast, etc.)	50-74%
Multi-spectral imagery (UAV, satellite, etc.)	50-74%
Demeter AIM maping	
Adapt to Demeter AIM	0 - 24%
Demeter Related Generic Enablers	
Interoperability and Security for basic models	0 - 24%
Interoperability and Security for advanced models	0 - 24%
Demeter Related Generic Solutions (WP4)	
Basic model integration with Demeter Adaptive Visualisations for Dashboards	0 - 24%
Advanced model integration with Demeter Adaptive Visualisations for Dashboards	0 - 24%
Demeter GIT Hub code repository	
class functions for crop-soil salinity & moisture, plant water status and irrigation basic models	50-74%
class functions for crop-soil salinity & moisture, plant water status and irrigation advanced models	0 - 24%
REST APIs/DLLs for basic models	50-74%
REST APIs/DLLs for advanced models	0 - 24%
Deployment and evaluation	
Basic models in Pilots round 1	0 - 24%
Advanced models in Pilots round 2	0 - 24%
Latest release for pilot assessment phase	0 - 24%

Responsible Partner (name and organisation): Manuel Mora (UMU).

Partners involved: TRAGSA, OdinS, ELGO, AGRICOLUS, INESCTEC, INIAV.

Pilots involved: 1.1 & 1.2 (TRAGSA & OdinS), 1.3 (ELGO), 3.1 (AGRICOLUS), 3.2 (INESCTEC, INIAV).

European Union
European Regional
Development Fund

5.3 *DSS AREA: C - Nutrition Management*

5.3.1 Component C.1 Nitrogen Balance Model

What: This component describes a crop nitrogen balance model that estimates crop nitrogen requirements and provides the scheduling of fertilization. AGRICOLUS (pilot 3.1 DSS to support olive growers) has developed a nitrogen balance model which is integrated in platform along with remotely sensed indices. This component is therefore a data model which can be used by other pilots dealing with fertilization management. SIVECO (pilot 1.4 DSS to support maize crops) will develop a nitrogen balance model which will be integrated in platform along with remotely sensed indices.

A component that estimates crop nitrogen needs and the crop fertilization scheduling during the season to optimize nitrogen fertilization, avoiding nitrogen excess.

Input: Farm data (geographic coordinates), crop traits, expected yield, date of sowing or transplanting for annual crop, plant density and pruning system for tree crops, soil texture, soil moisture, agrometeorological data (minimum and maximum temperature, rainfall), remotely sensed indices (e.g. NDVI from Proba–V satellite). Some of the data will be taken as farmer input, and farmers will enter data such as: type of crop, different treatments administered and the state of vegetation. Data taken from farmers will be correlated with satellite images.

Farmer Input Data	
Field coordinates and geometry	Mandatory
Crop info	Crop type: Mandatory Crop phenology: Mandatory
Soil data	Soil type: Optional
Previous fertilization data	Previous fertilization date: Mandatory Previous fertilization substance: Mandatory Previous fertilization quantity: Mandatory
System input data	
Remote Sensing Multi-temporal Imagery data	• Default: Sentinel-2 spectral information • Proba-V NDVI
Meteorological data	• Default: Data from Satellite (ECMWF ERA5, Copernicus, Darksky) • National or regional data and predictions • Local meteo station in the field (soil moisture)

Table 14: Component C1 Input Data

Output: Deliverables to farmers will consist of cartographic plans with recommendations for differentiated fertilization in the same plot, to obtain a higher yield. At the same time, cartographic plans will be made after the administration of fertilizing substances to diagnose the culture as well as possible. Thus, farmers can receive information such as: crop nitrogen status, fertilization requirements and scheduling during phenological development with different results in each section of the field, nitrogen prescription maps. At the same time, the fertilization recommendations will be made on and based on the sensors mounted in the field that transmit data regarding the temperature and humidity in the soil and air.

Another important aspect is that the agrometeorological stations also transmit the meteorological forecast and thus the optimal moment of fertilizer distribution will be determined.

Output Data

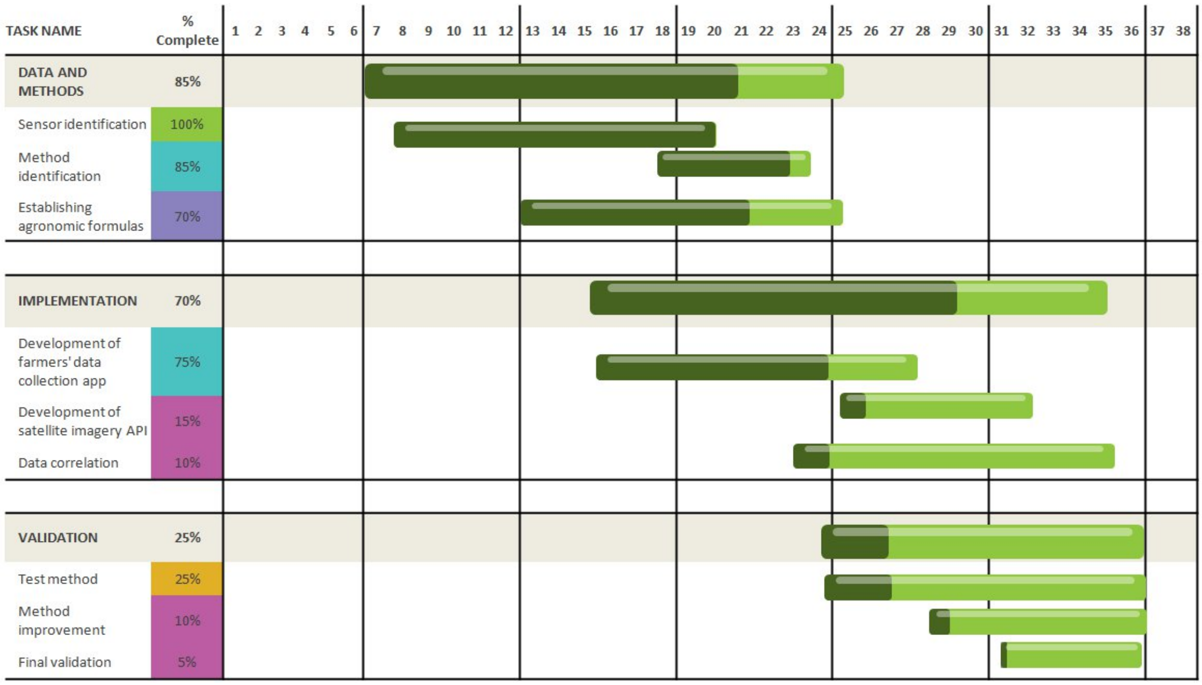
Area	Recommended zone for fertilization
Substance	Recommended active substance for fertilization (e.g. Nitrogen)
Period	Recommended period for applying the fertilization
Quantity	Recommended quantity of substance to be applied for the specified area

Table 15: Component C1 Output Data

Related Generic Enablers: Security Enabler, Semantic Enabler.

Status: To be produced by DEMETER.

Development Plan:



Responsible Partner (name and organisation): Component pending to be assigned

Partners involved: AGRICOLUS, UBIWHERE, SIVECO and APPR, LESPROJEKT

API:

Calls

- *POST /nutrition_management/fertilization/recommend: recommends zone for fertilization, active substance to be applied, time period of applying and quantity of substance to be applied for the specified area.*

Objects

- Field_geometry: polygon (mandatory - M).
- crop_type: a value from a crop_type list of values (M).
- crop_phenology: a value from a predefined list of crop phenologies (M).
- soil_type: a value from a soil_type list of values (optional - O).
- previous_fertilization_date: date (M).
- previous_fertilization_substance: a value from a fertilization_substance list of values (M).
- previous_fertilization_quantity: numeric (M).

Pilots involved: 3.2, 1.4, 2.3, (1.3, 3.1)

5.3.2 Component C.2 Nutrient Monitor

What: Nutrient monitoring can be done most correctly once a year. These tests are performed directly on the ground by various methods: directly in the field or in the laboratory. However, in this project we do not intend to make soil analyses, but to make analyses on the response of the soil through plants. The analysis will be made based on the layers with nutrients following the agricultural mapping of the soil, correlated with the satellite images and the obtained production plans. When the soil has all the necessary nutrients, the corn crop will have a very high height and density. We aim to analyse satellite images on cultivated areas in each stage of vegetation.

Input: As input data can be used files such as .shp, resulted from agrochemical maps made by farmers, on the land before planting. Thus, the input data consist of agrochemical mapping layers superimposed over the satellite images. Data from agrometeorological stations can be correlated with the diagnosis of soil nutrients. (E.g. if there was high rainfall, the soil is no longer very rich in nutrients).

Farmer Input Data	
Field coordinates, geometry, and agrochemical layers	Mandatory (shp)
Historical crop data	Crop rotation: Mandatory Previous crop traits: Mandatory
Soil data	Soil type: Mandatory Elevation: Optional
System input data	
Remote Sensing Multi-temporal Imagery data	• Default: Landsat7, Sentinel-2 spectral information • Proba-V
Meteorological data	• Default: Data from Satellite (ECMWF ERA5, Copernicus, Darksky) • National or regional data and predictions • National or regional historical data • Local meteo station in the field (soil moisture)

Table 16: Component C2 Input Data

Output: Following the analysis performed on the evolution of the maize crop during at least two whole vegetation periods, it will be possible to make planting recommendations for the next year. For example, in the best areas of the plot to sow a larger amount of seeds. Thus, in the next crop, several densities will be sown in the same plot to obtain higher yields. At the same time, depending on the soil nutrients, future soil fertilization recommendations can be made.

It is recommended that the result should be exported in a .shp file, compatible with the GPS of the agricultural machine, which will contain the favourable areas of the plot.

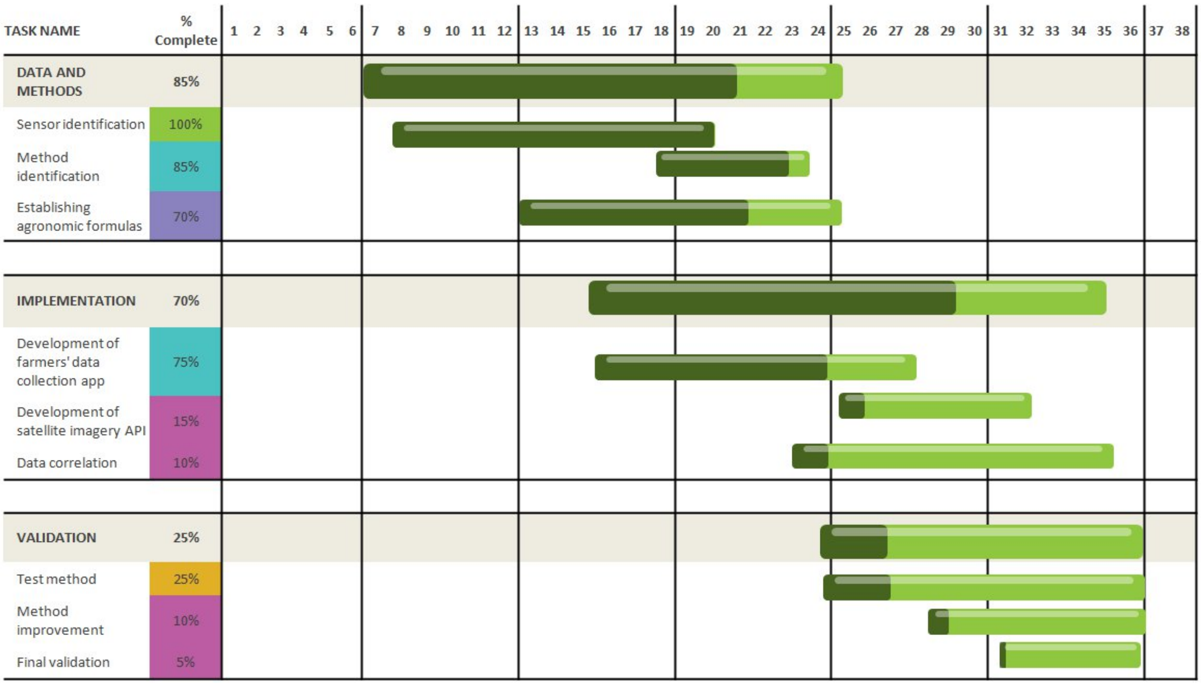
Output Data	
Area	Recommended zone for fertilization
Substance	Recommended active substance for nutrition (e.g. NPK)
Period	Recommended period for sowing
Density	Recommended seeds density for the area

Table 17: Component C2 Output Data

Related Generic Enablers: Security Enabler, Semantic Enabler.

Status: To be produced by DEMETER.

Development Plan: First, the existing agrochemical mapping plans will be obtained, then satellite images will be taken on the same area and the development of the plants will be followed.



Responsible Partner (name and organisation): Component pending to be assigned

Partners involved: AGRICOLUS, UBIWHERE, SIVECO and APPR, LESPROJEKT

API:

Calls

- *POST /nutrition_management/nutrition/recommend: recommends zone for sowing and nutrient applying, time period of sowing and quantity of substance to be applied for the specified area*

Objects

- agrochemical_layers: polygon (shp) (mandatory)
- crop_rotation: a value from a crop_rotation list of values (mandatory)
- crop_traits: a value from a crop_traits list of values (mandatory)
- soil_type: a value from a soil_type list of values (mandatory)
- elevation: numeric (optional)

Pilots involved: 3.2, 1.4, 2.3, (1.3, 3.1)

5.4 *DSS AREA: D - Machinery and Field Operations*

5.4.1 Component D.1 Emission

What: Demonstration of on-board sensors for in service monitoring and testing the applicability as alternative to Portable Emission Measurement Systems (PEMS) while considering aspects of data management.

Input:

Input Data	
Internal	Engine data from tractor on farms
	Test bench data
	Emission data of tractors
External	EU/national emission regulation

Table 18: Component D1 Input Data

Output:

Output Data	
Emission data	
data/status of different engine parameters	

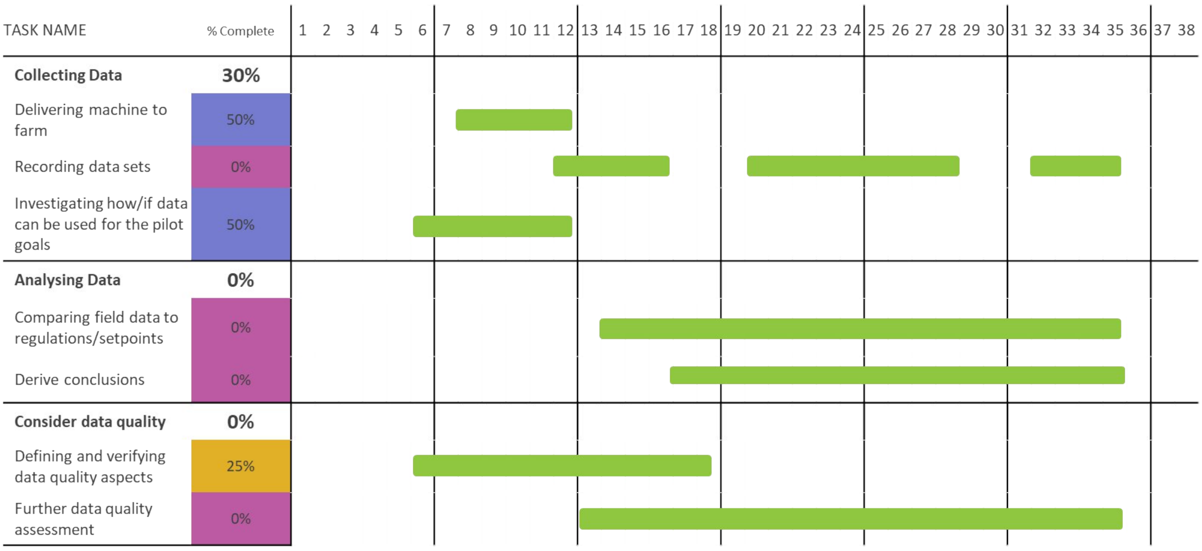
Table 19: Component D1 Output Data

Related Generic Enablers: Data Quality Assessment (WP2 T2.3 component).

Status: Under development; following activities has been done:

- Implementation of test hardware in machine.
- Definition of boundary conditions for data quality assessment.
- Initial data quality assessment.
- First measurement of sensor data in idle and analysis of recorded files.
- Started preparation to deliver machine to farm.

Development Plan:



Responsible Partner (name and organisation): Andreas Schroeder (JD)

Partners involved: John Deere, Fraunhofer IESE

API: The technical implementation itself will get more elaborated once a first set of data (from the operations on the field) is collected (expected by August 2020). As this real environment will have an impact on the technical constraints and the analysis, information about APIs cannot be provided.

Pilots involved: 2.1

5.4.2 Component D.2 Field Operation

What: The Farm data brokerage establishes a trust-based and compliant data market for agricultural enterprise data that sits between the owners and operators of agricultural data Clouds and the farmer. This data market will consist of both a technical platform and advisory services that will ensure the easy adoption of data and technology by farmers.

These agricultural enterprise data, exactly from area D – Machinery can be:

- Telemetric data (position, speed, consumption, type of work, description of an operator, etc.) all in time and position.
- Machine CAN data (e.g.: setting, performance, machine error messages, operating data: temperatures, pressures, levels, control units, software versions, etc.).
- Machine and implement ISO Bus data (speed, performance, application data).
- Definition of used land units (area, boundaries, owner, historical data, yields, trajectories of machine movement in the past).
- Current and old meteorological data needed for machine operation – an example for sprayers.

Input: Our current farmers for demonstration in our pilot do not have any information from category D – Machinery.

Input Data	
Vehicle ID	Unique id of vehicle
Position	GPS position of vehicle (if available)
Speed	Speed of vehicle
Temperature	Vehicle temperature
Breaking	Breaking
Meteorological data	Array of historical data
Fuel consumption	
Operator ID	
Field data	GPS coordinates and other relevant data

Table 20: Component D2 Input Data

Output:

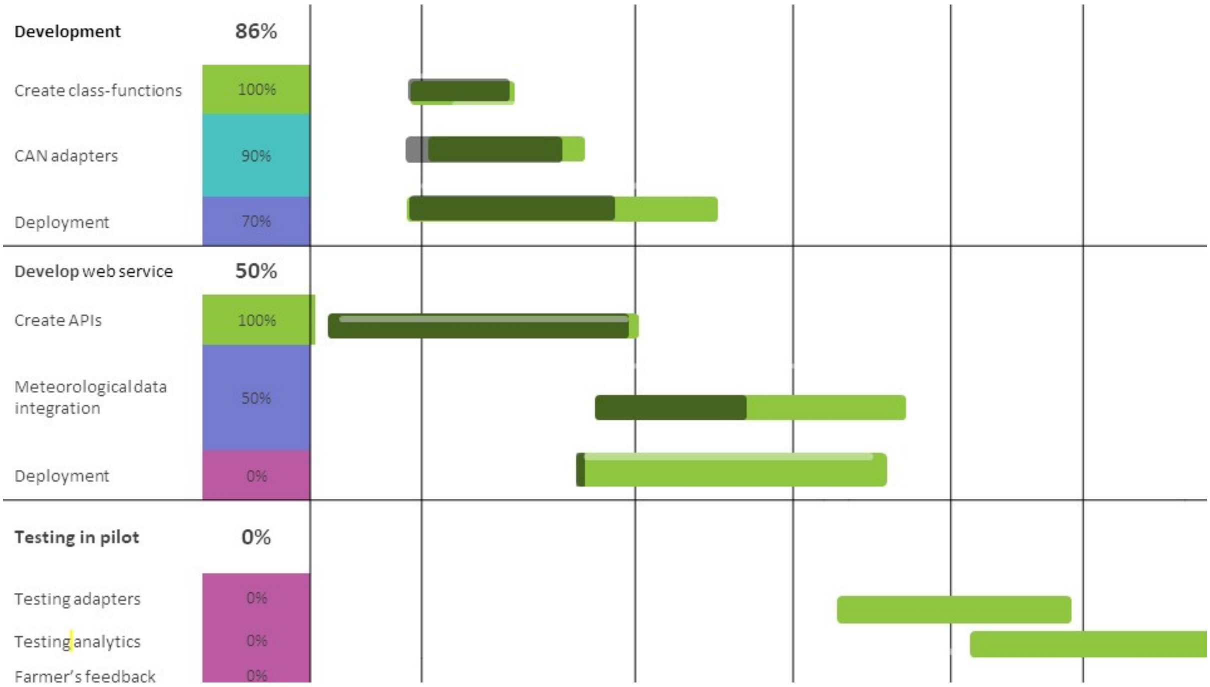
Output Data	
Vehicle ID	Unique id of vehicle
Driver behaviour	Behaviour based on breakage, fuel consumption
Machine trajectory	Analyse of duration, performance, and other stats
Vehicle performance	Data and stats about vehicle performance

Table 21: Component D2 Output Data

Related Generic Enablers: Security Enabler, Semantic Enabler.

Status: In development.

Development Plan:



Responsible Partner (name and organisation): Nenad Gligoric (DNET)

Partners involved: DNET, Plantaze, UDG, LESPROJEKT, WODR, AVINET, PSNC

API:

<u>Calls:</u>

- *POST /field_operations/can/{vehicle_uuid}: Parameters from CAN device.*
- *POST /field_operations/meteo/{area_id}: environmental parameters.*

<u>Objects</u>

- “input dataset” is an object with the following variables:
  - Vehicle ID
  - position
  - Speed
  - Temperature
  - Breaking
  - Meteorological data
  - Fuel consumption
  - Operator ID
  - Field data
- Output shapefile:
  - Vehicle ID
  - Driver behaviour
  - Machine trajectory
  - Vehicle performance

Pilots involved: 2.3, 5.1, 5.2

5.4.3 Component D.3 Variable Rate

What: Based on a raster image of biophysical parameters, i.e. NDVI or FAPAR, derived from Sentinel-2 satellite images, a farmer can request a shape-file for their field where NDVI/FAPAR pixel values are translated into a specific rate for variable rate fertilization.

Input: FAPAR satellite image for field.

Input Data Arable Crops	
Field coordinates and geometry	
Date	Date of the Satellite image to use for variable rate calculation; defaults to 'today'; if no image is available for that specific day, the last image before this date will be used.
Remote Sensing Imagery data	Sentinel-2 spectral information and derived biophysical parameters (NDVI or FAPAR)
Base dose	Fertilizer dose that the farmer would normally apply on the field
Strategy	For N-fertilization, the farmer can choose between two strategies, either applying a higher dose of fertilizer to zones with higher FAPAR values or applying a lower dose to these zones
Number of zones	Split up the field into N zones, where the zone boundaries will be calculated based on intra-field variability of FAPAR or NDVI values
Environment data	Data from deployed sensors
Prediction model	Pest/Disease prediction model

Table 22: Component D3 Input Data

Output:

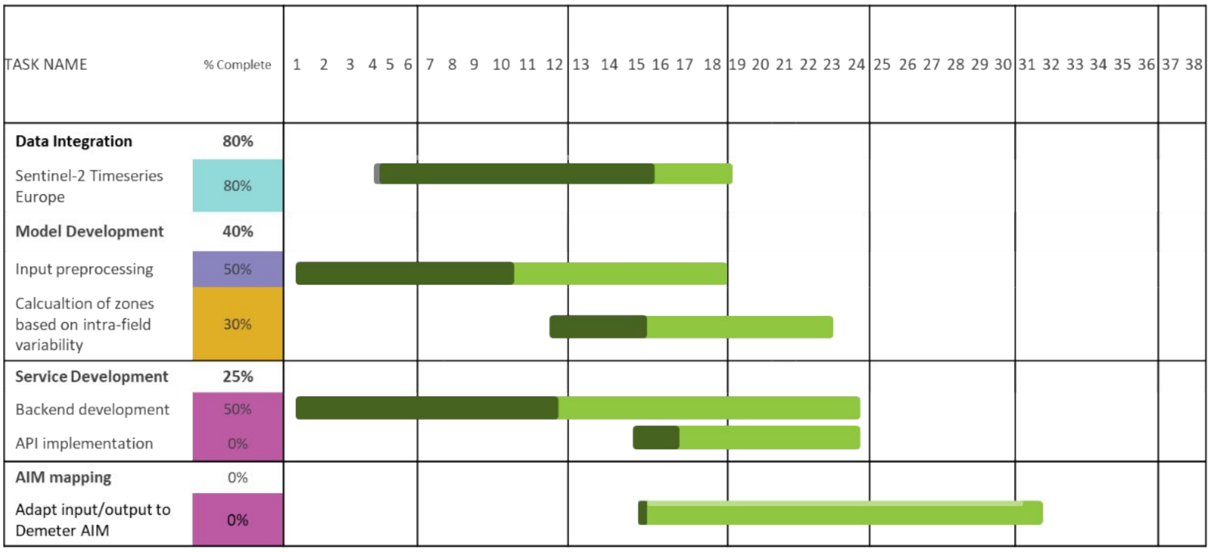
- A shapefile with for each zone the dose to be applied.
- Date of the Sentinel-2 input data used.
- Spraying order.
- Instructions for activities to be taken.

Related Generic Enablers:

- Semantic interoperability.
- Standardized interfaces for data retrieval from various sources:
    - Data security.
    - Data ownership.
    - Access rights.

Status: Basic building blocks are available, but workflow needs upgrades/modifications to be applicable European wide, and to match DEMETER needs.

Development Plan:



TASK NAME	% Complete
Data Integration	**80%**
Sentinel-2 Timeseries Europe	80%
Model Development	**40%**
Input preprocessing	50%
Calcualtion of zones based on intra-field variability	30%
Service Development	**25%**
Backend development	50%
API implementation	0%
AIM mapping	0%
Adapt input/output to Demeter AIM	0%

Responsible Partner (name and organisation): Isabelle Piccard (VITO)

Partners involved: VITO, John Deere, DNET, FEDE

API:

Calls

- *POST /variable_rate/get_variable_rate: get a shapefile with variable rate zones for the input dataset.*
- *POST /variable_rate/environment/{area_id}: Post environment data.*
- *POST /variable_rate/instructions/{area_id}: Spraying instructions.*
- *GET /variable_rate/getarchivelistnotifications: List of archived notifications for user.*
- *POST /api/addirrigationtofieldsregister: Add irrigation to fields register.*

Objects

- “input dataset” is an object with the following variables:
  - field geometry (polygon).
  - Date: date of the Sentinel-2 image to use.
  - Base dose: standard dose if variable rate is not applied.
  - Strategy: ‘positive’ or ‘negative’.
  - Number of zones.
- Output shapefile:
  - Shapefile of the field, where the field is split up in N management zones and each management zone has a specific rate.

Pilots involved: 3.4, 5.1, 5.2

5.5 *DSS AREA: E - Pest and Disease Management*

5.5.1 Component E.1 Computer vision-based counting module

What: A component that counts the number of appearances of a given element on an image using a model trained with similar images. The model requires a high number of images (with similar characteristics to the images to be used as prediction input), properly labelled to learn the nature of the elements to count. The results of the component might be impacted by different factors such as:

number of training images, similarity of training images to the prediction ones, quality of the images, quality of the labelling, etc.

Input: A set of images containing the elements to be identified properly labelled to be used as a training set and a set of unlabelled images to count the elements from the training set. The number of labelled images for training purposes should be high enough for ML application. Once the model is trained, the input will be the images to be used to estimate the number of elements to be counted.

Input	Type	Notes
Training image	Image file	An image to train the model. This image must be like the images used in the component, containing the elements to be counted. Supported image formats still to be defined (.jpg, .png, etc.). Each training image must be provided with its corresponding "Training image labelling" file.
Training image labelling	Annotation file	A file that defines the areas of the "Training image" where is located each one of the elements to be counted. Supported image labelling formats still to be defined (COCO, Pascal VOC, etc.).
Test image	Image file	An image file containing the elements to be counted. This image is to be provided without any labelling. The system will provide an estimation of the number of elements to be counted that have been identified by the model. Supported image formats still to be defined (.jpg, .png, etc.).
Model creation parameters	String	The model may support some configuration parameters at its creation: name of the objects to be counted, model configuration, model evaluation criteria, etc.

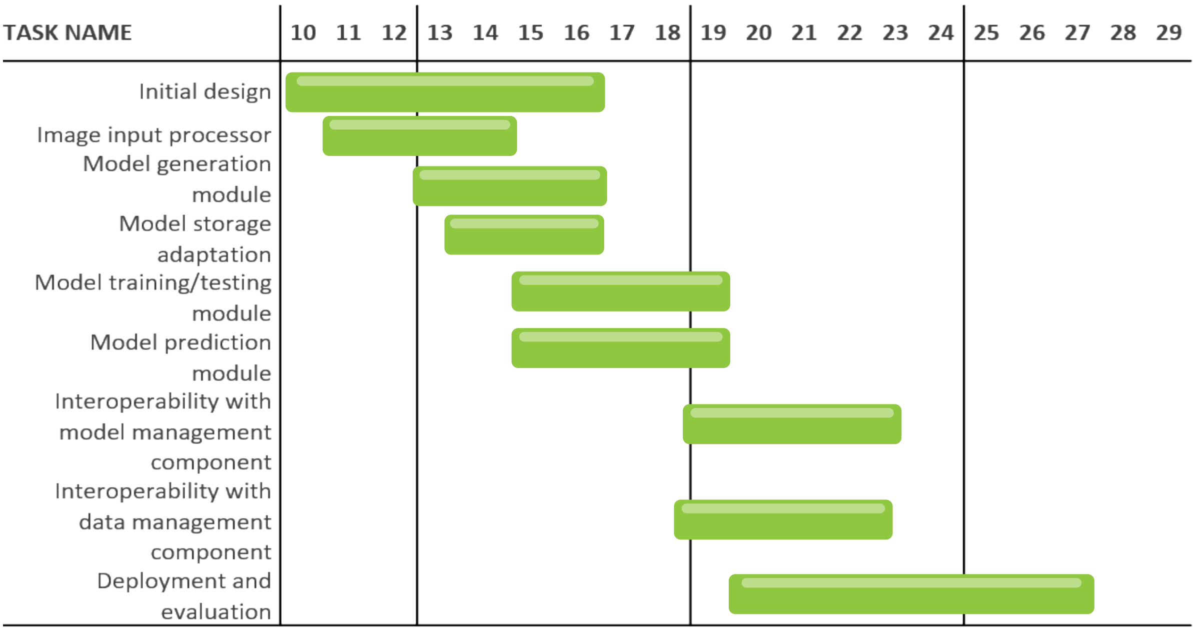
Table 23: Component E1 Input Data

Output: An estimation of the number of elements to be counted for each test image given to the model.

Related Generic Enablers: Model storage and management, Targeted data fusion, Data Management.

Status: To be produced by DEMETER (awaiting data dependencies).

Development Plan: This development plan might suffer changes related to data availability.



Partners involved: Atos

Responsible Partner (name and organisation): Sergio Salmeron (ATOS)

API:

<u>Calls (still under design):</u>

- */CreateModel: Creates an initial instance of a model from a set of parameters and stores it in the model storage and management component, generating a unique identifier for that model (model ID).*
- */UploadTrainingData: Uploads and stores an image and its labelling assigned to a model ID.*
- */TrainModel: Trains the model with the labelled images in the database that are assigned to it. Then shows the training evaluation results.*
- */GetPredicion: Given an image and a model ID, the model generates an estimation of the number of elements to be counted that appear in the image.*

<u>Objects (Still under design):</u>

- Image: An image file containing the elements to be identified. Supported formats still to be defined (jpg, png, etc).
- Image Labelling: A file with the definition of the areas where each component to be counted appears. Supported format and labelling style still to be defined (COCO, Pascal VOC, etc.).

Pilots involved: 1.1 & 1.2, 3.3, 5.1, 5.3

5.5.2 Component E.2 Estimate temperature-related pest events

The structure of the present component, currently under active development and validation, will be like the one described for the component in A.3 Plant Phenology estimation, on which it is being modelled. Essentially, the problem of predicting the temporal location of pest peaks is parallel in broad methodological terms to the one with phenology: we have an observed date with a maximum peak of presence of the pest (or initial presence of the pest) and we try to use it to build a model based on day degree to estimate the peak or the initial flight. In this sense, the data structure is like the one for phenology, but it is used with a different target.

What: A component that estimates the date and the extension of a pest event.

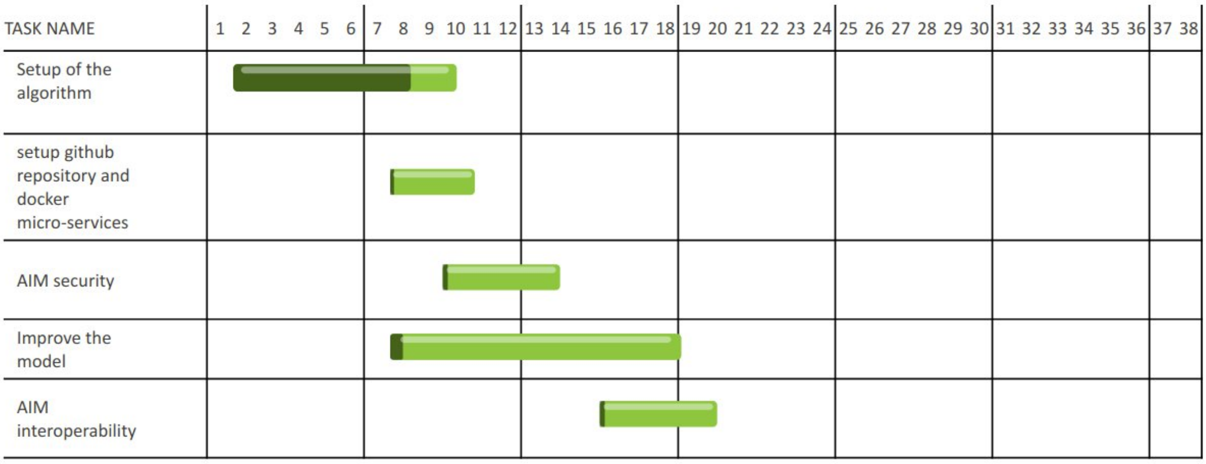
Input: Pest observations, containing for each observation the date (day of the year) and the pest catches in traps (according to a quantitative ordinal scale at least), and daily minimum and maximum temperature for growing degree day calculation, starting from the first day of the year. In general terms, ambient temperature is one of the important factors in the development of pest events. For this purpose, ambient temperature and date are among the raw data available. As for A.3 Plant Phenology Estimation, ambient temperature is transformed into GDD and the date is transformed into day-of-year (DOY).

Output: An estimation of the date at which the main pest phases occur.

Related Generic Enablers: The Security Enabler will be adopted to access the needed resources; Enablers for Functional Interoperability Enablers for the interactions with the DEMETER provider. We will, in general, activate all the mandatory enablers.

Status: It exists a proof of concept developed using the same methodology adopted by A.3 component. We have defined the structure of the component API.

Development Plan:



Responsible Partner (name and organisation): Diego Guidotti (AGRICOLUS)

Partners involved: AGRICOLUS, TRAGSA, ICE

API: the basic structure will be developed in Python. The output will follow the Knowage specs for visualisation.

<u>Calls</u>

- *POST /e2_pest_climate/{pest_code}/create: create a space for a specific pest/environment.*
- *POST /e2_pest_climate/{pest_code}/train: receive an array of the "training input features" to train the models; the training in incremental the features will be added to the training set; return an object with the training result.*
- *POST /e2_pest_climate/{pest_code}/predict: receive an array of the "predict input features" and return an array of "prediction result".*

<u>Objects</u>

- "training input features" is an object with the following variables:
  - date: date of the field observation
  - lat/lon of the trap
  - trap_data: number of insects captured by the trap in that specific days divided by the number of days the traps has been exposed
  - weather: an array of weather data; each element contains
    - date: date of observation
    - tmin: minimum daily temperature
    - tmax: maximum daily temperature
- "training result":
  - rmse: interpolation error of the models
- "predict input features":
  - contains the same variable of "training input features" except the trap_data value
- "predict result":
  - date: date of the field observation
  - lat/lon of the trap
  - estimated_trap: value of the estimated catches

Pilots involved: 3.1, 3.3

5.6 *DSS AREA: F - Animal Yield*

5.6.1 Component F.1 Estimate Milk Production

What: Milk yield forecasting using data from milking robots in 620 dairy farms. The dairy farms will provide basic cow data (age, lactation no., days in milk, breed) and milk production (milking frequency, milk yield). Data are integrated in Mimiro's platform, which is built on AWS technology, architecture, and analytic platform (Lakeformation, Sagemaker). The robot data are generating 182 million data rows per year, which will be used in the development of the milk forecasting DSS. In the machine learning approach, we will evaluate several algorithms. One alternative is to use random forest algorithms, where use of decision tree is the basic building block. In the project we will also evaluate other algorithms which have been used in development of forecasting models. In the pilot we will test the following alternatives, which are available on the AWS Sagemaker. The AWS Forecast predictor uses an algorithm to train a model, then uses the model to make a forecast using an input dataset group. The following algorithms will be evaluated:

- Autoregressive Integrated Moving Average (ARIMA).
- DeepAR+ Algorithm.
- Exponential Smoothing (ETS) Algorithm.
- Non-Parametric Time Series (NPTS) Algorithm.
- Prophet Algorithm.

In addition, we will use the K-Means clustering algorithm to develop models for herd culling strategy.

Input: basic cow data (age, lactation no., days in milk, breed) information and milk production (milking frequency, milk yield). Data is available from 2018-2020. The data Will be divided in two subsets, across years, one for model training and one for model development and testing. Below there is an example of classification of input data used to predict individual cow milk prognosis.

Input	Type	Notes
Animal_Id	Integer	Cow id
BirthDate	Date	Format: DD/MM/YYYY
Breed	Integer	
Parturition date	Date	
DIM	Float	Stage of lactation
MilkSampleDate	Float	
Average Daily Production	Float	
Daily Fat, protein, lactose	Float	From inline milk sensor
Monthly milk fat, protein, lactose, urea, cell count	Float	Based on monthly milk samples. Laboratory analysis
Health status	Integer	Animal health on control day, no/yes
ConcentrateDate	DateTime	
ConcentrateAmount	Float	Measured concentrate feed intake
ConcentrateUnit	Integer	
ForageAmount	float	Predicted forage intake

Table 24: Component F1 Input Data

Output: Milk forecasting DSS. Below there is an example of classification of output data used to predict individual cow milk prognosis.

Input	Type	Notes
Animal_Id	Integer	Cow id

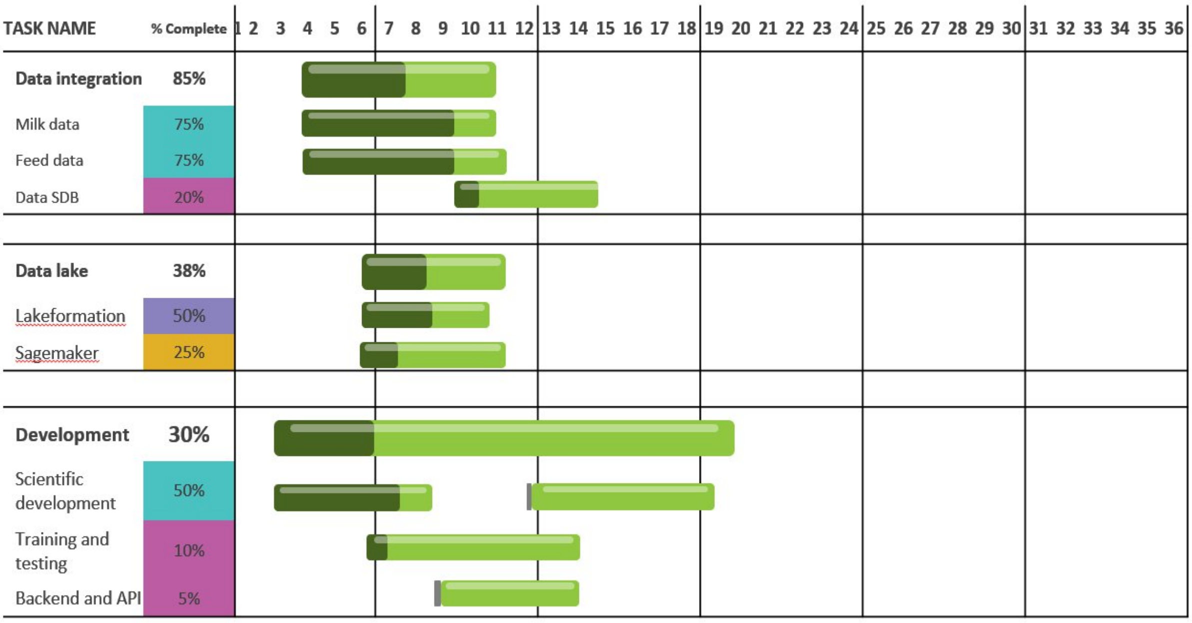
Breed	Integer	
Parturition date	Date	Format: DD/MM/YYYY
DIM	Float	Stage of lactation
MilkSampleDate	Float	
Average Daily Production	Float	
Daily Fat, protein, lactose	Float	From inline milk sensor
ConcentrateAmount	Float	Measured concentrate feed intake
ForageAmount	Float	Predicted forage intake
Predicted milk yield, n+1	Float	Individual milk forecast week n+1
Predicted milk yield, n+4	Float	Individual milk forecast week n+1
Predicted milk yield, n+N_n	Float	Individual milk forecast week N
Predicted milk yield	Float	Herd predicted milk yield week N
Animal culling value	Float	Predicted economical value of individual cows
Culling date	Data	Predicted data of culling

Table 25: Component F1 Output Data

Related Generic Enablers: Benchmarking dashboard, authentication, security.

Status: In development.

Development Plan:



Responsible Partner (name and organisation): Harald Volden (MIMIRO)

Partners involved: Mimiro

API:

Calls

- *POST /milk_yield_prediction/{architecture_uuid}/train: train this model architecture on a given training input feature set; return an identifier of the trained model.*
- *POST /milk_yield_prediction/{model_uuid}/predict: predict milk yield for a given prediction input feature set using the specified model; return the predicted yield value (milk/day, milk/week/milk/month).*

- *GET /milk_yield_prediction /trained_models: receive a list of trained models that are available to this user.*
- *GET /milk_yield_prediction /model_architectures: receive a list of ML model architectures that are available to this user for training on the user's training data, together with the configuration parameters per architecture.*

Objects

- "training input features" is an object with the following variables per animal:
  - Animal-id
  - Birth data
  - Parturition date
  - Days in milk
  - Culling data
  - Concentrate amount
  - Milk yield
  - ML Model configuration parameters
- "prediction input features" is an object with the following variable per animal:
  - Animal-id
  - Birth data
  - Parturition date
  - Days in milk
  - Concentrate amount
  - Predicted Milk yield
  - Predicted culling data
- Trained models:
  - A list of trained models available to the user
  - UUID
- Model architecture:
  - A list of available ML architectures (e.g. Autoregressive Integrated Moving Average (ARIMA), DeepAR, Exponential Smoothing (ETS) Algorithm, Non-Parametric Time Series (NPTS) Algorithm, Random Forest)
  - Name
  - Description
  - Configuration parameters

Pilots involved: 4.1, 4.2, 4.4, 5.4

5.6.2 Component F.2 Poultry Feeding

What: Monitoring food level in a silo bin and alarming feed supplier when food level reaches the threshold level. Feed producer will provide the inputs related to the silo bin characteristics (bin volume) and feed stored (density). Radar sensors deployed on top of the bin will measure the feed level. Data are integrated at the agroNET platform. By combining all inputs notifications are created thus providing optimization of delivering routes and forecasting of ordering needs.

Input:

Features	Type	Notes
id	Integer	
Silo dimensions	Float	For bin volume calculation

Type of food	Varchar	Food
Food density	Float	Density of specific food currently stored at the silo bin
Distance	Varchar	Average distance from top of silos to food measured with radar
Age of chicken	Float	Age of the flock

Table 26: Component F2 Input Data

Output:

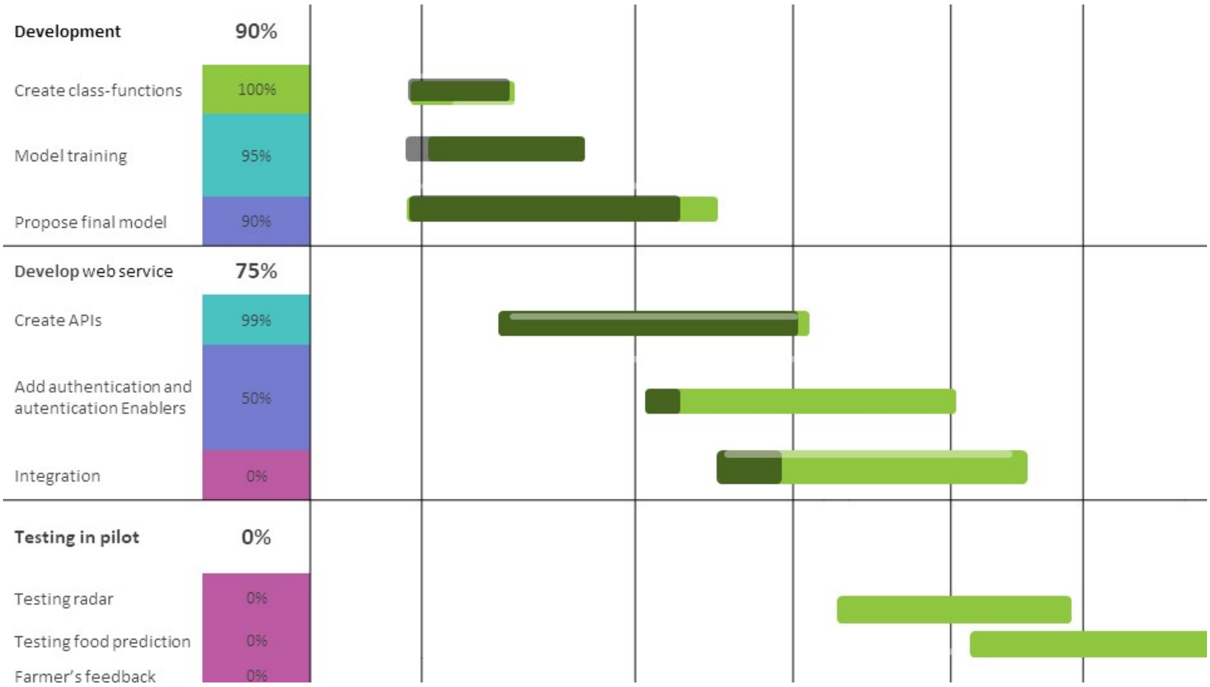
Features	Type	Notes
Poultry feeding	String	Good / not good

Table 27: Component F2 Output Data

Related Generic Enablers: None.

Status: Developed.

Development Plan:



Responsible Partner (name and organisation): Nenad Gligoric (DNET)

Partners involved: DNET, OT, SINKOVIC

API:

Calls

- *POST /poultry_feeding/{silo_uuid}/: returns the data about the silos including amount of food available.*
- *POST /poultry_feeding /{silo_uuid}/distance: data used for prediction, distance from top of silo to food.*

Objects

- Data about the silos:

    - silo-id
    - Volume
    - Type of food
- Data used for prediction:
    - Animal-id
    - Birth data
- Trained models:
    - A list of trained models available to the user
    - UUID

Pilots involved: 4.4, 5.4

5.7 *DSS AREA: G – Animal Welfare*

5.7.1 Component G.1 Estimate Animal Welfare Condition

What: classification application, where the values to predict are a discrete class label such as healthy or sick. In that case, the random forest algorithm will take a majority vote for the predicted class. During training, it is necessary to provide to the model any historical data that is relevant to the problem domain and the true value we want the model learns to predict. The model learns any relationships between the data (known as features in machine learning) and the values we want to predict (called the target). The decision tree forms a structure, calculating the best questions to ask to make the most accurate estimates possible. When a prediction is requested from the decision tree, the same data that has been used during the training (the characteristics) must be provided to have an estimate based on the structure it has learned.

Below there is an example of classification class problem with input data set to predict if the cows are affected by lameness or not.

Input:

Features	Type	Notes
Index	Integer	
Date	DateTime	Date only Format: DD/MM/YYYY
Pedometer	Integer	
Cow	Integer	
MID	Integer	
Lactations	Integer	
Daily Production	Float	
Average Daily Production	Float	
Daily Fat	Float	
Daily Proteins	Float	
Daily Fat/Proteins	Float	Fat and proteins ratio
Conductivity 1	Float	
Conductivity 2	Float	
Conductivity 3	Float	
Activity 1	Integer	
Activity 2	Integer	
Activity 3	Integer	

Total Daily Lying	Integer	
ActualLameness	String	Contains the true values for training as Healthy or Sick
PredictedLameness	String	Null, will contains the values predicted by the algorithm
ActualMastitis	String	Contains the true values for training as Healthy or Sick
PredictedMastitis	String	Null, will contains the values predicted by the algorithm
ActualKetosis	String	Contains the true values for training as Healthy or Sick
PredictedKetosis	String	Null, will contains the values predicted by the algorithm

Table 28: Component G1 Input Data

Output:

The output parameters produced by the component are the same in structure and in the model as the input parameters; consequently, only the parameters predicted by the algorithm are described:

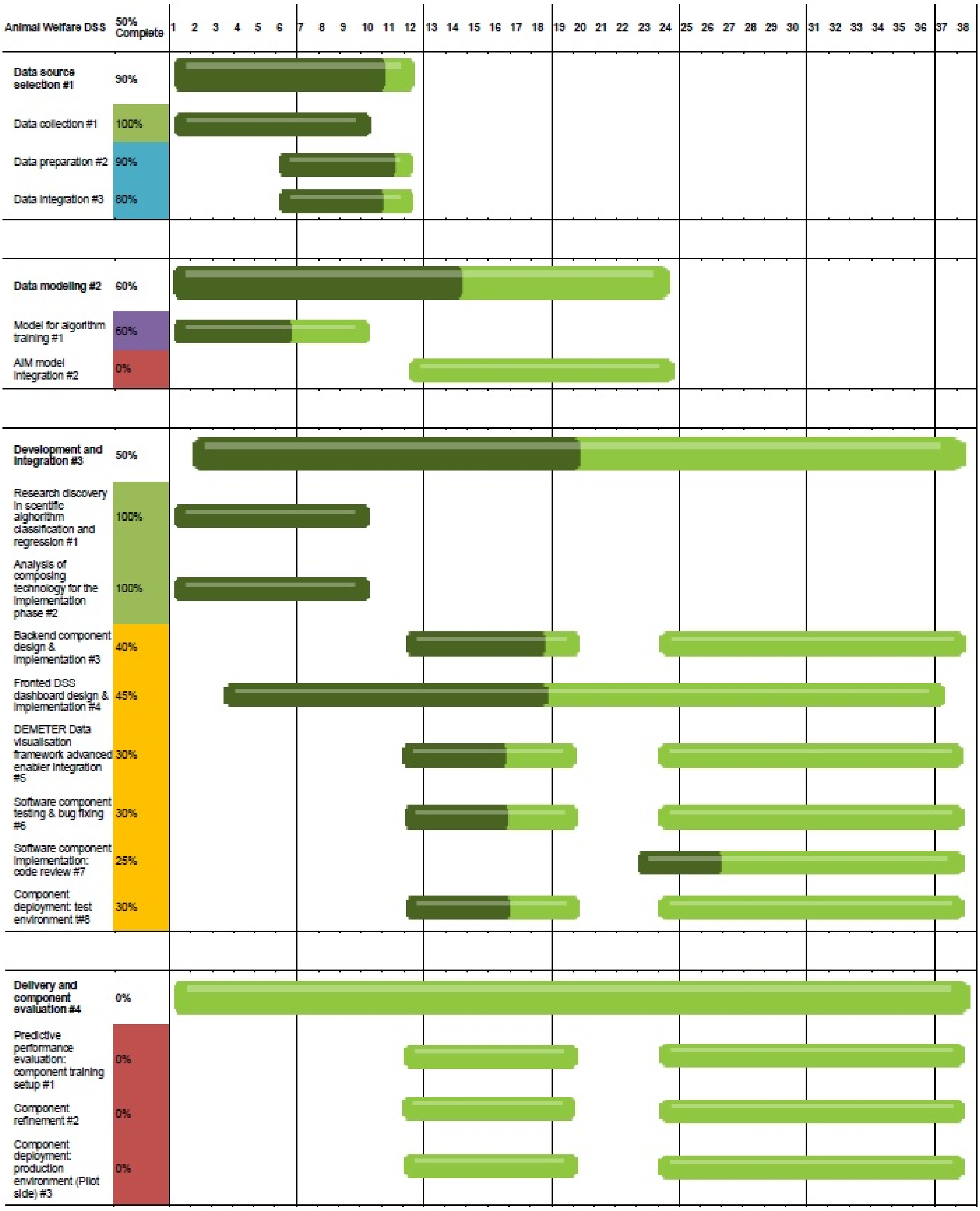
Features	Type	Notes
PredictedLameness	String	Predicted values as Healthy or Sick
PredictedMastitis	String	Predicted values as Healthy or Sick
PredictedKetosis	String	Predicted values as Healthy or Sick

Table 29: Component G1 Output Data

Related Generic Enablers: Security Enabler.

Status: In development but needs upgrades/modifications to match DEMETER needs.

Development Plan:



Animal Welfare DSS	50% Complete
Data source selection #1	90%
Data collection #1	100%
Data preparation #2	90%
Data integration #3	80%
Data modeling #2	60%
Model for algorithm training #1	60%
AIM model integration #2	0%
Development and integration #3	50%
Research discovery in scentific alghorithm classification and regression #1	100%
Analysis of composing technology for the implementation phase #2	100%
Backend component design & implementation #3	40%
Fronted DSS dashboard design & implementation #4	45%
DEMETER Data visualisation framework advanced enabler integration #5	30%
Software component testing & bug fixing #6	30%
Software component implementation: code review #7	25%
Component deployment: test environment ##8	30%
Delivery and component evaluation #4	0%
Predictive performance evaluation: component training setup #1	0%
Component refinement #2	0%
Component deployment: production environment (Pilot side) #3	0%

Responsible Partner (name and organisation): Antonio Caruso (ENG)

Partners involved: ENG, DNET

API: Embedded Python component - Knowage Function Catalogue

Pilots involved: 4.2, 4.3, 4.4, 5.4

5.7.2 Component G.2 Stress Recognition: Support Vector Machine for Poultry Stress detection

What: Recognition of stress for poultry is of paramount importance for the well-being of the chicken and real-time detection of such context could significantly help farmer to increase production and meat quality. Analysis of chicken sounds in intensive livestock production uses speech emotion recognition and voice quality evaluation techniques. The set of features used commonly for the emotion recognition is expanded with several new features, to be able to efficiently detect disease in poultry: mel-frequency filter bank (MFFB) [2] outputs and mel-frequency cepstral coefficients (MFCCs) [2]. As chicken sound depends on chicken age, environmental conditions, and the stress cause, can be classified into the following categories:

- normal (sounds in usual chicken activity)
- stressed caused by environment parameters increase (temperature, humidity, light, airflow CO_2)
- stressed caused by fear during maintenance (power losses)
- stressed caused by fear during catching and transferring procedure
- noised (noise generated by fan or feeding system masks chicken sounds)

Each above-mentioned chicken sound class will be represented by 4 models corresponding to the chicken age expressed in weeks (1, 2, 3 and 4+ weeks). These models can be treated as subclasses. If a classifier could not distinguish these subclasses with the same main class, the confused classes are joined into single.

Additionally, in case of high similarity between two classes corresponding to stressed caused by fear, there is possibility to aggregate them into single class. It is worth to note that there are systems used for sound detection as a symptom of respiratory infection, but it would not be the objective of this phase, since we do not have appropriately labelled audio data.

Input:

Features	Type	Notes
Raw sound	Blob	
Extracted features from sound	Varchar	Set of characteristics extracted as numerical float data
Air temperature	Float	
Air humidity	Float	
Air Flow	Float	
Light Intensity	Float	
CO2	Float	
Power loses	Integer	yes/no
Flock age	Float	

Table 30: Component G2 Input Data

Output:

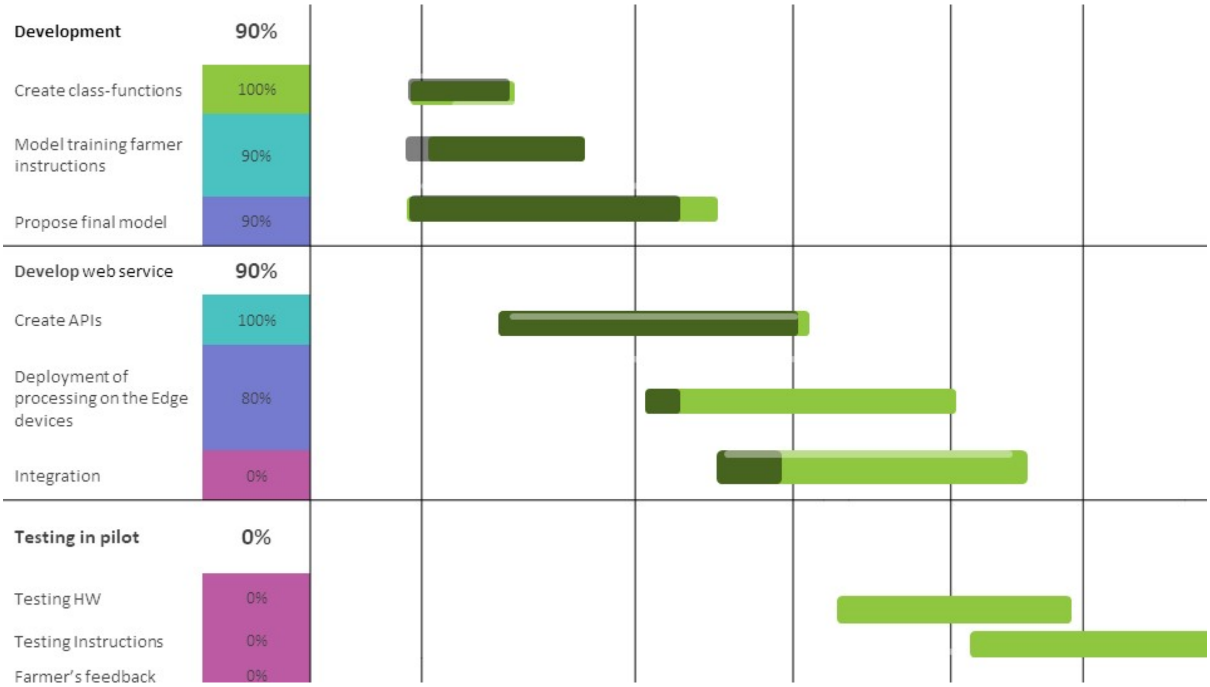
Features	Type	Notes
Id of flock	String	GS1 format of UUID
Level of stress	String	
Instructions for farmers	String	

Table 31: Component G2 Output Data

Related Generic Enablers: Pilot related component (no dependencies), Ontology mappings

Status: Present but needs upgrades/modifications to match DEMETER needs.

Development Plan:



Responsible Partner (name and organisation): Nenad Gligoric (DNET)

Partners involved: DNET, SINKOVIC, OT

API:

<u>Calls</u>

- *POST /poultry_stress/environment/{flock_uuid}/: posts environment data (temp, humidity, air flow light, CO2, power loses, etc.).*
- *POST /poultry_stress/flock/{flock_id}: set data about the flock.*
- *GET /poultry_stress/flock/{flock_id}: returns data about the flock.*

<u>Objects</u>

- Data about the flock:
    - flock-id
    - flock age
    - Type of flock
    - Stress-level
- Data used for prediction:
    - Air Temperature
    - Air Humidity
    - Air Flow
    - Light Intensity
    - CO2
    - Power Losses
- Trained models:
    - A list of trained models for instruction for the user

- UUID

Pilots involved: 4.4, 5.4

5.8 *DSS AREA: H - Traceability*

5.8.1 Component H.1 Traceability

What: Classification application, where the values to predict are a discrete class label such as high, medium, or low. In that case, the random forest algorithm will take a majority vote for the predicted class. During training, it is necessary to provide to the model any historical data that is relevant to the problem domain and the true value we want the model learns to predict (Actual Quality). The model learns any relationships between the data and the values to predict. The decision tree forms a structure, calculating the best questions to ask to make the most accurate estimates possible. When we ask the decision tree to make a prediction, we must give it the same data it used during training excluding the true value and it gives us an estimate based on the structure it has learned.

Input:

Features	Type	Notes
Date	DateTime	Date only Format: dd/MM/yyyy
Time	DateTime	Time only, format hh:mm:ss
Product name	String	
AciditySH	Float	
Casein	Float	
Density	Float	
Fat	Float	
Freezing Point mC	Float	
Lactose	Float	
Protein	Float	
SNF	Float	
Urea	Float	
Remark	String	
Actual Quality	String	Contains the true values for training as High, Medium, or Low
Predicted Quality	String	Null, will contains the values predicted by the algorithm

Table 32: Component H1 Input data

Output:

The output parameters produced by the component are the same in structure and in the model as the input parameters; consequently, only the parameters predicted by the algorithm are described:

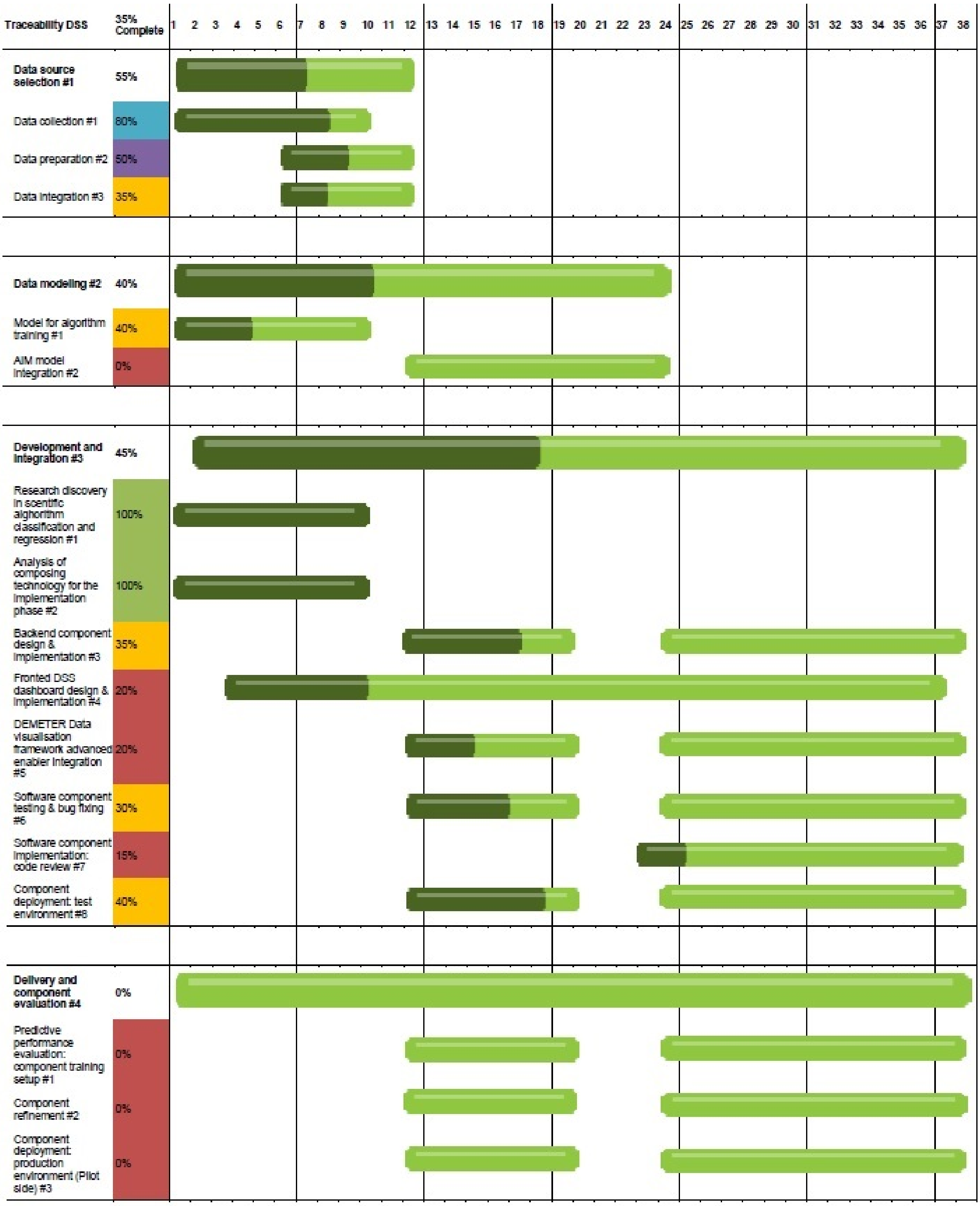
Features	Type	Notes
Predicted Quality	String	Predicted values as High, Medium, or Low

Table 33: Component H1 Output data

Related Generic Enablers: Security Enabler.

Status: In development but needs upgrades/modifications to match DEMETER needs.

Development Plan:



Responsible Partner (name and organisation): Antonio Caruso (ENG)

Partners involved: ENG

API: Embedded Python component - Knowage Function Catalogue

Pilots involved: 4.2

5.8.2 Component H.2 Transport Condition

What: The transport condition in which the product is transported throughout the supply chain from source to consumer. The available data should ensure the origin of the product, including producer and environmental conditions in which some package was transported. The most important data are written into the blockchain, including a unique item-level id to be able to assess the transport conditions for each package individually.

Input:

Features	Type	Notes
id	Varchar	Unique item-level id GS1 format
Place of production	Varchar	
Certificates	Varchar	
Poultry type	Varchar	e.g., Hybrid
Producer	Varchar	
MHR	Varchar	vaccination, medical treatments
Transport conditions	Integer	Environment conditions - good/not good based on a set of parameters

Table 34: Component H2 Input Data

Output:

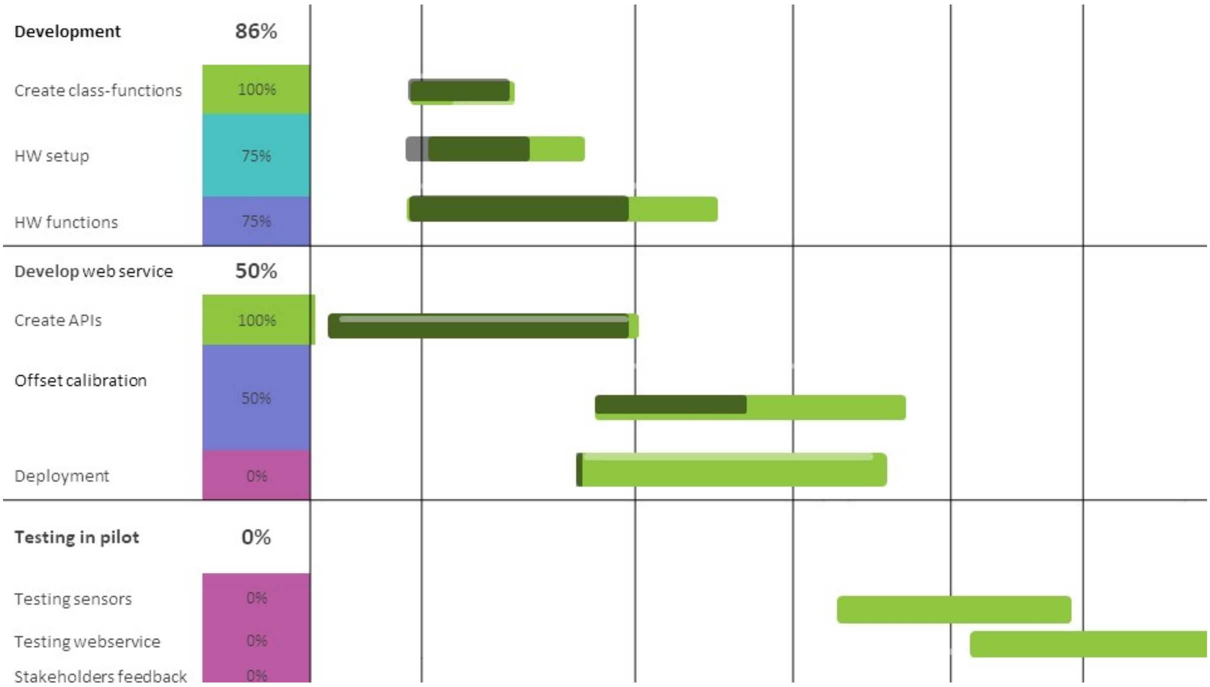
Features	Type	Notes
Id of package	String	GS1 format of UUID
Transport conditions	Boolean	

Table 35: Component H2 Output Data

Related Generic Enablers: Ontology mappings.

Status: In development.

Development Plan:



Responsible Partner (name and organisation): Nenad Gligoric (DNET)

Partners involved: DNET, OT, SINKOVIC, UDG, Plantaze

API:

Calls

- *POST /transport_conditions/environment/{device_uuid}/: posts environment data (temperature).*
- *POST/transport_conditions /set_package/{package_id} set data about the flock.*
- *GET /transport_conditions /package/{package_id} returns data about the flock.*

Objects

- Data about the package:
  - UUID
  - Type of goods
  - Transport conditions
- Data used for prediction:
  - temperature
  - CO2

Pilots involved: 5.1, 5.4

5.8.3 Component H.3 Field Book and FaST

What: In almost all European countries, farmers must create a field book where they must record some information about their farms, among other things all the information related to phytosanitary treatments. This field book is being done on paper being a process tedious and without allowing exploited this information properly.

Each region, even within the same countries, has different fields required, so the main objective from this component is to make it easier to the farmers to maintain this field book by providing them with a tool that lets them do it in a digital way and that can be adaptable to the different fields required.

In addition, when this information will be digitalized, it could be exploited in different ways as, for example, farmers can access easily and even use different DSS to take properly decisions.

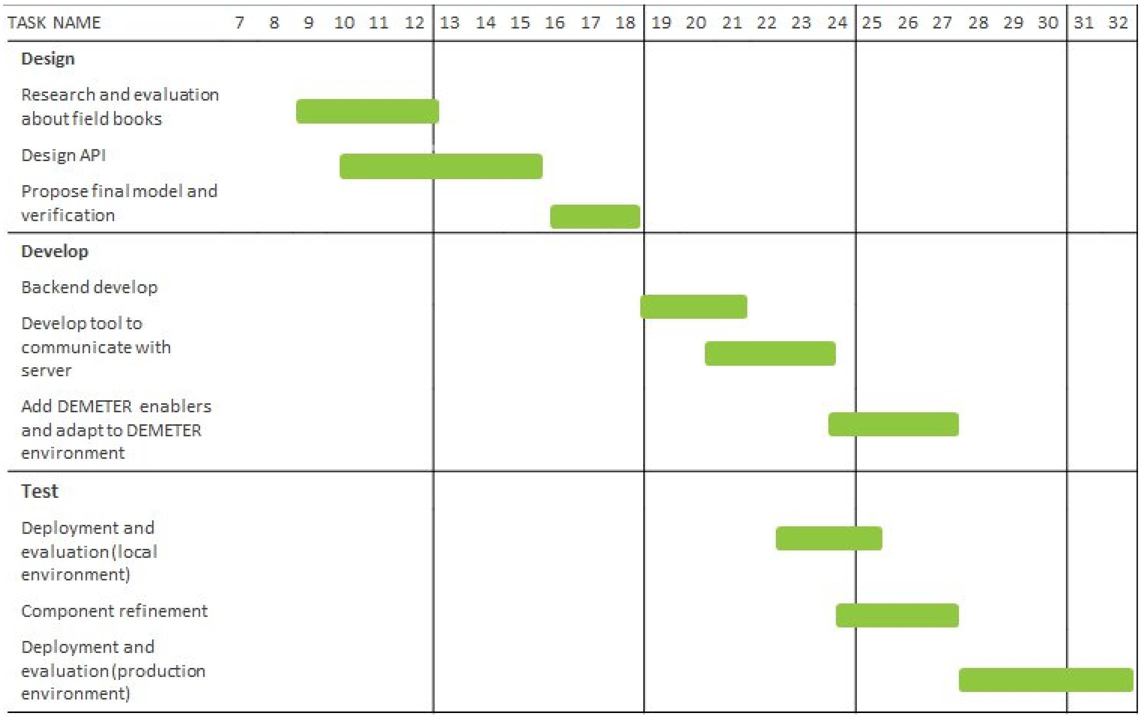
Input: General information on farms and persons related, identification of farm plots, information on phytosanitary treatments, analysis of phytosanitary products analysis, commercialized harvest register and fertilization register.

Output: General information on farms and persons related, identification of farm plots, information on phytosanitary treatments, analysis of phytosanitary products analysis, commercialized harvest register and fertilization register.

Related Generic Enablers: Ontology mapping (eCrop & LUCAS) and Security enabler.

Status: To be produced by DEMETER.

Development Plan:



Responsible Partner (name and organisation): Pablo Gallego (TRAGSA)

Partners involved: TRAGSA

API: It will be a RESTful API so all CRUD operations will be let to all the objects with next format "/objectType/objectID":

- *exploitation: object that identifies a farm and that will include general attributes about it like identification numbers or location.*
- *Person: object that identifies a person that will have some relation with the exploitation, include general information like name, phone, or PIN.*
- *personPhytosanitary: object that identifies a person that will have some phytosanitary relation with the exploitation, include person object and specific phytosanitary information.*
- *machineryPhytosanitary: object that will record all the machinery that will be use in some phytosanitary treatments, includes information like kind of machinery and dates of purchase/last revision.*
- *farmPlots: object that identifies a particular plot in the exploitation, include information like location or crops.*
- *phytosanitaryTreatments: object that will record all the different treatments phytosanitary that has been done in the exploitation with information related to his motive and his results.*
- *treatedSeed: object that will record the different treatments phytosanitary that has been done to the seeds before their plant.*
- *treatedPostHarvest: object that will record the different treatments phytosanitary that has been done post-harvest.*
- *externalTreated: object that will record the different treatments phytosanitary that has been done external to the plots, for example to warehouse or conveyance.*
- *productsProtectionAnalysis: object that will record the analysis that has been done.*
- *harvestedMarked: object that will record the information related to the marked harvested.*
- *fertilizationRegistry: object that will record if some fertilization operation has been done and information about it.*

Pilots involved: Any crop-related pilot

5.9 *DSS AREA: I - Benchmarking*

5.9.1 Component I.0 Indicator Engine for Benchmarking Purpose

The Indicator Engine is a component, which allows the identification of indicators to assess the current agronomic, economic, and environmental sustainability with data available at the farm level. The selected indicators will be calculated to be used for benchmarking purposes.

The components will have the same features:

- **Indicator Management**: The core part of the components will be the repository of the indicators available to be calculated in the system. Each indicator will be associated with a general benchmarking area: agronomic, economic, or environmental. For each indicator, the input data required to calculate the indicator will be defined.
- **Indicator Calculators**: For each indicator, the way to calculate the indicator value will be defined. There will be different types of calculators:
    - Simple Indicators: if the indicators can be calculated with an algebraic formula from the input data, the registration of the formula will allow the calculation of the indicators. That functionality will be integrated in the I0 component.
    - Component/based indicators: if the indicator calculation will be based on a specific algorithm, it will be possible to call a DEMETER components to calculate the result of the indicators; e.g., the water efficiency can be expressed comparing the actual irrigation amount with the water needs estimated by a Decision Support System component (e.g. DSS Component-B.1); the calculator will prepare the input for the external components to get the results.
- **Indicator Storage**: The benchmarking engine, by design, will never store the input data from the farms. The actual storing of the calculated indicators will be optionally activated according to each user's decision. Any direct reference to the farm will be removed from the indicator value and the indicators will be aggregated to calculate a set of reference values to be used by the other benchmarking components.

A first set of potential indicators has been built based on the result of the questionnaires. In the development of the components we will start from the most requested performance indicators, starting with the yield comparison, then moving on to the water efficiency and farm and field water footprint. These indicators will be developed along with the related DSS Area A (Crop Yield), F (Animal Yield) and B (Irrigation Management). In addition, there will be environmental indicators related to greenhouse gas emission on the farm level as described in deliverable D4.1 - section 9.2.2 "Environmental indicators". The data input for the calculations for the environmental indicators will be collected via API's from public databases with soil types, weather data, and farm management systems that the farmer uses for livestock or plant production, farm accounts, digital invoices, and digital settlements.

Input: The inputs may cover all different data covering the farm, as defined by section 5.7.4 "Results on data availability" of D4.1. In any case, we need a spatial and a temporal definition of the input data. For spatial information, the farm can decide to share only the administrative division of the farm centre (NUTS3), if it does not want to share the geographic location.

The data will be grouped in thematic groups: farm general structure and composition, detailed farm spatial information, weather and sensors data, soil data, yield (at different scales of detail), logs of crop practices (irrigation, fertilisation, soil tillage, pest management), data on animal welfare, machinery data, economic data (input, output, profits).

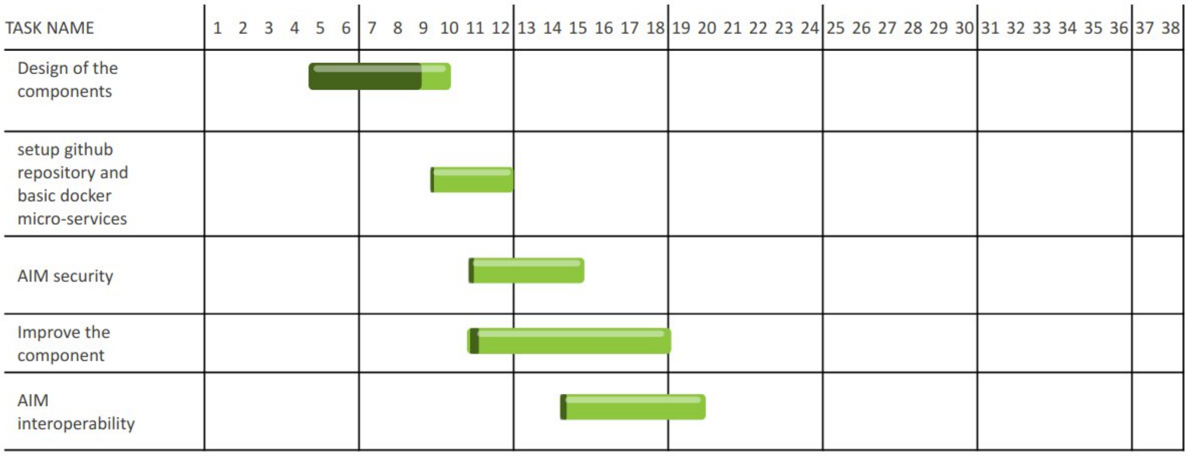
Output: The system will produce a set of indicators. An interface showing the indicators calculated with data available at the farm level will be produced.

The results will be an array of indicator values associated with the indicator code, the area of the indicators, the year/time period related to the indicators and, if available, the reference value to be used in the comparison.

Related Generic Enablers: The component will be integrated with all the mandatory generic enablers; the main integration will involve the Semantic interoperability enablers – to provide a standardized interfaces for data retrieval from various sources e to make available the result in a standard mode, and the data security, data ownership and access right enablers.

Status: While the system infrastructure has been designed; the MVP system is in the initial stage of development.

Development Plan:



Responsible Partner (name and organisation): Diego Guidotti (AGRICOLUS)

Partners involved: AGRICOLUS, ENG, MIMIRO, Landbrukets, INTRA, TRAGSA, SINTEF, POZNAN

API:

<u>Calls</u>

- *POST /i0_indicator/add_indicator: receive a “benchmark_indicator” input object describing the indicator.*
- *POST i0_indicator/indicator_code/calc: receive an input with the input data an array of “benchmark_input”.*

<u>Objects</u>

- “benchmark_indicator” is an object describing a DEMETER indicator:
    - indicator_code: unique identifier of the indicators
    - variables: array listing the needed variable (using the AIS standard): for each variable, the system needs:
        - variable_code
        - unit_of_measure
        - mandatory: true/false

    - indicator_type: simple / microservice: a simple indicator can be defined with an algebraic “formula”, or if it we need to use an external microservices to calculate it
    - formula: the algebraic formula expressed using MathSON
    - url_call: the URL of the microservice for the complex calculation
- “indicator_input”: an array of single data input:
    - variable_code: code of the variable
    - value: value
- “indicator_output”: the basic object defining the result value of the indicator engine
    - indicator_code: code describing a specific indicator
    - value: the value calculated of the indicators by the benchmarking algorithm

Pilots involved: All the pilots will be involved.

5.9.2 Component I.1 Generic Farm Comparison

The component will provide, to each farm, a set of basic economic indicators to be used to get a general benchmark of the farm activities. One of the main limitations for the benchmarking solution is the actual availability of farm data due to the difficulties in collecting them and the lack of trust of the farmers in providing this data.

We have defined the minimal set of data that can be easily available to create the “minimum viable product” (MVP). The system will be connected to the Farm Accountancy Data Network (FADN) and will be able to provide to each farm an estimated reference of the economic farm performance indicators: Expected output, expected input and expected profit, along with an estimation of the input and output division in general areas.

From the farm general structure, a set of general indicators (European regions, dimension, surface by crops, composition of livestock) will be defined. The FADN database will be used to find the closest reference using a search to minimize the Euclidean distance of the farm indicators with the FADN record reference. The economic performances of each farm can be compared with the average value of the class to which it belongs.

Input: A minimum set of mandatory data has been identified. This basic information is the generic location of the farm centre (using coordinates or the related administrative region), the surface of the farm by crop group (e.g. cereals, permanent crops, horticultural crops, etc.) and the number of livestock units by species.

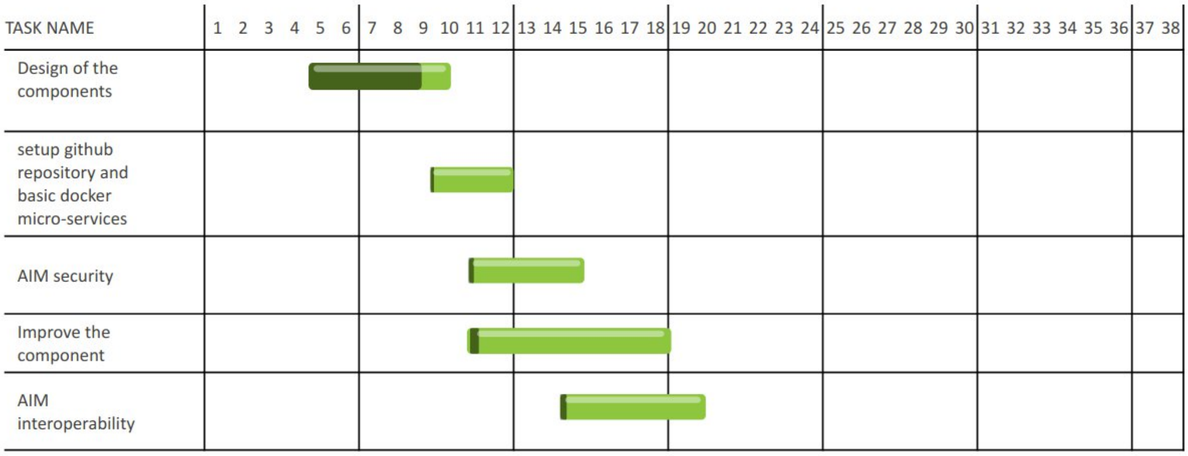
If available and shareable by farms, the system may also acquire the basic information from the farms balance sheet (input, output, profit) to calculate the ratio between the expected value and the actual ones, otherwise the user can get the reference value comparing them privately with his own data.

Output: A set of economic indicators derived from FADN or EuroStat that a farm can use to get a reference (e.g. expected output, input and profit) and make the comparison with the same indicators calculated with the farm’s own data.

Related Generic Enablers: The component will be integrated with all the mandatory generic enablers; the main integration will involve the Semantic interoperability enablers – to provide a standardized interfaces for data retrieval from various sources e to make available the result in a standard mode, and the data security, data ownership and access right enablers.

Status: While the system infrastructure has been designed; the MVP system is in the initial stage of development.

Development Plan:



Responsible Partner (name and organisation): Diego Guidotti (AGRICOLUS)

Partners involved: AGRICOLUS, ENG, MIMIRO, Landbrukets, INTRA, TRAGSA, SINTEF, POZNAN

API:

Calls

- *POST /i1_generic/bench: receive a "generic_benchmark_input" input object producing a "generic_benchmark_output" object.*

Objects

- "generic_benchmark_input" is an object with a set of data; the format may be improved in time according with the proposed AIM data format:
    - lat/lon location of the farm center
    - crop: an array with the surface per each crop (per year):
        - year
        - crop_code: the code of the type of crop
        - surface: the surface in hectares associated to the crop
    - livestock:
        - year
        - animal_code the animal species
        - number: the number of animals expressed in standard livestock unit (LSU)
- "generic_benchmark_output": all the benchmarking API will produce an array of "benchmark_result"
- "benchmark result": the basic object defining the result value of the:
    - indicator_code: code describing a specific indicator
    - year: year for which a benchmarking is calculated
    - value: the value calculated of the indicators by the benchmarking algorithm
    - value_ref: the reference value of the indicator
    - indicator_name: a label explaining the indicators

Pilots involved: All pilots will be involved.

5.9.3 Component I.2 Neighbour benchmarking

The component allows a group of farms (e.g., cooperatives, consortia, other organizations belonging to a specific area) to share data and compare performance. They select what kind of data they want to share to make performance comparison and which kind of output they wish to obtain.

The first available function is the "group creation", which allows the group coordinator (usually a system integrator) to create a group. The group can be open (a user with a link to the group can join freely) or closed (only a predetermined set of users can participate). The group creator can also define the list of target indicators that need to be calculated and it will be possible to define the mandatory data needed to participate in the group.

The coordinator can choose the method used to calculate the reference values (e.g., average, median, top 10 percentile). Finally, it will be possible to create a set of profiles within a group to allow heterogeneous farms to compare with farms belonging to the same type (e.g., create sub-group related to specific location, create a group for organic and one for conventional farms etc…).

The second function will allow a user to join the group and to share the data with the other group users. The single farm data and indicator results will never be shared, but the value will be used to create the benchmarking target value that will be shared by all the participants. After connecting their own data to the Benchmarking system, the user will be able to access a dashboard showing their own performance with a comparison with the group reference values. The dashboard will also evaluate the variability in time, make a comparison of the actual farm indicators along with the ratio to the reference target in time.

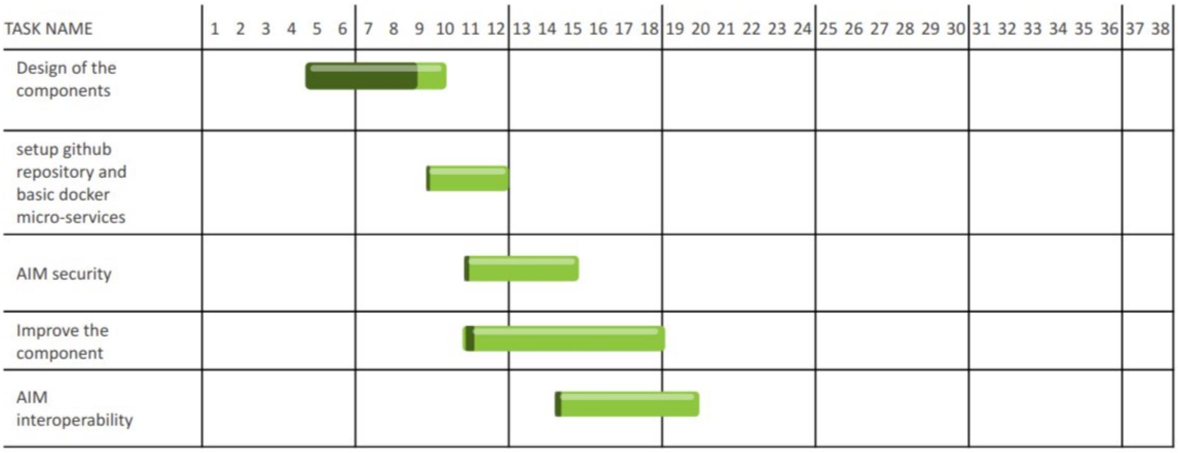
Input: the system requires the data selected by the group coordinator as mandatory; if the farm has insufficient data, it will be received feedback about the missing information.

Output: an interface showing how it is performing each farm, by comparing the selected indicators with the neighbouring indicator average or the "best in class" among neighbours.

Related Generic Enablers: The component will be integrated with all the mandatory generic enablers; the main integration will involve the Semantic interoperability enablers – to provide a standardized interfaces for data retrieval from various sources e to make available the result in a standard mode, and the data security, data ownership and access right enablers.

Status: While the system infrastructure has been designed; the MVP system is in the initial stage of development.

Development Plan:



Responsible Partner (name and organisation): Diego Guidotti (AGRICOLUS)

Partners involved: AGRICOLUS, ENG, MIMIRO, Landbrukets, INTRA, TRAGSA, SINTEF, POZNAN

API:

Calls

- *POST /i2_neighbour/:group_code/create the call create a new "benchmarking space" that can be shared from a group of users to share their data; need a "neighbour_groups_settings" object.*
- *POST /i2_neighbour/:group_code/:id_farm/add: allow an user to add a set of farm data to be used to calculate the indicators; The input will be an array of "benchmark_input" objects.*
- *GET /i2_neighbour/:group_code/:id_farm/ return the result of the benchmarking using an array of "benchmark result" object; the value_ref in the object is the average of the indicator calculated for the group of neighbour; the object has been already described in I.1.*

Objects

- "neighbour_groups_settings" is an object defining the structure of the group of neighbours:
  - "group_name": name of the group
  - "accessibility_level": who can access the group ("open", "closed")
  - variables: array of variables to be used for benchmarking (e.g. "milk_protein", "olive_yield" etc.) the variable will be described using AIS standard
  - associated_users: array with the user that can access the cluster (if not open)
  - geometry: if available the shape of the area related to the benchmarking
- "benchmark_input":
  - "date": date of acquisition of the data
  - "variable_code": code of the specific variable (e.g. "milk_protein", "olive_yield")
  - "value": value of the variable measured in the farm

Pilots involved: the component will be available to all the pilots, but there is already a set of pilots that are actively working on these benchmarking activities:

- Pilot 2.4 Benchmarking at Farm Level Decision Support System (benchmark productivity and sustainability performance of the farms)
- Pilot 3.1 Decision Support System to support olive growers (share olive yield and farming practices)
- Pilot 4.1 and 4.2 Benchmarking milk production and quality

5.9.4 Component I.3 Technology benchmarking

This component fits the general DEMETER Objective 3: "Establish a benchmarking mechanism for agriculture solutions and business, targeting end-goals in terms of productivity and sustainability performance of farms, services, technologies, and practices based on a set of key performance indicators that are relevant to the farming community".

The farms productivity and sustainability performance has been addressed in the previous components, the current one is focusing on benchmarking services, technologies, and practices with two specific aims:

- provide a reusable component allowing a farmer or a group of farmers to evaluate the performance of a technology from the agronomic, economic, and environmental point of view.
- use the developed component as a DEMETER benchmarking mechanism, using the data collected by the farms participating at the several pilots to support the calculation of the DEMETER project KPIs.

The component has the following functions:

- create a group related to a specific solution or set of solutions, the solution coordinators will create the comparison group associating the group with the user that is testing the technology or the solution, the function is similar at the "group creation" already described in component I.2:
  - the group will contain two sub-groups, the target group, with the cases adopting the technology and a control group with the cases that have not adopted the technology.
  - choose a list of KPIs to compare the adoption of the solution, it will be defined a preliminary set of KPIs that will be combined later.
- Associate each group of the specific solution with the farms or the fields that have adopted that solution: some solutions should be tested at farm level because they involve all the farm activities (e.g., a general digitalisation of the farm), other solutions should be tested on a group of specific fields to make an internal comparison (e.g., testing the solution only on a sub-set of fields); each case should be added to the target or control group.
- Data collection: the system will acquire the required data.

Input: the following types of data will be needed by the component:

- Yield - the yield is needed to estimate the economic output of the production and to calculate the solution efficiency (e.g., for water management solution the water efficiency will be expressed in unit of water per unit of yield); it can be included also crop quality parameters if impacted by the solution (e.g., the increase of fat content in milk due to a precision feeding solution);
- Specific data - the system will require the input data if the technology has produced a saving or a change of agronomic input (e.g., monitor water used for irrigation to calculate the reduction of volume after using an irrigation DSS); this type of input may include workforce, machinery, water, fertilizer, pesticides etc…
- Prices - to calculate the main economic benchmarking we need to know the price of:
  - output: it can be a reference value for all the cases or defined case by case (the technology may affect the product price).
  - input: cost of each unit of input.

- cost of the solution: this should include the estimation of the total investment cost connected with the adoption of the technology (licences, machinery, training, etc…).

If the Economic data are not available, only the agronomic indicators will be calculated.

Output: The output of the component is a report with the value of a set of indicators for the target group (farms that have adopted the technology) and the control group (farms that have not adopted the technology).

The following is a list with some test indicators. A set of indicators are general and will be needed for all the types of adopted solutions, some other indicators will be dedicated to some sectoral solution. (e.g., the irrigation DSS will be evaluated for a set of water-related indicators).

Agronomic indicators		
Indicators	**Description**	**Type**
Yield (t/ha)	yield per hectare	general
N use efficiency (kg/kg)	yield /Nitrogen unit	specific
Water use efficiency (kg/m3)	yield/total water consumption	specific
Treatments (n)	number of treatments	specific
Environmental indicators		
Indicators	**Description**	**Type**
Total water consumption (m3)	total water used for irrigation	specific
Total nitrogen consumption (N units)	total nitrogen used for fertilization	specific
Water footprint (m3)	total water consumed for the production, including the irrigation water and the water needed to produce the input	specific
Economic indicators		
Indicators	**Description**	**Type**
Gross saleable production (GSP)(€/ha)	yield per unit selling price	general
Specific costs (€/ha)	yum of the variable costs per area unit	general
Gross margin (€/ha)	GSP - specific costs	general

Table 36: Component I3 Input Data

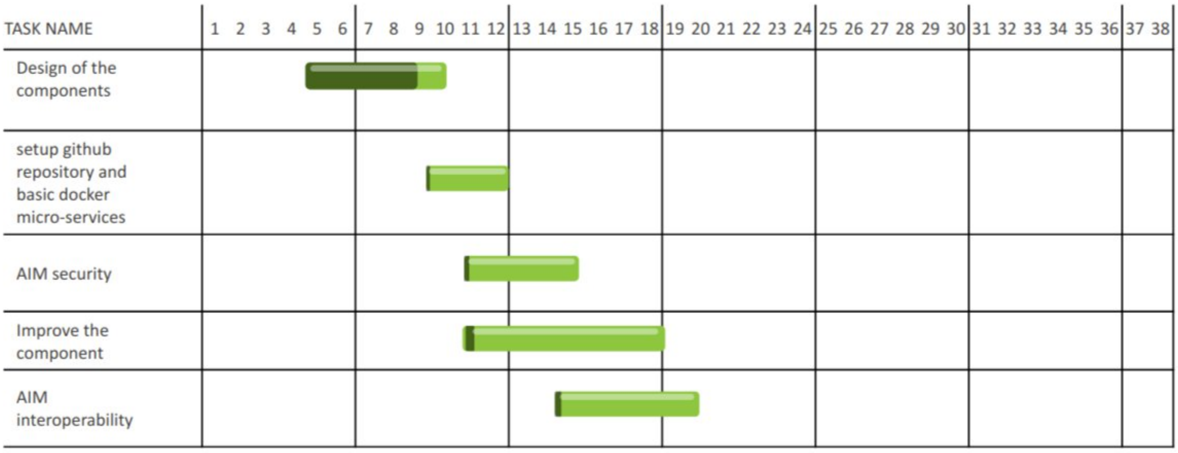
For each indicator we will calculate the KPIs of the technology, calculating the difference of the estimated value for target and control group (e.g., the decrease of water and pesticide usage per yield unit and per hectare after the usage of DEMETER enabled DSS).

If the economic indicators are calculated, the system will estimate the return of investment (ROI) of the solution, calculating the differential gross margin of the target cases divided by the total cost of the investment for the enabling technology.

Related Generic Enablers: The component will be integrated with all the mandatory generic enablers; the main integration will involve the Semantic interoperability enablers – to provide a standardized interfaces for data retrieval from various sources e to make available the result in a standard mode, and the data security, data ownership and access right enablers.

Status: while the system infrastructure has been designed, the MVP system is in the initial stage of development.

Development Plan:



Responsible Partner (name and organisation): Diego Guidotti (AGRICOLUS)

Partners involved: AGRICOLUS, ENG, MIMIRO, Landbrukets, INTRA, TRAGSA, SINTEF, POZNAN

API:

<u>Calls</u>

- *POST /i2_technology/:technology_code/create the call create a new "benchmarking space" that can be shared from a group of user that are using the same technology; need a "technology_groups_settings" object.*
- *POST /i2_technology/:technology_code/:id_farm/:id_level/add: allow an user to add a set of data to be used to calculate the indicators; The data should be The input will be an array of "benchmark_input" objects.*
- *GET /i2_technology/:technology_code/:id_farm/ return the result of the benchmarking using an array of "benchmark result" object; the value_ref in the object is the average of the indicator calculated for user using the same level; the object has been already described in I.1.*

<u>Objects</u>

- "technology_groups_settings" is an object defining the structure of the group of users using the same technology:
  - "group_name": name of the group
  - "levels": an array of levels of comparison; by default we have two levels: "control" : a case not using any technology and "test" a case using the technology; it will be possible to define a custom list of levels
  - "accessibility_level": who can access the group ("open", "closed")
  - variables: array of variables to be used for benchmarking (e.g. "irrigation_water", "field_profit" etc… the variable will be described using AIS standard
- "benchmark_input":
  - "date": date of acquisition of the data
  - "variable_code": code of the specific variable (e.g. "milk_protein", "olive_yield")

- "value": value of the variable measured in the farm

Pilots involved: The solution will be available for all the pilots.

6 Adaptive Visualisations for Dashboards

6.1 Overview

Data visualization is the presentation of data in a pictorial or graphical format, and a data visualization tool is the software that generates this presentation. Data visualization provides users with intuitive means to interactively explore and analyse data, enabling them to effectively identify interesting patterns, infer correlations and causalities, and supports well informed activities ([3]).

Data visualization and user interaction play a key role in exploiting Big Data sources ([4]). They support users for browsing, understanding, and discovering data insights. Nonetheless, Big data characteristics, such as volume, variety, and velocity pose a number of challenges for visualization ([5]).

Indeed, current visualization and exploration systems should effectively and efficiently handle the following aspects:

- **Real-time Interaction**. Efficient and scalable techniques should support the interaction with billion objects datasets, while maintaining the system response in the range of a few milliseconds.
- **On-the-fly Processing**. Support of on-the-fly visualizations over large and dynamic sets of volatile raw (i.e., not pre-processed) data is required.
- **Visual Scalability**. Provision of effective data abstraction mechanisms is necessary for addressing problems related to visual information overloading (a.k.a. over plotting).
- **User Assistance and Personalization**. Encouraging user comprehension and offering customization capabilities to different user-defined exploration scenarios and preferences according to the analysis needs are important features

Nowadays, an increasingly large number of diverse users (i.e. users with different preferences or skills) explore and analyse data in different sectors. For example, farmers need to measure and understand the impact of huge amount and variety of data which drive agricultural production and livestock management.

Big Data is expected to have a large impact on Smart Farming and involves the whole supply chain. Smart sensors and devices produce big amounts of data that provide unprecedented decision-making capabilities. Big Data will influence the entire food supply chain. Big data are being used to provide predictive insights in farming operations, drive real-time operational decisions, and redesign business processes for game-changing business models.

Domains of Big Data Visualisation and Analysis for Agriculture could be:

- **Yield predictions**: The concept of yield productivity zones was introduced in the FOODIE project and further developed in the DataBio Project ([6]). It aims at the discovery, verification, and user-friendly visualisation of long-term high and low yield productivity zones.
- **Analysis and visualisation of Agriculture Linked Open Data**: Linked data are increasingly becoming one of the most popular methods for publishing data on the Web. There is still a lack of datasets related to the agricultural facilities and farm management activities. This is in part due to the lack of standardized models for the representation of such data. The FOODIE project addressed such issues with the development of the FOODIE data model. The key motivation was to represent a continuous area of agricultural land with one type of crop

species, cultivated by one user in one farming mode (conventional vs. transitional vs. organic farming).

- **Farm machinery visualization**: Monitoring of machinery fleet movement and especially its spatiotemporal changes brings new insights about the consequences of human decisions from many areas. Economic reasons are related to economic evidence for a farmer, including fuel consumption, efficiency of trajectory etc. to revenue authority or subsidies management. On the other hand, ecologic motivations aim to decrease of environmental burden caused e.g. by high CO_2 emissions due to a lack of movement optimisation, water pollution by nitrogen due to excessive fertilization.
- **3D interactive visualization of yield productivity zones**: Allows users to explore yield productivity zones with respect to topography (represented as DTM – Digital Terrain Model, DSM – Digital Surface Model, slope, slope orientation, and topography wetness index). The perspective view contains a three-dimensional model of the farm plots and shows the area of interest in a perspective projection taking into account the observer position and their line of sight.
- **Precision Livestock Farming (PLF)**: Visualization of data coming from various sensors including the number of animals, mortality, temperature and humidity, feed and water consumption, activity and distribution of the animals and number of coughs. The output can be customised for each farm, based on its specificities. It includes information on production parameters, climatic conditions as well as behaviour, health, and welfare of the animals.

Some visualization tools in agrifood domain like Grafana ([7]), HSLayers NG ([8]) or Apache Zeppelin ([9]) (to name a few), could meet the specifications of an adaptive framework to be used in the DEMETER context. However, the technology selected to meet this need in the project is represented by Knowage ([1]).

Knowage is an open source suite developed by ENGINEERING ([10]) that combines traditional data and big data sources into valuable and meaningful information. Knowage provides advanced self-service capabilities that give autonomy to the end-user, who can build his own analysis, get insights on data, and turn them into actionable knowledge for effective decision-making processes. The software is flexible because it adopts open standards and can be used in various environments without considerable requirements. Main advantages:

- Open source and open standard
- Community edition & enterprise edition
- Modular and scalable architecture approach
- Easy customization and user-friendly solutions
- Responsive GUI, ready for mobile
- Community edition & enterprise edition
- OEM enabler.

Knowledge as well as minutes of analytical drivers, provides a whole series of graphic tools to represent the data, their meaning allowing an adaptive management of the contents within the analytical documents represented by the cockpits. The Cockpits represent an intuitive interactive interface, then can be created with a few clicks and simple drag and drop. This allows to compose analytical documents with multiple widgets and define associations among them, so that clicking on one widget data are automatically updated in other widgets.

6.2 Reference technology

6.2.1 Knowage suite for general-purpose dashboard visualisation

Knowage is the open source suite that implements a full set of features, such as data federation, mash-up, data/text mining and advanced data visualization, with a special focus on big data analytics and comprehensive support to rich and multi-source data analysis. The new Knowage BD product satisfies the whole spectrum of analytical needs in any Big Data architecture. Knowage suite is realized by Engineering Group - the leading Italian software and services company, with about 8,200 people and more than 40 offices in Italy, Belgium, Republic of Serbia, Norway, Brazil, Argentina, and United States.

Knowage provides different products, each one focused on a specific domain but mutually combinable, allowing you to build a tailored product for your own analytical scope:

- Big data is not only a matter of volume; it means be able to manage different data types and sources, combining structured enterprise data with external multi structured ones.
- Smart Intelligence means be able to easily access to enterprise structured data not only with pre-built analysis but having the full control of the analysis, with advanced ad-hoc and self-service capabilities.
- Enterprise Reporting assures that a lot of users receive their information at the right time, with a pixel-perfect and ready-to-print layout. Assurance concerns security based on end-user roles, timing as punctuality without wait, information in terms of certified data and format.
- Location Intelligence means be able to plot business data over a map, over a space, over a schema, over a vectorial picture. The goal is the production of immediate insight with mash-up techniques and without moving data between GIS system and DWH environments.
- Performance management concerns the ability to define strategic scorecards, measure and evaluate performances through thresholds and from different point of views, considering goals and efforts, alerting for critical issues, and giving facilities to make the reaction possible soon.
- Predictive analysis concerns the ability to perform advanced processing using data mining techniques for forecasting and prescriptive purposes. Moreover, predictive analysis means also be able to simulate an action, evaluating its effects on different and related assets.
- Open data is more than a collection of freely available datasets. It involves the linked (open) data technology, inherited from the semantic web. Linked data means a data management approach driven by meanings instead of by the structures and promises an alternative federative paradigm.
- Embedded Intelligence makes Knowage open. It can be linked to third-part solutions under the condition of the general AGPL v3 license ([11]).

Knowage places itself on the market as the open source suite for business analytics ensuring the freedom to access the source code and collaborate in an international community thanks to industry-leading collaborations, while letting you build your own business solution that ensure strategic decision-making and improved productivity.

Thanks to the innovation and research coming from the open source community, the Knowage suite evolves to meet the latest analytical needs, even combining traditional data with any Big Data source, and provide you the needed tools to make your projects successful.

Knowage's architecture is composed by different layers. The first one or data integration layers, is represented by software modules and technologies that allow to feed Knowage datasets through the integration of heterogeneous data sources like Web Platform or Hadoop ([12]) based systems. This is possible through different kind of technology like RESTful APIs, which allow Knowage to interact with heterogeneous and technologically advanced systems. The data can be standardized in universal datasets or created meta-models capable of integrating multiple data model and therefore multiple types of data can be well integrated in a single reference model. The highest part of architecture or core part of Knowage technology is represented by modules, technologies and tools which represent the analytical drivers of the suite: among these there are components that deal with data mining (which can natively use technologies such as Python ([13]) or R ([14]) languages), and tools such as managing the data through ETL (Extract-Transform-Load) processes or module like analytical reports and charts to much closer to the data visualization part of this technology. Knowage UI, contained in the cockpit concept, can combine a whole series of graphic widgets, added simply by using drag and drop mechanism, which allow to perform many types of graphs on the data, or to interact with they through functions such as the possibility to modify some aspects of the cockpit layouts and so on. The following figure show Knowage's Architecture meeting the different layers and application modules in evidence:

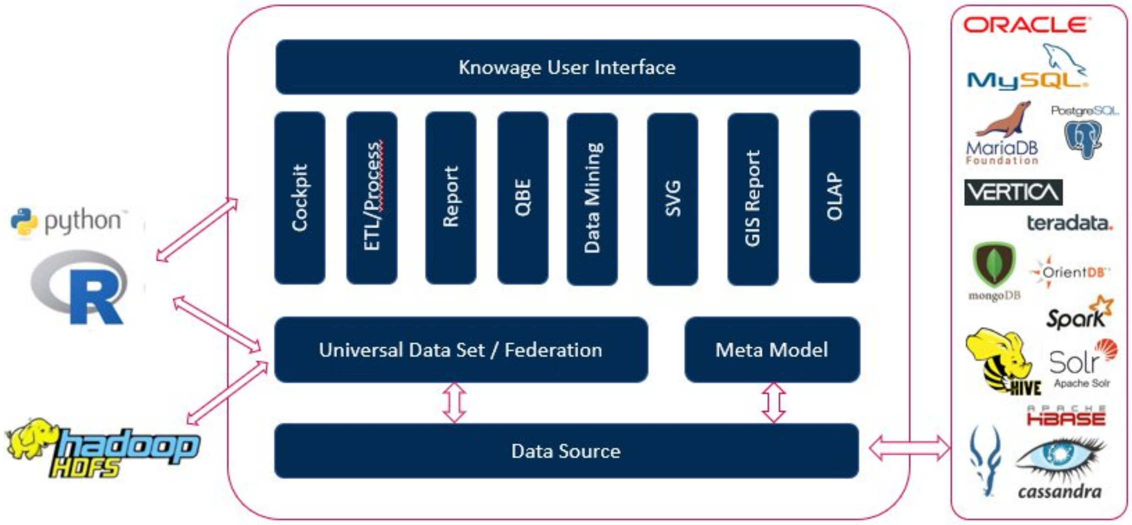


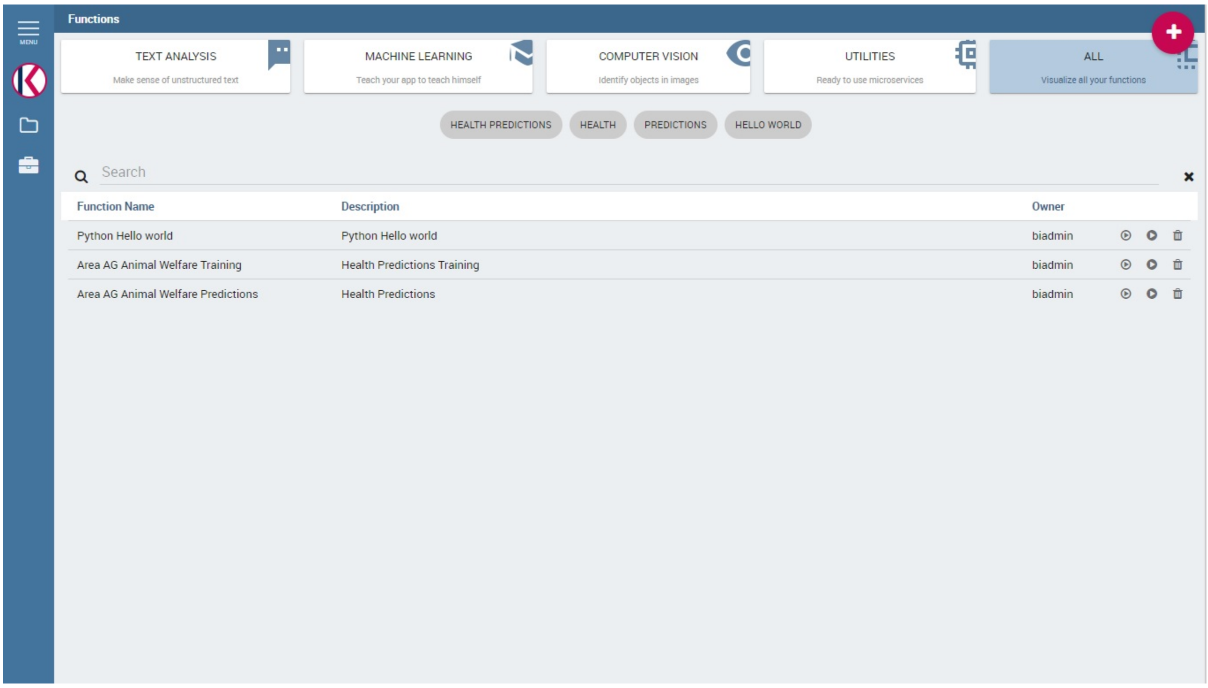
Figure 1: Knowage Architecture

Many of Knowage analytical drivers can be used in DEMETER project, in addition to the intrinsic nature of the technology regarding data visualisation. One of the drivers that could better meet the needs of the project is represented by predictive analytics modules that allow to formulate hypotheses and evaluate the impact interactively through the Dashboard or user interfaces (UI). In fact Knowage could not only offer graphic support for DSS, but also analytical support for those pilots who intend to use the native functions of technology and the 'R' and 'Python' functions for the training and prediction phase of their algorithms. In this way, like described in the next paragraphs, it is possible to have a single environment after extracting value from the data by showing the results of those analyses in general-purpose dashboard.

6.2.1.1 *DEMETER technical integration: use of Knowage function catalogue*

Data mining and analytical features in Knowage can be managed through the function catalogue. The functionality offers an interface in which it is possible to manage a series of functions, giving the possibility to create new ones.

The function catalogue in DEMETER project will be composed through a list of functions that will correspond to the DSS Areas defined during the clustering process of WP4 components. It should be highlighted here, that not all DSS will be part of this catalogue, but only those that deem it appropriate to use Knowage's data mining functions.


MENU
Functions
TEXT ANALYSIS
Make sense of unstructured text
MACHINE LEARNING
Teach your app to teach himself
COMPUTER VISION
Identify objects in images
UTILITIES
Ready to use microservices
ALL
Visualize all your functions
HEALTH PREDICTIONS
HEALTH
PREDICTIONS
HELLO WORLD
Search

Function Name	Description	Owner
Python Hello world	Python Hello world	biadmin
Area AG Animal Welfare Training	Health Predictions Training	biadmin
Area AG Animal Welfare Predictions	Health Predictions	biadmin

Figure 2: Knowage Function Catalogue’s Interface

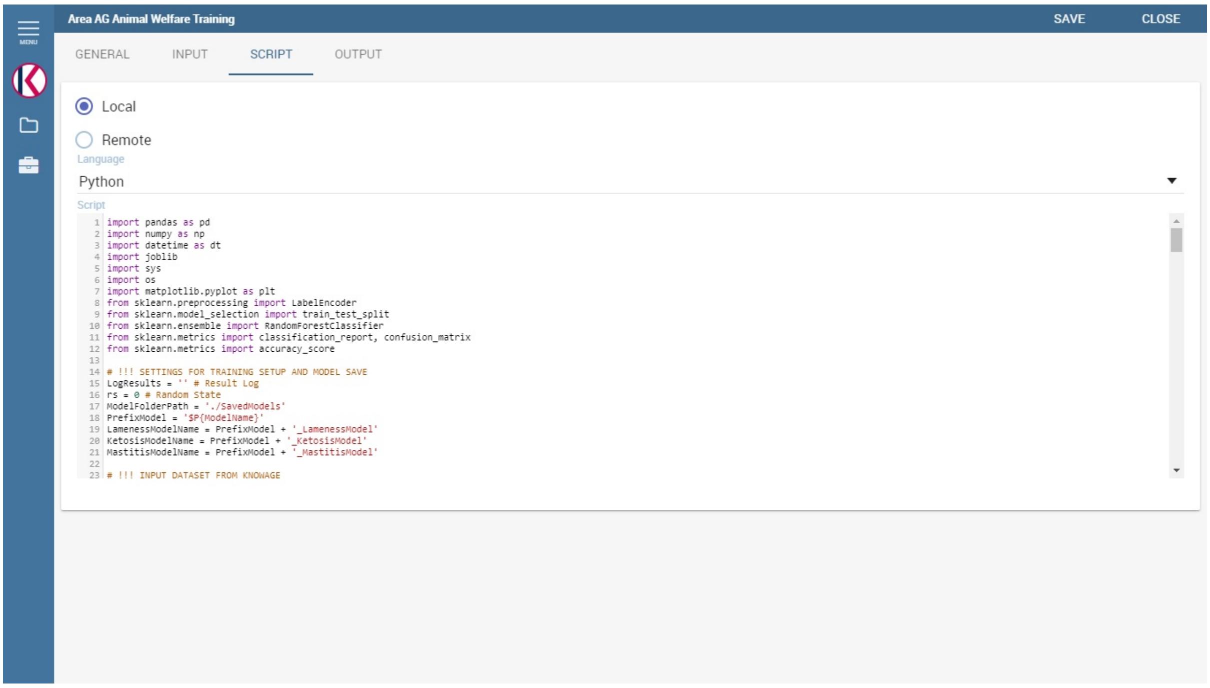
Each function in the catalogue are shown by default. It will be possible to classify functions based on their definition and objectives through the use of categories, while the tags are used to easily refine the search and easily invoke the functions that are tagged with that word.

The categories for the functions depend on the type of function; below is the list of categories with which it is possible to associate a specific function of the catalogue:

- Text Analysis: Make sense of unstructured text
- Machine Learning: Teach your app to teach himself
- Computer Vision: Identify objects in images
- Utilities: Ready to use microservices
- All: Visualizes all functions; this is the only category that cannot be changed or removed.

The categories are mainly used to search for functions and to have a logical grouping relating to their meaning. A function can be performed using the appropriate icon that starts it with predefined values or using the action that starts the calculation after inserting new values for the data. Running a function means running the script in 'R' or Python code. This means that Knowage natively supports these two lengths to allow the insertion of application and business logic to which the function is linked.

The most significant part of the functions catalogue is represented by the Tab Script (which can be found in the definition of all the functions of the catalogue) in which it is possible to insert in R and Python language (still supported by Knowage) the application code for each function; this can feed both the component training phase and the prediction phase.



● Local

○ Remote

Language: Python

Script

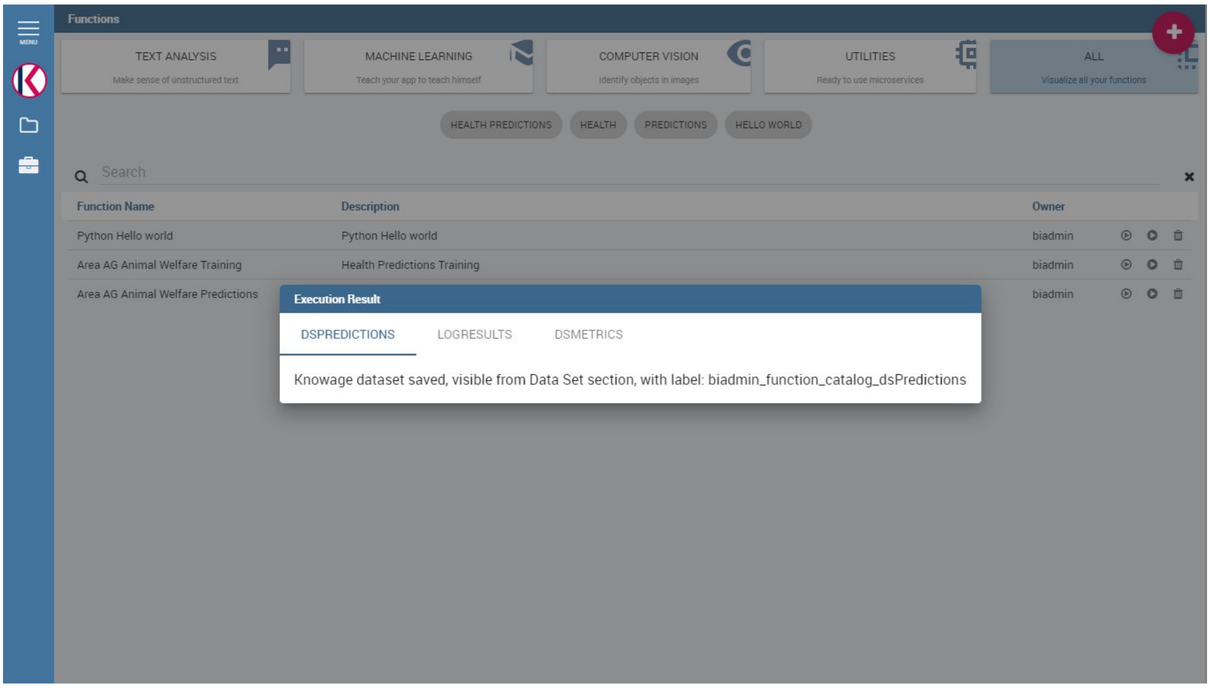
```
import pandas as pd
import numpy as np
import datetime as dt
import joblib
import sys
import os
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.metrics import accuracy_score

# !!! SETTINGS FOR TRAINING SETUP AND MODEL SAVE
LogResults = '' # Result Log
rs = 0 # Random State
ModelFolderPath = './SavedModels'
PrefixModel = '$P{ModelName}'
LamenessModelName = PrefixModel + '_LamenessModel'
KetosisModelName = PrefixModel + '_KetosisModel'
MastitisModelName = PrefixModel + '_MastitisModel'

# !!! INPUT DATASET FROM KNOWAGE
```

Figure 3: Knowage Function Catalogue's Tab Script

In the example shown in the figure above, we see how it is relatively simple to insert the analytical code that will allow the creation or update of a dataset associated then with a Dashboard. The datasets are associated with the function via the output tab which indicates the data container that will contain the results of the processing of the function and therefore of the script. It is also simple to execute the function after adding all the information required in the various tabs as shown in the following image:


Functions
TEXT ANALYSIS
Make sense of unstructured text
MACHINE LEARNING
Teach your app to teach himself
COMPUTER VISION
Identify objects in images
UTILITIES
Ready to use microservices
ALL
Visualize all your functions
HEALTH PREDICTIONS
HEALTH
PREDICTIONS
HELLO WORLD
Search

Function Name	Description	Owner
Python Hello world	Python Hello world	biadmin
Area AG Animal Welfare Training	Health Predictions Training	biadmin
Area AG Animal Welfare Predictions		biadmin

Execution Result

Knowage dataset saved, visible from Data Set section, with label: biadmin_function_catalog_dsPredictions

Figure 4: Knowage Function Catalogue's execute function

Knowage runs the script that executes its logic: the function can be linked to a training dataset of the algorithm or to a prediction one. In both cases, the application will notify the user that the new dataset has been saved with the generated data of the script.

6.2.2 Embedded integration: Knowage native data mining engine

The data mining engine is thought and implemented to supply Knowage with data mining capabilities, but it also enhances OS R and Python with several distinguishing features.

Basically, the integration is done through the Java R package, JRI library and the JPY library for Python. R/Python scripts are written inside the data mining documents template and evaluated server side once the document is executed. Nevertheless, developers can combine several scripts, each one with its own outputs. The main component is the command object, that holds the activation of one script, together with its outputs. There can be many commands, but the one which is executed at document start up, is the one with mode set to auto, whilst all the others will be executed once user clicks on the corresponding element. Each script can run on two kinds of datasets. The first one is the file type, that means users can upload their files, interacting with the GUI. Note that the extension of the file, as well as how to read it (comma separated or tab separated, is header present or not, …) can be specified in the proper Daset tag in document's template. The other type of dataset is the Knowage parametrization and customization dataset. This feature allows to use inside your R scripts data retrieved from many kinds of sources, without ad-hoc R packages utilization. Moreover this is very useful for Big Data data sources (Hive ([15]), Impala ([16]), HBase ([17]), HBase, NoSQL databases as MongoDB ([18]), OrientDB ([19]), Cassandra ([20]), etc.), because not all of them are connectable from R, and if they are, in most cases, R must be installed on their cluster. Therefore, this is the first powerful feature Knowage adds to R. Each Knowage dataset is readable as a CSV file, so developers must apply the proper functions.

Knowage data mining document can perform a set of scripts and visualize them according to the associated predefined outputs, that can be images as well as text. This combination of results, that can be modified on-the fly using variables and shared across the network through Knowage web application, can exploit R workloads. Indeed, Knowage can provide role's privileges to the document's access or execution.

Knowage offers hence the possibility to integrate algorithms like Random Forest directly inside the platform using a feature called "Functions catalog".

Random Forests ([21]) is a supervised learning algorithm. It can be used both for classification and regression. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests creates decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance. Python offers a solid and easy way to set up a random forest algorithm. The main library used is scikit-learn (sklearn). It contains various functions like pre-processing, model selection, metrics, and ensemble. The library joblib is mainly used for saving or import models. Another main library to use is Pandas ([21]) for dataset manipulation within the python script.

Before explaining how functions catalogue works, it is important to do an overview to how upload data inside Knowage. The dataset management panel offers a series of tool able to define datasets among a wide range of types. Moreover, you can add parameters, define scope, manage metadata, and perform advanced operation on datasets. In that case, using the "File" type, it let the user to upload XLS or CSV files from file system and convert the data into a dataset. There is also the possibility to preview the dataset and check the values before saving. The functions catalogue is a powerful framework for managing data mining. When defining a new function, it is possible to choose between three types of input:

- Dataset: Takes values from a Knowage dataset (uploaded through the dataset management panel). It is also possible to ask for the preview so the user can check if the values suit the wished requests.
- Variable: The user can insert one or more variables and match them with values using the dedicated area.
- File: The user is asked to browse folders and upload the wished document remembering to give an alias to it.

Functions catalogue offers a script tab, where an expert user defines the function through the usage of data mining languages R or Python. The engine uses a Java/Python interface, which allows to submit scripts and get result from a Python (or R) environment already installed on the machine where the data mining engine runs. For this reason, Python environment need to be installed on the same machine of Knowage server. If the function takes variables or files as input, it is possible to recall them through their name (as specified in the input tab).

Furthermore, it is important to specify what kind of outputs the function will produce:

- Dataset: The function will return a set of records as a the Knowage dataset way
- Image: the function will return one or more graphics showing the results through bar or pie charts or other kind of visual tools
- Text: The function will return a window containing some text
- File: The function will return a file

Through the functions catalogue framework, it is possible to implement a random forest algorithm, then it is possible to define more than one output for the same function. The input datasets were loaded using the dataset management panel on Knowage.

Two functions are defined using Python script language for training and predictions functions; this approach would be used by all pilots who intend to use Knowage's native support for analytical functions:

- Training algorithm
- Predictions function

The **Training Function** requires as input data a single dataset containing all the necessary values and the respective solutions, to train the algorithm to predict the health status of animals for lameness, mastitis, and ketosis diseases. As a second input, a variable is defined to set a name that will be assigned to the models, saved at the end of the training phase. The input dataset is processed to set up three different training to perform, one for each type of disease. After the training and accuracy test phase, three models are saved (one for each type of disease). The records that have been taken as tests are converted into a dataset containing the results of the predictions made by the algorithm. Accuracy metrics are also converted into a separate dataset. These two datasets are the output of the function, they will be saved as a Knowage datasets and can be seen on the dataset management panel. If the output datasets already exist among the Knowage datasets, then their values will be updated.

The **Prediction Function** requires as input data a single dataset containing all the necessary values, to predict the health status of animals for lameness, mastitis, and ketosis diseases. The second input defines a variable that allows to set the name of the models that will be loaded and used to make the predictions. Right before beginning the dataset preparation process, it is possible to load the data models specified by input, that will be used for making the predictions. The input dataset is then processed to set up three different predictions, one for each disease. At the end of the predictions phase, a single dataset is provided as output, containing all the input values and the predictions made. Even in this case the dataset is saved among Knowage datasets.

6.2.3 Remote integration: Knowage REST Dataset

Before describing the main way of integrating the analytical components that will be produced within WP4 (but also in WP2), it is necessary to introduce the concept of dataset according to Knowage meaning. Knowage dataset is a data provider for analytical documents.

A dataset is a way to read data from different sources and represents the portion of data used by various documents. Suppose you want to create a bar chart showing the sales trend for the current year; in this case you need to pass to the document the total sales amount for each month of the current year. You can create your own dataset uploading an XLS or a CSV file or use a dataset already defined. Knowage offers you also the chance to download open data from WEB thanks to CKAN integration. Moreover, you can create your own and more complete dataset from different sources through the dataset federation. In the following we will describe all these functionalities.

The datasets are classified and categorised according to their types:

- My dataset: datasets created by yourself uploading a CSV or XLS file or creating a query on a business model using the QBE ([23]) interface.
- Enterprise dataset: certified datasets, namely datasets created by the technical/expert users and shared with the end user.
- Shared dataset: datasets created and shared by other end users.
- CKAN dataset ([24]): in this area you can download public datasets and visualize your CKAN datasets.
- All dataset: in this folder are stored all the available datasets, namely all datasets contained in the classes just described.

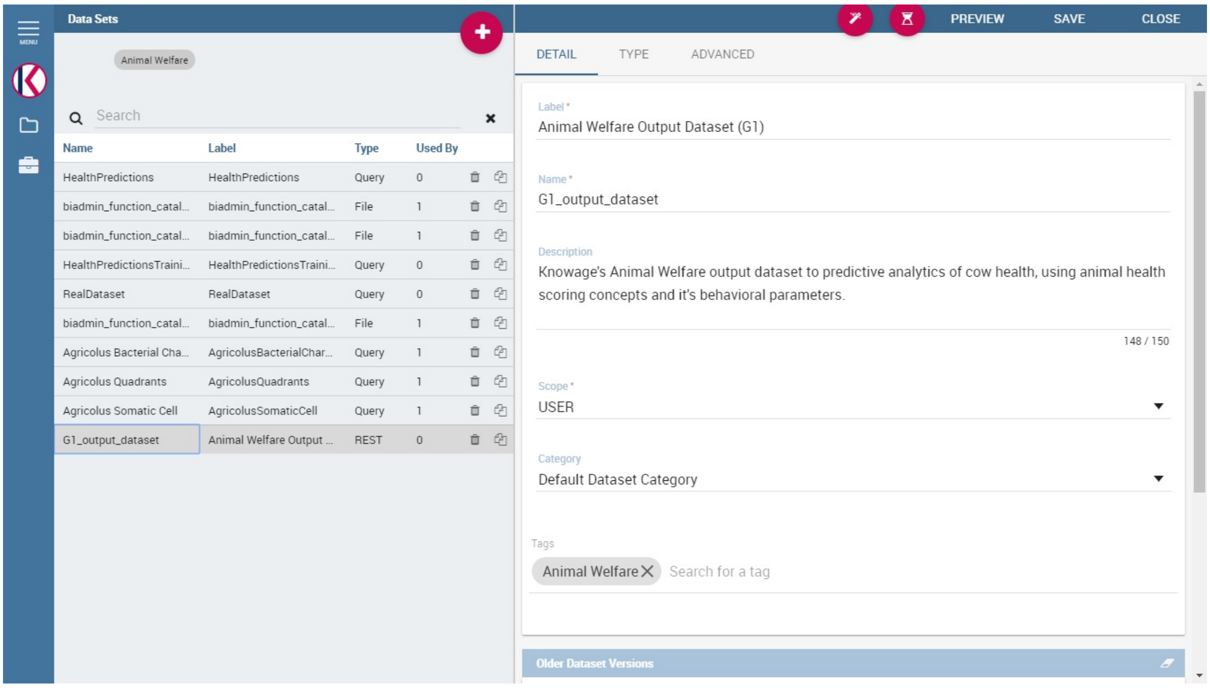
Furthermore, Knowage supports different technologies that allow to read data from heterogeneous sources. For this reason, the complete list of dataset types that can be supported is as follows:

- File
- Query
- Java Class
- Script (Groovy ([25]), JavaScript, Embedded JavaScript or ECMAScript ([26]))
- QBE, query over the metamodel
- Custom
- Flat
- CKAN
- Federated
- REST
- Big Data
- Solr

This heterogeneity certainly represents an added value for this technology, which allows existing platforms to interact with different technologies and data formats; however in the DEMETER project for all the DEMETER Pilots who intend to integrate the visualisation of their DSS the method of integration with the envisaged Knowage technology is represented by the REST dataset. This is due to various reasons including the wide use of REST APIs in the project or of Software as a Service and wide use of components such as microservices; but also and above all since the DEMETER data model (AIM) will be based on the JSON-LD data format that is a pure JSON format.

So, if the pilot prefers to use scripting languages that are not supported by Knowage, or prefers not to use the native support or the functions of the catalogue, all he will do is expose a REST service that contains the output of the data produced by its components. analytical and connect to Knowage through the REST dataset.

Through Knowage dataset management panel, it will be possible to create a dataset using a REST service as the source as shown in the following image:



Name	Label	Type	Used By
HealthPredictions	HealthPredictions	Query	0
biadmin_function_catal...	biadmin_function_catal...	File	1
biadmin_function_catal...	biadmin_function_catal...	File	1
HealthPredictionsTraini...	HealthPredictionsTraini...	Query	0
RealDataset	RealDataset	Query	0
biadmin_function_catal...	biadmin_function_catal...	File	1
Agricolus Bacterial Cha...	AgricolusBacterialChar...	Query	1
Agricolus Quadrants	AgricolusQuadrants	Query	1
Agricolus Somatic Cell	AgricolusSomaticCell	Query	1
G1_output_dataset	Animal Welfare Output ...	REST	0

Label *
Animal Welfare Output Dataset (G1)

Name *
G1_output_dataset

Description
Knowage's Animal Welfare output dataset to predictive analytics of cow health, using animal health scoring concepts and it's behavioral parameters.

148 / 150

Scope *
USER

Category
Default Dataset Category

Tags
Animal Welfare
Search for a tag

Older Dataset Versions

Figure 5: Knowage REST Dataset Panel

By setting up the various parameters correctly, it is possible to send a request to the service that encapsulates the logic and the development of the pilot's analytical component (be it training or prediction), and obtain a response from the service that will contain the output data of the component itself, according to the AIM model of the DEMETER data. Finally, by setting the REST service endpoint it will be possible to convert the data into datasets useful for Knowage and to feed the Dashboards that will be created. It will also be possible to check the output data set of the service using another feature of the Knowage dataset, or the preview function that allows you to check the correctness of the parameters and values it contains.

6.2.3.1 Benchmarking Dashboard

Benchmarking represents the analysis of data comparison. Knowage's benchmarking dashboards highlight the comparison of the data relating to the current year, the previous year and the average of the neighbouring companies, to determine the trend of the analysed aspect: therefore if the data of the current year show growth, improvement or decrease, deterioration, compared to the previous year and to the local average.

The first release of the Knowage dashboards will be focusing on the several parameters: animal health, quality of milk produced, level of nutrition, productivity:

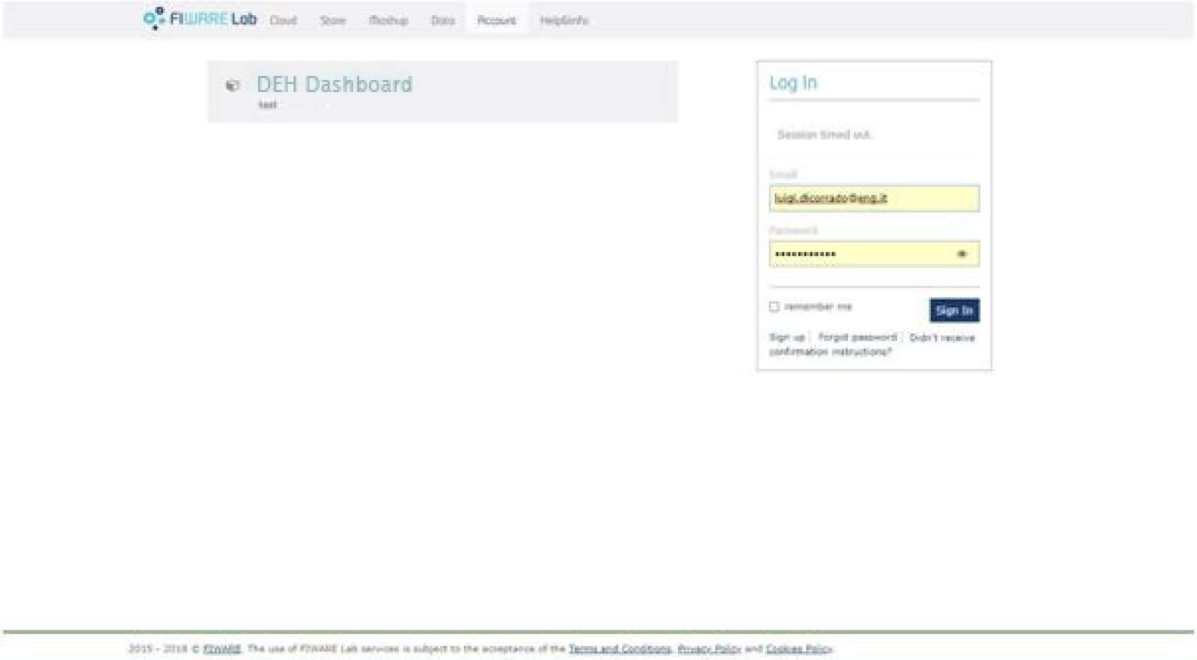
- **Animal Health**: The degree of health of the animal is determined by studying the content of somatic cells and the bacterial load present in the freshly milked milk: these values allow to identify the possible presence of mastitis.

- **Quality of Milk**: The quality and typology of the milk produced is mainly determined by the quantity of fats and proteins.
- **Nutrition**: Animal welfare is mainly given by the quality of its nourishment. The presence of high levels of uric acid in the milk produced reveals the possible presence of ketosis, caused by an imbalance in the daily diet of the ratio between fats and proteins.
- **Productivity**: indicates the amount of milk produced monthly. It is an effect of the aforementioned factors analysed: if the animal is fed well, cared for, it enjoys well-being, then the quality and quantity of milk will be high.

Knowage acquires the flow containing the data and, for animal health, quality of milk produced and level of nutrition, processes them, identifying, through queries, the maximum values of the current year, comparing them with the thresholds of the respective parameters, to determine whether the data falls or not within the normal range.

IN the project development it will be adapted the tested version of the dashboards to other Benchmarking data belonging to other areas: crop yield, water, and nutrient indicators, farming economic and environmental performances. In any case all the dashboard will share the same structure and functionalities.

As with the other DSS Dashboards that will be visible in Knowage, a user must make the login before accessing the specific Dashboard. Knowage will be integrated for the authentication part with DEMETER Security Enablers. The following image shows the Fiware-IdM application that will be connected to Knowage to manage the user authentication phase:

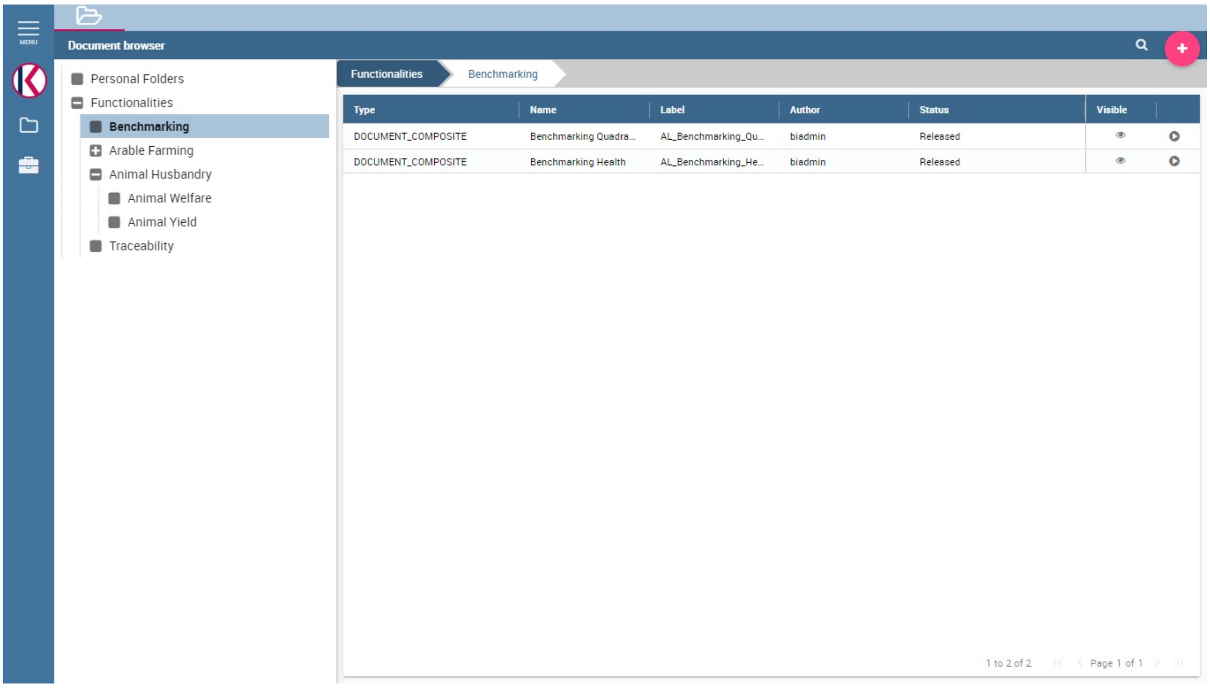


luigi.dicorrado@eng.it

2015 - 2018 © FIWARE. The use of FIWARE Lab services is subject to the acceptance of the Terms and Conditions, Privacy Policy and Cookies Policy.

Figure 6: Knowage Single-Sign-On with DEMETER Security Enabler

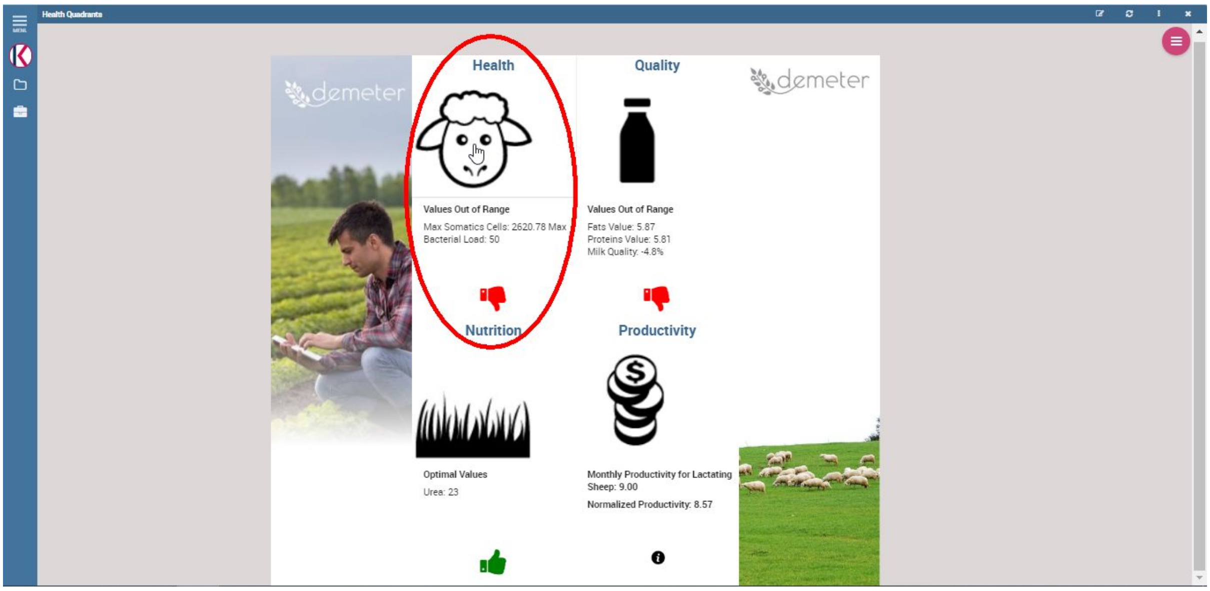
Before accessing the dedicated DSS section, the user can choose which analytical document or Dashboard registered from a drop-down list corresponding to the DSS areas defined within WP4:



Type	Name	Label	Author	Status	Visible
DOCUMENT_COMPOSITE	Benchmarking Quadra...	AL_Benchmarking_Qu...	biadmin	Released	
DOCUMENT_COMPOSITE	Benchmarking Health	AL_Benchmarking_He...	biadmin	Released	

Figure 7: Knowage drop-down DSS list

Benchmarking consists of a summary dashboard, which shows four quadrants, and a detailed dashboard for each quadrant.



Values Out of Range
Max Somatics Cells: 2620.78 Max Bacterial Load: 50

Values Out of Range
Fats Value: 5.87
Proteins Value: 5.81
Milk Quality: -4.8%

Optimal Values
Urea: 23

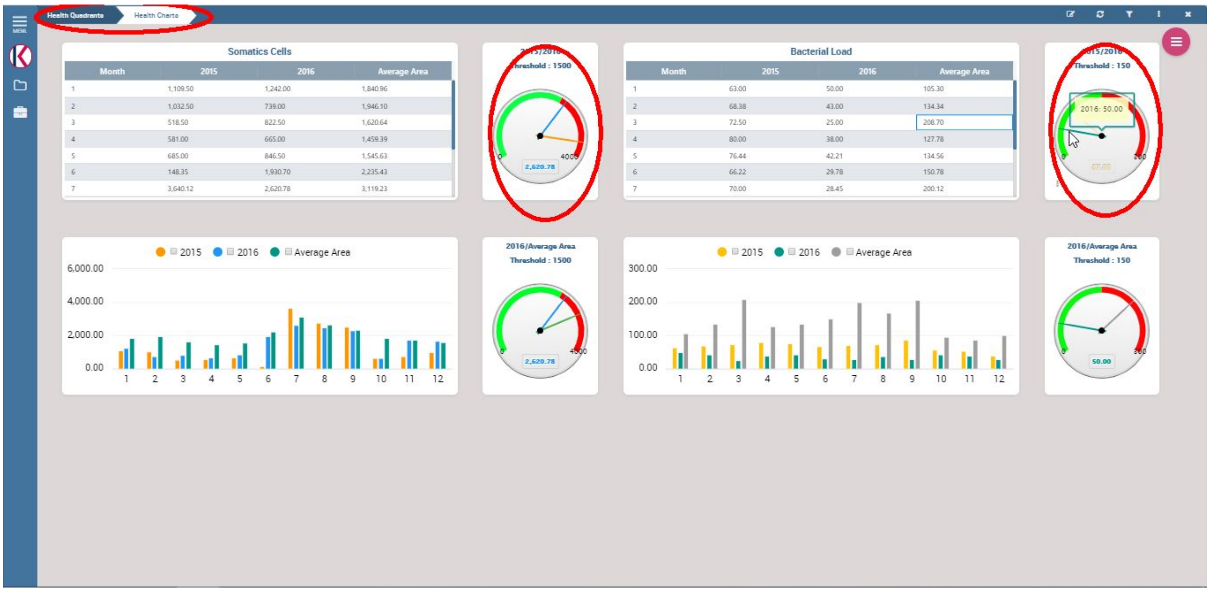
Monthly Productivity for Lactating Sheep: 9.00
Normalized Productivity: 8.57

Figure 8: Knowage Dashboard - Benchmarking Synthetic Quadrant

They expose the maximum values found in the most recent year. If they fall within the normal range, established through standard thresholds or foreseen by regulations, they are considered Optimal Values, vice versa Out of Threshold Values. A separate discussion regarding the productivity analysis, for which obviously there are no thresholds.

The figure 1 shows the health quadrant, where it is possible to observe the maximum values of somatic cells and bacterial load, found in the current year (in the example 2016). The threshold values for the 2 parameters have been set respectively at 1500 and 150. So, the value of the somatic cells is outside the threshold: therefore, the thumb is "towards".

By clicking on the quadrant, you can view the detailed dashboard.



Somatics Cells

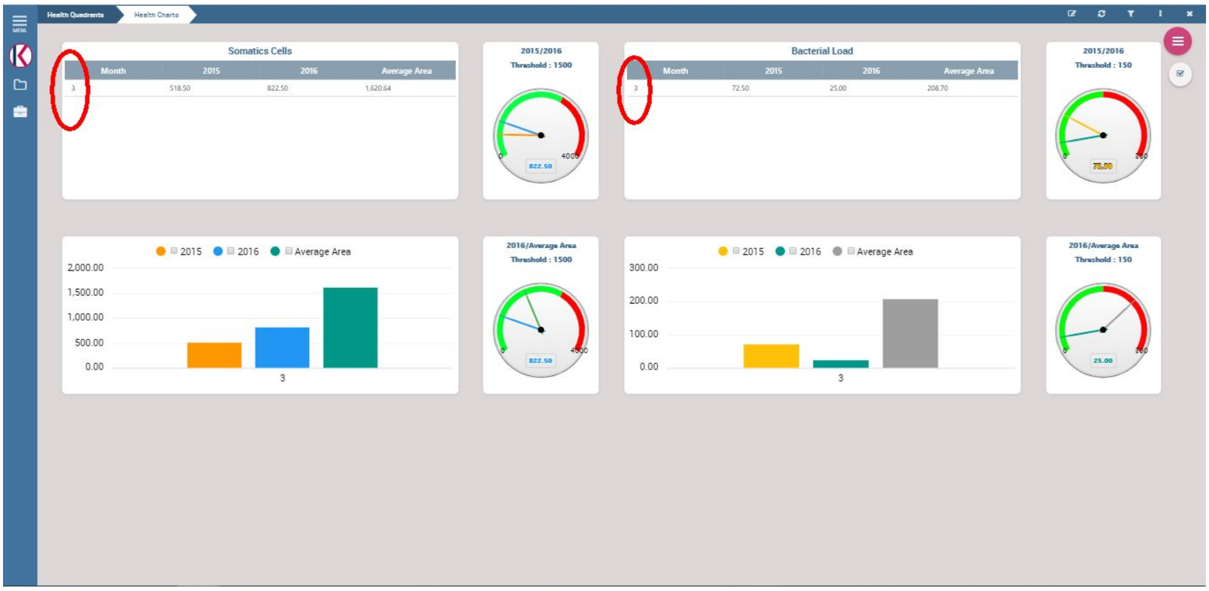
Month	2015	2016	Average Area
1	1,109.50	1,242.00	1,840.96
2	1,032.50	739.00	1,946.10
3	518.50	822.50	1,620.64
4	581.00	665.00	1,459.39
5	685.00	846.50	1,545.63
6	148.35	1,930.70	2,235.43
7	3,640.12	2,620.78	3,119.23

Bacterial Load

Month	2015	2016	Average Area
1	63.00	50.00	105.30
2	68.38	43.00	134.34
3	72.50	25.00	208.70
4	80.00	38.00	127.78
5	76.44	42.21	134.56
6	66.22	29.78	150.78
7	70.00	28.45	200.12

Figure 9: Knowage Dashboard - Benchmarking Detail

The tables and bar graphs show the data concerning the analyses of the years and the comparison with the average values of the data of the local companies. In the Figure 2, the "gauge" graphs show the comparison between the data of the year 2015 and 2016, and between those of 2016 and the average values of local companies. If you do not click on any month in the table, the "gauges" show the maximum values.



Somatics Cells

Month	2015	2016	Average Area
3	518.50	822.50	1,620.64

Bacterial Load

Month	2015	2016	Average Area
3	72.50	25.00	208.70

Figure 10: Knowage Dashboard – Benchmarking selected month detail

It is obtained by clicking on one of the rows of one of the two tables. The "gauges" show the values corresponding to the selected month.

6.2.4 Knowage SDK for iFrame Injection

Knowage SDK ([27]) represent a set of APIs compliant with the SOA architecture, allowing external applications to interact with the Knowage Server and its metadata. The interaction between these two components is possible thanks to Knowage SDK module. The users can display the list of analytical documents which are available on the server and download them, to modify them on their own server.

Knowage SDK is a complete a JavaScript Framework API that helps users to embed parts of Knowage suite inside a web page or to retrieve information about datasets and documents. Because of this policy web browsers do not let a script in a web page to retrieve data from a URL if this URL has a different origin respect of the web page. The origin is different if the URI schema, host name or port are different. That means that if you use Knowage SDK in a different origin from the Knowage server the browser will not let your script to communicate with Knowage. There are two types of functions in Knowage SDK:

- iFrame injection: They inject a Knowage page inside an iFrame
- REST services: They use REST services of Knowage to retrieve some kind of information

For the first type of function same-origin policy is not a problem. For the second one it is. To solve the problem Knowage SDK use two different approaches:

- JSONP ([28])
- CORS ([29])

The jsonp version of these functions is in the same API stack as the functions of the first type (iFrame injection). The CORS version of the same functions (those based on REST services) is found under the same name as the JSONP counterpart.

New Knowage API (v7.1) gives the same functionalities of these functions were developed using CORS instead of JSONP. There are three main advantages on using CORS over JSONP: all the methods are available while in JSONP only GET request can be done; if an error occurs it is possible to manage it with CORS, while in JSONP it is only possible to set a timeout; However, JSONP is supported by all browser while CORS does not work properly in Internet Explorer 8 and 9 (in IE 7 and earlier versions is not supported at all).

To use Knowage SDK functions the third- party application must be authenticated. If the user is already authenticated in Knowage the SDK use cookies to retrieve the session and all functions can be used. If the user is not already authenticated, he must log in. There is the authenticate function for that.

6.3 Pilots DSS integration with Knowage technology

The Knowage technology offers, as described in detail in the sections of this deliverable, different ways of integration with other existing platforms or technologies through both a whole series of data connectors and through the support of embedded analytical drivers. However, the integration of DEMETER Pilot DSS user interfaces will be the main objective of the Pilots technical responsible in the coming months. All the details of this integration will be reported in the second iteration of this deliverable (D4.4) foreseen for M22. This work of DSS user interfaces porting has already started with benchmarking dashboard and will continue also in other DSS areas where a user visualization is foreseen. The UI design has already started with WP4 collection requirements, in which many Pilots have designed the reference DSS interfaces for their Pilot through web mock-up. This process will continue throughout the application development period foreseen by the DEMETER project, where close collaboration is expected between WP4 Partners and Pilots technical responsible.

The integration of the analysis software components or algorithms that produce the output data necessary for the DSS interfaces to work, will connect to Knowage through the REST data sets (or through the embedded functions represented by the analytical drivers made available through

Knowage function catalog); the output indicators extracted from the algorithms will feed the Knowage dashboards for the DSS and the data will be graphed within the cockpit.

This approach involves generating general-purpose dashboard and not Pilot driven, that is, not a mirror copy of the DSS interface that the pilot has so far, but a common interface as much as possible for the pilots involved in a specific DSS area. Figure 11 shows this generic approach for all Pilots who intend to connect their data with the data visualization framework:

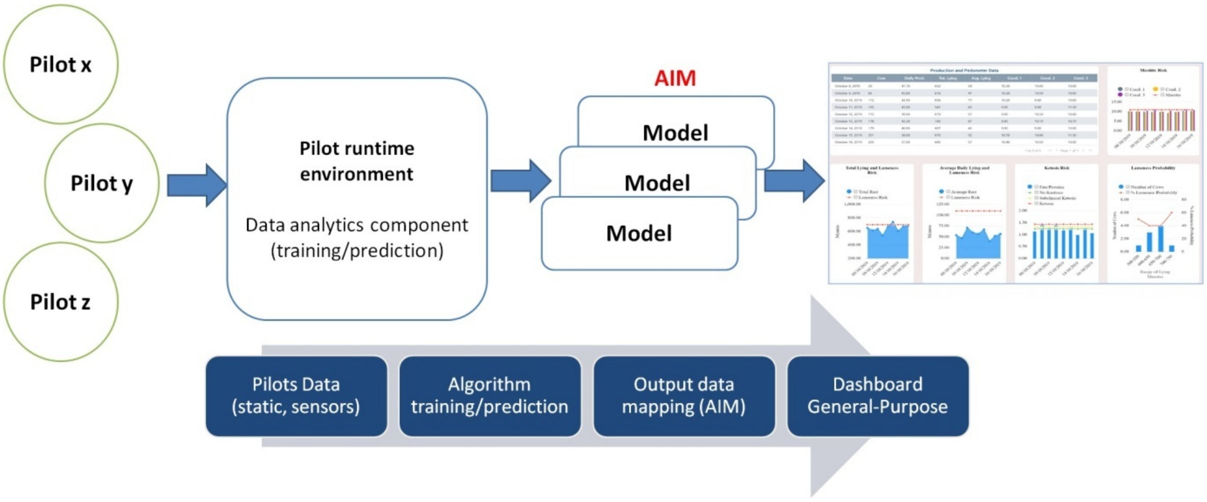

Pilot x
Pilot y
Pilot z
Pilot runtime environment
Data analytics component (training/prediction)
AIM
Model
Model
Model
Pilots Data (static, sensors)
Algorithm training/prediction
Output data mapping (AIM)
Dashboard General-Purpose

Figure 11: Pilot DSS tuning information flow

The Pilots run the analytical components in their environment both in the training phase and in the prediction phase, feeding them with the data pre-washed from the available sources (sensors, databases, static data etc.). Knowage REST dataset services will lock the output of the data produced by the algorithms in AIM format; Knowage depending on the type of service and the output produced, will be able to select the most appropriate DSS interface or dashboard, showing the end user the result of the processing.

The main phases of Pilot DSS integration and Knowage technology can be summarized as follows:

- **Pilot selection**: Make an analysis of possible DSS that may have common views (involving Pilot Technology Providers).
- **Merging data visualisation UI**: Make a match between the proposed user interface (mockup) to standardise the DSS view in a single general-purpose dashboard.
- **Dataset selection**: Selection of datasets that will be used to feed models and dashboards.
- **Widgets and charts selection**: Selection of widgets and graphs/charts that will shape the dashboards (by using Knowage suite primitives).

Finally, DSS document catalogue will be produced which will contain all technological details of DEMETER DSS: the analytical software components for training and prediction phase, user DSS dashboard used to feed algorithm data output in AIM format for end-user visualisation.

6.4 Knowage factsheet

The following table summarize valuable resources linked to the Knowage technology in Release 7.1:

Download page	https://www.knowage-suite.com/site/knowage-download/
Apiary Project	https://knowage.docs.apiary.io/
GitHub repository	https://github.com/KnowageLabs
ReadtheDocs	https://readthedocs.org/projects/knowage-suite/

Docker Container	https://github.com/KnowageLabs/Knowage-Server-Docker https://hub.docker.com/r/knowagelabs/knowage-server-docker/
GitHub Release (Knowage 7.1.8)	https://github.com/KnowageLabs/Knowage-Server/releases
Question & Answers	https://www.knowage-suite.com/qa/
Bug tracker	https://github.com/KnowageLabs/Knowage-Server/issues

Table 37: Knowage factsheet

6.5 Proposed deployment

The deployment of the Knowledge suite will follow the general indications for the Enablers of the DEMETER Project: that is, using docker ([30]) containerization to allow easy software distribution. Knowage will therefore be containerized and all drivers will be able to distribute it within their production environments.

The reference version for all DEMETER pilots is Knowage ([31]) Community Edition v7.1, the open source suite that ENGINEERING takes care of by developing new features and bugs fixing reported by the developers or customers who use this software in their application environments.

However, ENGINEERING will make available on DEMETER central cloud a Knowage Enterprise Edition ([32]) instance for the execution of the project with limited access only to those Pilots who need features not present in the community version.

6.5.1 Pilot infrastructure

DEMETER pilots will have the opportunity to download and install the community version of Knowage in the form of container dockers. This will happen in their cloud or on-premises production environments.

The Knowage docker container at the time of execution will include some basic configurations, such as connectivity to the Security Enabler (Fiware-IDM ([33]))), and a basic structure for DSS Dashboard or analytical documents. The Pilot will only have to connect his analysis services to the Knowage REST Datasets that feed the Dashboards. Obviously, the pilots have to previously test those services so that they produce the correct data for that specific DSS function in the AIM data format, which represents the basic model of all the information that is exchanged between the DEMETER Enablers.

6.5.2 DEMETER central cloud

The centralized cloud solution, hosted by DEMETER central cloud solution, will act as a collector for those Pilot instances that have determined needs that do not meet technological constraints and specific functionalities compared to the Community version of Knowage. In this case it will be possible to connect the analytical services of the Drivers with an Enterprise instance of Knowage where these drivers are present. This will allow to map a wider quantity of DSS solutions and to go against the needs of requests for visualization by the Pilots themselves. Pilots who decide to adhere to this form of configuration will have to expose their services on the internet, so that Knowage can connect the REST Datasets to those services. At that point, the users of those Pilots have access to the Knowage Enterprise cloud instance via login procedure (also this instance will be connected to the central server or federated instance of DEMETER authentication service). The user indicated by the pilot as administrator have a profile on the Knowage application instance that will allow him to independently manage his data, connect his services and view the results in the DSS Dashboard.

This possibility, however, will remain limited to only exceptional cases where the community version will not be sufficient to cover the functional and technological needs of the pilots, otherwise the deployment mode of the Knowage visualization component for all Drivers will be to download the component and run it in their application environments., i.e. the installation of the Community instance of Knowage all inside your production environment.

Furthermore, DEMETER central cloud solution can meet the needs of the project, such as its execution, or even demonstration events such as demos or workshops organized in the environment of the DEMETER consortium.

7 Stakeholders Open Collaboration Space (SOCS)

7.1 Overview

DEMETER represents an answer to the need highlighted by the EC (DG Agriculture and Rural) that through EIP-AGRI ([34]), fosters the knowledge exchange and lessons learnt in the agricultural, forestry and rural development sectors as well the “interactive innovation model” among all the interested stakeholders.

The interactive innovation model is implemented through the MAA (Multi Actor Approach). DEMETER implements the MAA across the full chain, from farmers to service advisors and suppliers. In this MAA, suppliers cover the full diversity of providers including ICT, data sources, knowledge, developers, software, and hardware providers.

The MAA is implemented through a complete set of mechanisms structuring the human interaction with all stakeholders and is supported through digital spaces, with each space catering to different phases of the interaction leading to the co-creation and deployment of new solutions at the farmers. Therefore, DEMETER delivers three spaces, the **Stakeholders Open Collaboration Space** (SOCS) and the Agricultural Interoperability Space (AIS) connected by the DEMETER Enabler Hub (DEH).

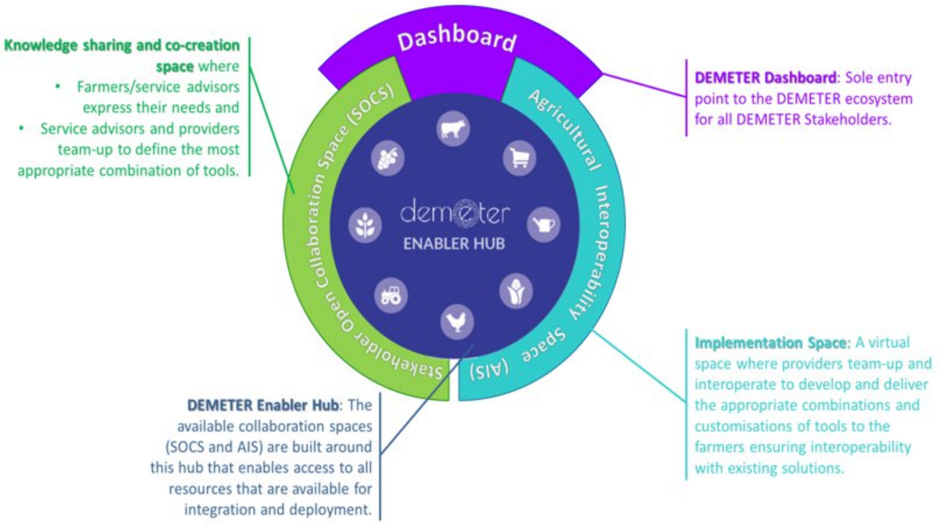

Knowledge sharing and co-creation space where
• Farmers/service advisors express their needs and
• Service advisors and providers team-up to define the most appropriate combination of tools.
Dashboard
DEMETER Dashboard: Sole entry point to the DEMETER ecosystem for all DEMETER Stakeholders.
Stakeholder Open Collaboration Space (SOCS)
Agricultural Interoperability Space (AIS)
demeter
ENABLER HUB
Implementation Space: A virtual space where providers team-up and interoperate to develop and deliver the appropriate combinations and customisations of tools to the farmers ensuring interoperability with existing solutions.
DEMETER Enabler Hub: The available collaboration spaces (SOCS and AIS) are built around this hub that enables access to all resources that are available for integration and deployment.

Figure 11 DEMETER Digital Space

To get to the focus of this section, the **SOCS** is a space **dedicated to all stakeholders** (farmers, advisors, and suppliers) where they can **collaborate, share best practices and participate to co-creation processes**. This collaboration space makes a farmer need visible to advisors and developers and conveys the information coming from the farmers as input to select the most suitable resources registered in DEH to be used to build the optimal solution. The SOCS will act as a facility intended to support knowledge sharing, lessons learnt and co-creation of solutions, based on human to human interaction, communication, and technology information.

The SOCS aims to “put farmers fully in control of their needs, of their choices, of their speed of adoption of solutions, of their data” and represents a response to their need **to be supported when they have to choose between different solutions, to be aware about the compatibility/interoperability of a solution with their legacy systems and to trust in data sharing**.

The SOCS will be based on well- established technologies, further described in Annex A, which give a set of basic services and tools to create, share and find contents, information, knowledge, experts, and ideas as well as to effectively and productively manage collaboration and work together. This deliverable intends to focus on the first release of the SOCS, but also give some insights about the methodology (connected with MAA) that will be followed for the next two releases that will be available at M22 and M38.

7.2 The interactive innovation model

As already said, in DEMETER the interactive innovation model is implemented through the MAA which allows to “put farmers fully in control of their needs, of their choices, of their speed of adoption of solutions, of their data”, “reverses the relationship between farmers and providers” since “farmers define their needs together, individually or with advisors”, “suppliers collectively and individually, share the responsibility of elaborating one (or more) solutions to answer the need” and “the solutions made are comparable based”.

In the overall agriculture scenario, farmers must face a lot of difficulties related to:

- **assess the Return on Investment** ascribable to digital innovations
- **trust** the technology and IT providers
- **trust in data sharing**
- understand about **compatibility/interoperability** with current solutions
- choose between different solutions without a clear logic between benefits, features, and economic models
- **being locked in with one vendor**

In this scenario, the SOCS Platform intends to represent a support facility that helps farmers to become more competitive by improving their business/production processes as well as products quality and yield by means of digital technology. The SOCS indeed will include a wide range of features that, together, deliver the **Knowledge-driven services** and **Collaboration Services** to the space users, structuring the human-in-the-loop dimension of DEMETER.

It is worth saying that also the design and development process (further details in 7.3.3) of the SOCS will be influenced by the MAA. To this end, firstly a **survey** was organized to collect interests and feeling from pilots with respect to social networking and co-creation features. The results of this survey allowed to determine the prioritized features to include in the first version of the SOCS (SOCS_v1). Secondly, results from a **platform vision workshop** (organized in the context of WP7) and feedbacks from WP7 and WP4 on the SOCS_v1 demo will allow to better define the identified features and tailor a user journey to end up in SOCS_v2 at M22. Finally, SOCS_v3 will be available at M38 and will include a refinement of SOCS_v2 services and the final version of the application to support the co-creation process grounded on **WP7 collaboration**.

7.2.1 The SOCS Survey

This deliverable focuses on the prototype version (SOCS_v1) whose implementation took as reference the answers collected from the SOCS survey. This survey was prepared by ENGINEERING and reviewed by FRAUNHOFER and T4.5 members. All the results collected through the survey are gathered in this deliverables and in the next phases will be merged with results of DEMETER Vision workshop to take into account of them for the next versions of this deliverable (D4.4 and D4.5).

7.2.1.1 Scope

This survey scope was to understand the feeling that pilots have with respect to the use of the social digital space and to define the basic SOCS features. The SOCS survey, indeed, proposed a list of possible social networking and co-creation features based on an ENGINEERING asset and aimed at finding a prioritization for the implementation of those features, moreover it gave the possibility to add additional comments on proposed features and suggestions for any other missing features.

7.2.1.2 Participants

Since SOCS is a space for all DEMETER stakeholders, pilots' members were considered as main SOCS survey recipients. The survey was sent to all pilots' members by means of the online survey-management system, EU survey ([35])) and 17 answers were collected from Pilot leaders, that can be considered representatives of all the pilot members. As already said, this survey, conducted at February, aimed to understand the feeling about the basic features of the SOCS, more end user centred activities will take place in collaboration with WP7 later.

7.2.1.3 Results collection

As mentioned, pilots were asked to answer to open and close questions. The survey structure and content are available in Annex B.

Thanks to the open questions, it was possible to know if pilots interested to participate in the social space within the context of DEMETER project, the main objectives that could be addressed through the social space in the context of DEMETER project and the main concerns about the use of a digital space by farmers.

Indeed, there were a good number of respondents providing inputs for the section on general questions on social digital space.

It can be identified that a few number (4/17) have already available a Social Digital Space of innovation, with different levels of development and activities considered, which is also a question for future discussion to have a common understanding on what we can consider a social digital space for innovation and minimum requirement to harmonise its future development adapted to the different realities occurring within DEMETER.

One of the respondents mentioned that they have a Communications manager, that manages a dedicated web page, the social networks profiles (LinkedIn and Facebook), and they also promote communications activities and organise Hackathons. Another respondent is more oriented to make available different tools (internal and external) to promote online collaborations and discussions. Another respondent mentioned in relation with the social space for innovation the SmartAgriHubs H2020 project and the last one reported that their social space for innovation is under construction, it is linked to a farmers organisation and its scope is to provide information about innovative projects (managed also by operational groups of the EIP-AGRI) in the agricultural sector.

These social digital spaces for innovation have different functionalities that are fully aligned with the Multi-actor approach fulfilment:

- Promotion and sharing of mutual information
- Development of a common format to make projects available for knowledge exchange purposes

- Development of an internal chat to promote discussions and knowledge sharing among the participants
- Capacity development
- Web conferences and other events to allow the interaction between the different participants

It is also of interest to highlight the capacity of the social spaces of innovation to act as a one stop shop for innovation integrating all the possible actors in the area/region following the connecting development within the SmartAgriHubs project.

A good number of respondents stressed the interest to participate in Social spaces of innovation within the context of DEMETER project however there are few concerns related to develop this participation based on farmers expressed needs which relies on the MAA concept for the development of a co-creation process all along the project.

Finally, the main objectives that could be addressed by the social spaces in the context of DEMETER:

- Assessment of real farmers needs to develop practical solutions rather than pushing technologies.
- Work with other technology suppliers to enable and co-create best solutions set for customers.
- Research, testing and validation of agriculture solutions based on common standards.
- Increase agriculture technology knowledge and awareness among Agricultural Advisors community as they could act as innovation brokers providing knowledge and innovation to farmers and in the way back providing the grassroots needs for research and technology developers and suppliers.

Finally, the survey’s results highlighted that some farmers are not very skilled in using computers and they would need communication in local language.

The closed questions were organized in three classes and were aimed to find a prioritization for the proposed features related to:

- Social Networking
- Co-creation
- SOCS dashboard

To summarise the answers obtained to the closed questions, the following three charts (one for each class) show the aggregated results. From those charts it is possible to conclude that the proposed features were considered by the most as “essential” or “desirable” meaning that were all considered as features to be included in the first version of the SOCS (SOCS_v1).

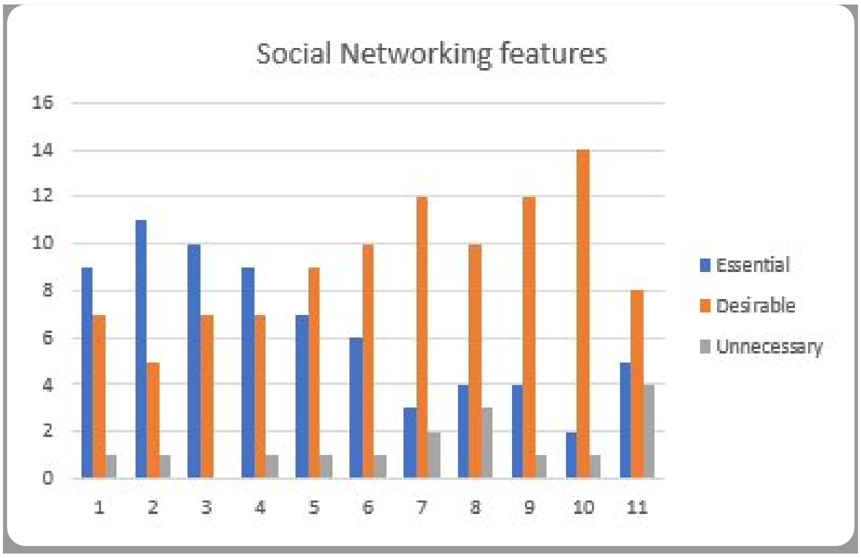


Figure 12 Social Networking features aggregated answers

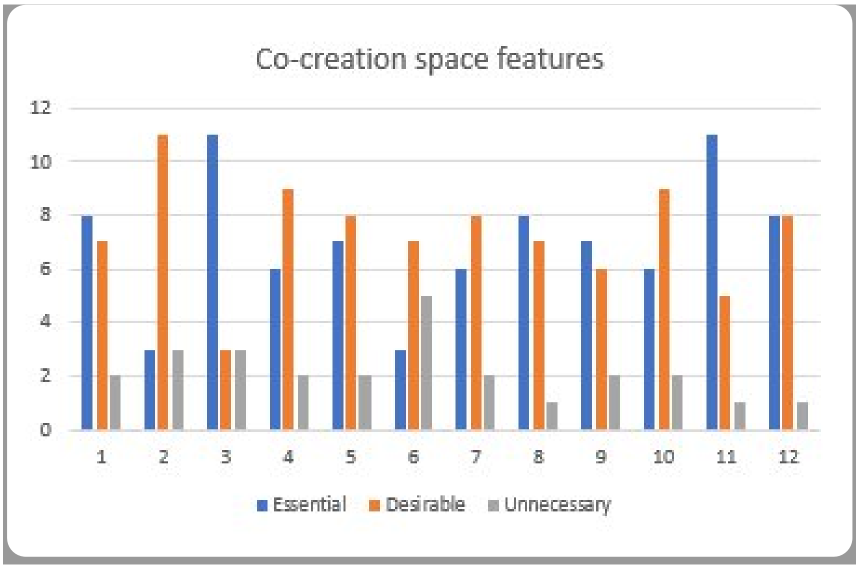


Figure 13 Co-creation space features aggregated answers

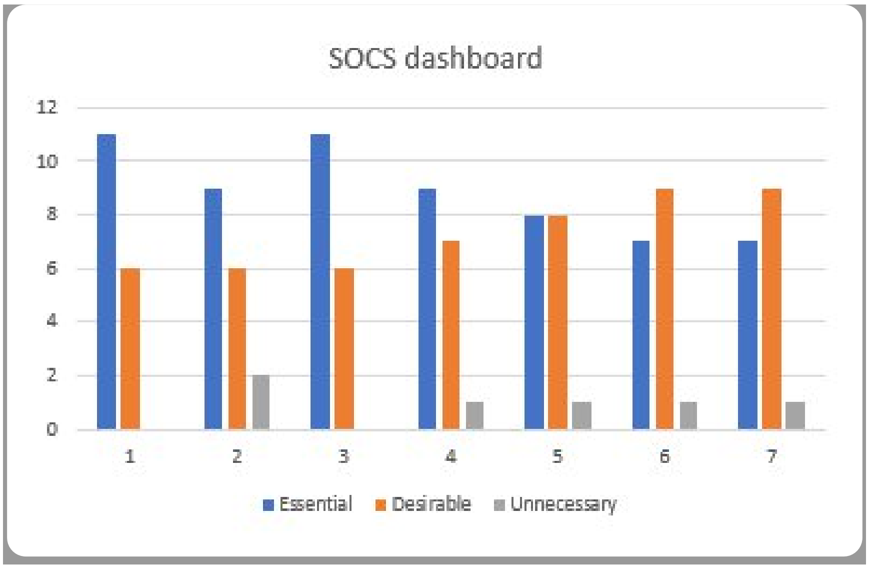


Figure 14 SOCS dashboard features aggregated answers

7.2.2 The interaction with WP7

To ensure the user-centred development, the SOCS survey is just the first measure adopted. Secondly, the process of developing a vision scenario was initiated in the context of WP7. This process serves to sharpen the benefit of DEMETER´s Digital Spaces and bring the idea closer to users who have less technical expertise by using an easily understandable language.

A strict collaboration between T4.5 and WP7 will enable the upgrade from SOCS_v1 to SOCS_v2. The first results from the interaction with WP7 will convey in deliverable D4.4 "Decision Enablers, Advisory Support Tools and DEMETER Stakeholder Open Collaboration Space – Release 2" due at M22.

7.3 SOCS: The Innovation and Collaboration platform for DEMETER

The relationship between farmers, society and the rural world is changing due to diminishing resources, growing populations and pressure on the environment. It is for these reasons that farmers need new knowledge and innovative ideas to develop and manage smarter and more sustainable production systems. Approaches to knowledge exchange, learning and innovation in agriculture are rapidly evolving and the emerging need is to adapt to this new context. Nowadays, knowledge is co-created by farmers, scientists, suppliers, advisers, enterprises, no profit and associations, etc. and, through new ways to share knowledge and expertise, is possible to keep agriculture and food production competitive and rural areas vibrant in the 21st century.

DEMETER project attempts to play an important role: to ensure that farmers in Europe can fully benefit from digital innovations aiming at upgrading solutions, improving processes, and adapting business models to the digital age.

7.3.1 The Platform Users

The **SOCS is a space dedicated to all DEMETER stakeholders:** farmers, advisors, and suppliers.

Farmers can use the SOCS to define their needs together, individually or with the support of advisors, while IT providers, collectively and individually, share the responsibility of elaborating one (or more) solutions to answer farmers need.

Farmers	Advisors	IT providers
• Define their needs. • Access to advisor support. • Access to potential collaborators/new partners. • Co-design new solution based on their needs. • Share experiences. • Access to agricultural experiments. • Exchange knowledge. • Joint events.	• Access to potential collaborators/new partners and customers. • Increase knowledge. • Joint events. • Access to success stories. • Support the farmers in the design of new solution that match farmers' need.	• Learn about new software/networking standards, technologies and methods that are specific to the agriculture domain, while examining their benefit and applicability. • Access to potential collaborators/new partners and customers. • Access to new technology. • Co-design solution with farmers and advisors. • Increase knowledge. • Access to success stories. • Joint events.

Table 38: SOCS users and needs

7.3.2 The SOCS services

The objective of the **SOCS** is to enable a space **dedicated to all stakeholders** where they can find services to **collaborate, share best practices and participate to co-creation processes.**

The services available may be categorized under three pillars:

- Collaboration space: Concerned with a set of tools useful to share, validate ideas and co-design new solutions
- Catalogues: Concerned with a common repository for human profiles and competencies, to enable to discover who is part of the network and look for collaborators/partners in the new solution design
- Knowledge Management: Concerned with provide online access to information and materials and provide content-based (data, information, knowledge) and organizational outputs (e.g. advice, answers, facilitation, practices)

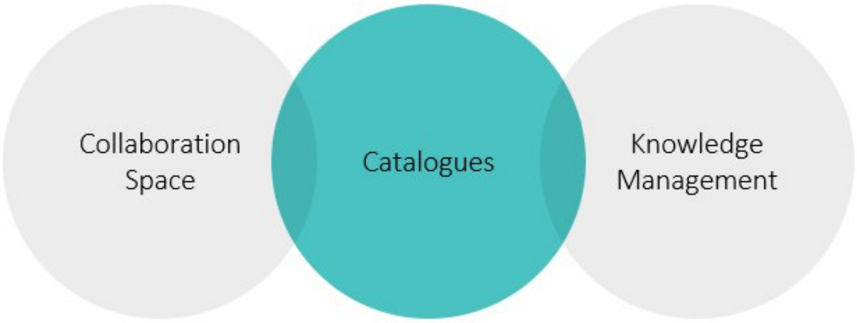
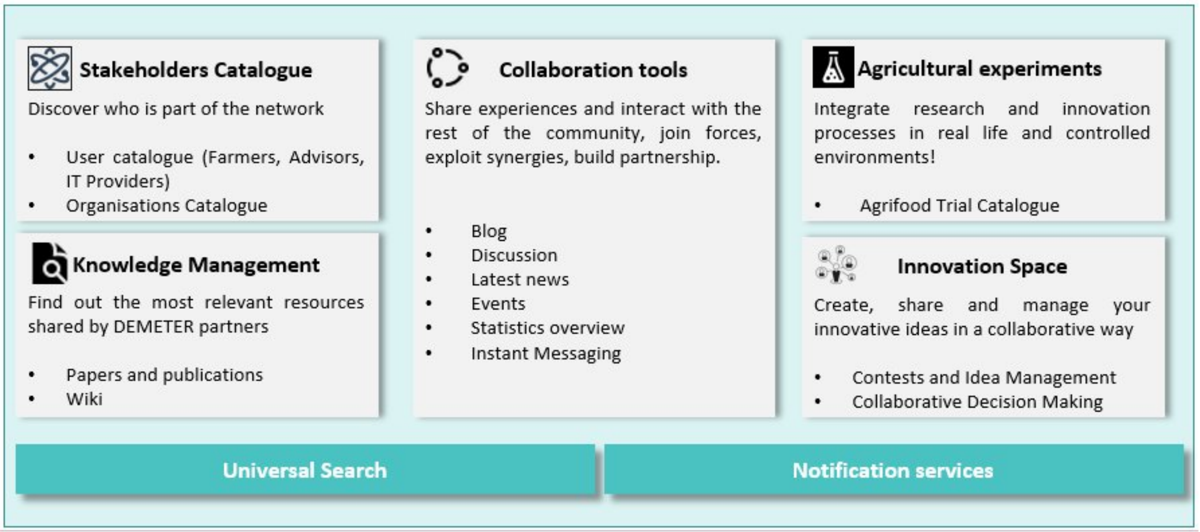


Figure 15 SOCS main components

In the following picture, more details in relation with the SOCS portfolio of services are shown. Moreover, cross functionalities support the general mode of operation, these are: Universal search functionalities and notification services.



Stakeholders Catalogue

Discover who is part of the network

- User catalogue (Farmers, Advisors, IT Providers)
- Organisations Catalogue

Knowledge Management

Find out the most relevant resources shared by DEMETER partners

- Papers and publications
- Wiki

Collaboration tools

Share experiences and interact with the rest of the community, join forces, exploit synergies, build partnership.

- Blog
- Discussion
- Latest news
- Events
- Statistics overview
- Instant Messaging

Agricultural experiments

Integrate research and innovation processes in real life and controlled environments!

- Agrifood Trial Catalogue

Innovation Space

Create, share and manage your innovative ideas in a collaborative way

- Contests and Idea Management
- Collaborative Decision Making

Universal Search

Notification services

Figure 16 SOCS Portfolio of services

7.3.2.1 *“Access to Stakeholders catalogue” services*

In case a new application requires specific knowledge, a collaborative process may be initiated to find the right competences. This is the reason why it is necessary to develop a common repository to discover human competencies and companies core activities, to enable partners’ selection.

This section is composed by two different applications:

- **Users catalogue**: A collection of platform users, characterized by their skills and competencies (according to a predefined vocabulary), belonging to specific organisations; users’ role, skills, activities, etc. are detailed in this section. A table containing a list of all users is shown, and for each user a detailed description is available.
- **Organisations catalogue**: An application used to define organization main characteristics and offered competencies, described according to a predefined taxonomy for organization description; their main characteristics such as application domains, competencies, projects they are involved in are detailed here. A table containing a list of all partners will be shown, and for each partner a detailed description will be available.

7.3.2.2 *“Access to Knowledge” services*

Access to “Knowledge services” include online access to educational information and materials and consultancy services related to different thematic domains.

It will be a multimedia library based on three main services:

- **Papers and Publications**: It represents a knowledge base, structured according to partners’ needs; here it will be possible to find paper or publications related to agriculture domain.
- **Wiki**: It is a collaborative tool to share user’s knowledge in a formal structured way, making it available to all Platform members.

7.3.2.3 "Access to Agriculture Experiment" services

Farmers can get inspiration for their ideas, by analysing the existing experiments. It is possible to take advantage of the best practices and exploit test open results. The objective is to support farmers and advisors in understanding new experimental and technological trends in agriculture domains and explore and test new technologies that can be potentially applied to processes or products, with a "hands on" approach.

The main service here is the **Agrifood trial catalogue**, which is a showcase of successful agricultural cases and DEMETER pilots. Farmers and advisors that have implemented products or services, utilizing the enablers provided by DEMETER, and that have been successful in experiments tell their story (initial problem, actors, processes, solution, technological partners involved, technologies, results achieved expressed with KPI, contact information of the trial etc.), to be an example for other farmers.

7.3.2.4 "Access to Collaboration tools" services

"Access to Social Networking" services will offer the opportunity to connect and interact with all individuals and organizations contributing to the ecosystem. Social networking spaces have been created to facilitate communication between all the actors accessing to the SOCS Space, sharing information in the easiest way. It is possible to find:

- **Latest news**: This section collects recent information internal to DEMETER or belonging to organisations within the consortium.
- **Statistics overview**: This includes descriptive information on the individuals as gender, age, education level, professional position, pages viewed, services accessed, etc. The results could be used to make an analysis on the usage of the platform and eventually to better tailor the SOCS Space in terms of features and functionalities.
- **Blog**: It represents an informal way to share and publish different types of contents (e.g. news, experiences, observations, opinions, etc.); posts are displayed in reverse chronological order, so that the most recent post appears first, at the top of the web page.
- **Discussion**: It is a dedicated space for informal debate; discussions represent consideration or examination by argument, comment, especially to explore solutions, enabling consideration of a question in open and usually informal debate.
- **Calendar & Events**: It represents a useful solution to share information about interesting events, project meetings, etc.
- **Instant Messaging**: All the members of the Platform can use an IM service to chat with the other online users.

7.3.2.5 "Access to Innovation Space" services

"Access to Open Innovations services" gather a set of tools useful for SOCS users to share and validate their ideas, to co-design new solution. This set of applications has a twofold perspective: for farmers to express their needs and for advisors and IT providers to team up to define the most appropriate combination of tools.

- **Contests and Idea Management**: This application will be used to start a competition to involve other people interested in a particular subject, and will enable users to participate to contests, creating and sharing their ideas in a collaborative creation way. Through this co-creation process farmers can express their needs and suppliers can provide solutions for their satisfaction.

- **Collaborative Decision Making**: It gathers a set of tools and voting mechanism useful to encourage user's opinions and aiming to take a well-balanced decision reaching group consensus.

7.3.2.6 *Cross Functional Services*

There are two features of the SOCS that represent important cross functional services:

- **Universal Search:** The platform includes several search features to help users quickly find what they are looking for, or just to browse through content, people, and groups. With this service (always on top of the user interface) it is possible to search for specific words, to find content by categories or tags or restrict to certain data ranges creating logical expressions.
- **Notification services:** The Notification services are used to send a notification to user. The notification can be sent to the inbox section of the platform or to the inbox of the user's email account based on user preferences.

7.3.3 SOCS development planning

The **development process** and release of the SOCS will follow an **iterative approach**. The SOCS will be released in its first version at M10 as instance of OPENNESS (SOCS_v1) that is a result of an ongoing private research project under development at ENGINEERING R&D Department. It represents a platform fully developed using open source technologies and leveraging on relevant results from research fields such as Open Innovation, Collective Intelligence, Enterprise Social Software. OPENNESS gives a set of basic services and tools (forum, blog, wiki, document library, calendar, etc.) to create, share and find contents, information, knowledge, experts, and ideas as well as to effectively and productively manage collaboration and work together.

The prototype version (SOCS_v1) will be shared with WP4 and WP7 "Multi-Actor Ecosystem Development". Feedbacks from WP4 and WP7 along with results from DEMETER Vision workshop will be taken into account to address user journey, interaction with WP7 will allow to design a first version of the co-creation application and all this will end up in the second version (SOCS_v2) available at M22 (D4.4 – Decision Enablers, Advisory Support Tools and DEMETER Stakeholder Open Collaboration Space – Release 2). The SOCS final version (SOCS_v3) will be available at M38 (D4.5 – Final Release and Support Report for Decision Enablers, Advisory Support Tools and DEMETER Stakeholder Open Collaboration Space) and will include a refinement of all SOCS services and the final version of the co-creation application.

The following table is used to outline the SOCS development planning. For each of the selected features is shown the development scope meaning in which iteration the initial, refined, and final version of each service is planned.

Service	Scope		
	SOCS_1	SOCS_2	SOCS_3
Stakeholders Catalogue	Initial version	Refined version	Final version
Organisations Catalogue	Initial version	Refined version	Final version
Papers and publications	Initial version	Final version	
Wiki	Initial version	Final version	
Experimental facilities	Initial version	Refined version	Final version
Latest news	Initial version	Refined version	Final version
Blog	Initial version	Final version	
Discussion	Initial version	Final version	
Statistic overview	Initial version	Refined version	Final version

Service	Scope		
	SOCS_1	SOCS_2	SOCS_3
Calendar & Events	Initial version	Final version	
Instant messaging	Final version		
Contests and Idea Management		Initial version	Final version
Collaborative Decision Making		Initial version	Final version

Table 39: SOCS development planning

8 Conclusions

This document has been devoted to three main tasks:

- develop a self-service dashboard framework capable of integrating a number of service frontends, transformation operators and visualisation widgets.
- develop generic enablers as open components that accelerate the development of sector or pilot specific decision support systems.
- develop cloud-based collaboration tools that enable and support collaboration among different stakeholders in a pilot domain.

From the definition of the different DSS components, several advancements have been put forward. These components have evolved from their initial definition (started in D4.1) and, in D4.2, their description has been extended by adding links to WP2 and WP3 (by means of the related Generic Enablers), a draft development plan has been prepared and an initial API has been proposed. Alongside these aspects, responsible and interested partners have been put on the table. Future activities on this topic will be dedicated to extending the level of detail of the development plan as well as freezing the API together with fostering the collaboration across the different areas forming DEMETER DSS.

Regarding the other two important points reported in this deliverable, the first one is the data visualization module with which users will be presented with intuitive meaning thanks to Knowage, developed by ENG. Knowage is the technology selected to meet this need in DEMETER and next activities will seek to integrate with the outcomes of T4.2 (Benchmarking) by offering a common space for the farmers. The second point deals with the implementation of the SOCS, where initial steps have been taken to provide a description of this space where farmers and related stakeholders can collaborate, share best practices, and participate in collaboration activities. This space makes a farmer's needs visible to advisors and developers. Next actions under this activity will be focusing on boosting the usability for non-technical users.

Planned activities are constantly monitored during the weekly WP4 conference call. All partners will be encouraged to share all their source-code and related tasks in the DEMETER GitLab. Further advancements will be reported in future deliverables, namely Deliverable 4.4 "Decision Enablers, Advisory Support Tools and DEMETER Stakeholder Open Collaboration Space – Release 2" due in M22 (June 2021).

9 References

[1] Knowage suite. https://www.knowage-suite.com/site. Online, accessed June 2020.
[2] Mel-Frequency Cepstral Coefficients (MFCCs). https://en.wikipedia.org/wiki/Mel-frequency_cepstrum. Online, accessed June 2020.
[3] Bikakis, N.: Big Data Visualization Tools. ATHENA Research Center, Greece.
[4] Association, B.D.V.: Big Data Value Strategic Research and Innovation Agenda v3.0.

[5] Mansmann, F., Fischer, F., Keim, D.A.: Dynamic visual analytics—facing the real-time challenge. In: Expanding the frontiers of visual analytics and visualization, pp. 69–80. Springer (2012).

[6] Databio EU Project. https://www.databio.eu. Online, accessed June 2020.

[7] Grafana. https://grafana.com. Online, accessed June 2020.

[8] HSLayers-NG site. https://ng.hslayers.org. Online, accessed June 2020.

[9] Zeppelin Apache Project. https://zeppelin.apache.org. Online, accessed June 2020.

[10] Engineering site. https://www.eng.it. Online, accessed June 2020.

[11] GNU Affero General Public License. https://www.gnu.org/licenses/agpl-3.0.en.html. Online, accessed June 2020.

[12] Hadoop Apache Project. https://hadoop.apache.org. Online, accessed June 2020.

[13] Python site. https://www.python.org. Online, accessed June 2020.

[14] R Project site. https://www.r-project.org. Online, accessed June 2020.

[15] Hive Apache Project. https://hive.apache.org. Online, accessed June 2020.

[16] Impala Apache Project. https://impala.apache.org. Online, accessed June 2020.

[17] HBase Apache Project. https://hbase.apache.org. Online, accessed June 2020.

[18] MongoDB site. https://www.mongodb.com. Online, accessed June 2020.

[19] OrientDB site. https://orientdb.com. Online, accessed June 2020.

[20] Cassandra Apache Project. https://cassandra.apache.org. Online, accessed June 2020.

[21] Random Forest https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#overview Online, accessed June 2020.

[22] Pandas – Python Data Analysis Library site. https://pandas.pydata.org. Online, accessed June 2020.

[23] Query by Example definition in Wikipedia site. https://en.wikipedia.org/wiki/Query_by_Example. Online, accessed June 2020.

[24] CKAN Data Management System Documentation site. https://docs.ckan.org/en/ckan-1.7.4/domain-model-dataset.html. Online, accessed June 2020.

[25] Groovy site. https://groovy-lang.org. Online, accessed June 2020.

[26] Standard ECMA-262 - Ecma International site. https://www.ecma-international.org/publications/standards/Ecma-262.htm. Online, accessed June 2020.

[27] Knowage JavaScript API site. https://knowage.readthedocs.io/en/latest/user/JS/README/index.html. Online, accessed June 2020.

[28] JSON with Padding definition in Wikipedia site. https://en.wikipedia.org/wiki/JSONP. Online, accessed June 2020.

[29] Cross-origin definition in Wikipedia site. https://en.wikipedia.org/wiki/Cross-origin_resource_sharing. Online, accessed June 2020.

[30] Docker site. https://www.docker.com. Online, accessed June 2020.

[31] Knowage Download site. https://www.knowage-suite.com/site/knowage-download. Online, accessed June 2020.

[32] Knowage Enterprise site. https://www.knowage-suite.com/site/licensing/enterprise-edition. Online, accessed June 2020.

[33] FIWARE – IDM (Identity Manager – KeyRock) site. https://fiware-idm.readthedocs.io/en/latest. Online, accessed June 2020.

[34] EIP-AGRI Concept EU site. https://ec.europa.eu/eip/agriculture/en/eip-agri-concept. Online, accessed June 2020.

[35] EU Survey site. https://ec.europa.eu/eusurvey. Online, accessed June 2020.

[36] Liferay site. https://www.liferay.com. Online, accessed June 2020.

[37] Openness Engineering site. http://openness.eng.it. Online, accessed June 2020.

Annex A SOCS Reference technology

The integrated and interconnected solutions of the SOCS aim to support efficiency, visibility, and collaboration processes. Indeed, the SOCS enables and supports the development, integration and delivery of knowledge sharing and collaboration services based on social networking, collective intelligence, collaboration, sharing, transparency, and self- empowerment. The SOCS is grounded on two main Open Source pillars: Liferay and OPENNESS.

Liferay ([36]) is a widely used Open Source and state-of-the-art Content Management System. Liferay Portal is a free enterprise portal software product written in Java. Liferay includes a built-in web content management system allowing users to build websites and portals as an assembly of themes, pages, portlets/gadgets, and a common navigation.

The second background component exploited is **OPENNESS** ([37]), an asset developed and managed by the ENGINGEERING R&D Department. This Open Source platform can support the collaboration among different stakeholders, such as farmers, advisors, farmer's association, and developers.

OPENNESS (OPEN Networked Enterprise Social Software) represents a platform fully developed using open source technologies, and leveraging principles related to Open Innovation, Collective Intelligence, Enterprise Social Software. The platform enables the development, integration and delivery of services and collaborative environments based on models of social networking and collective intelligence and provides in SaaS mode (Software-as-a-Service) features and services such as management and development of the community, knowledge management and collective experience and applications, enabling collaborative working and social analytics.

Moreover, the OPENNESS platform includes several search and retrieval features to help a user quickly find information, or just to browse through content, people, and teams.

Universal Search manages interactive search by inserting filters on the search bar. The users can search for specific words by simply entering the search terms to see content containing all the specified words in any order. They can find resources by person, title and content, and benefit from suggestions shown in real-time in the search box when typing the query. Furthermore, it also shows a set of tips to enrich the query that consider the user's activities in the community.

All the services that will be introduced in the following paragraphs have been implemented by collaborative business processes supported by OPENNESS.

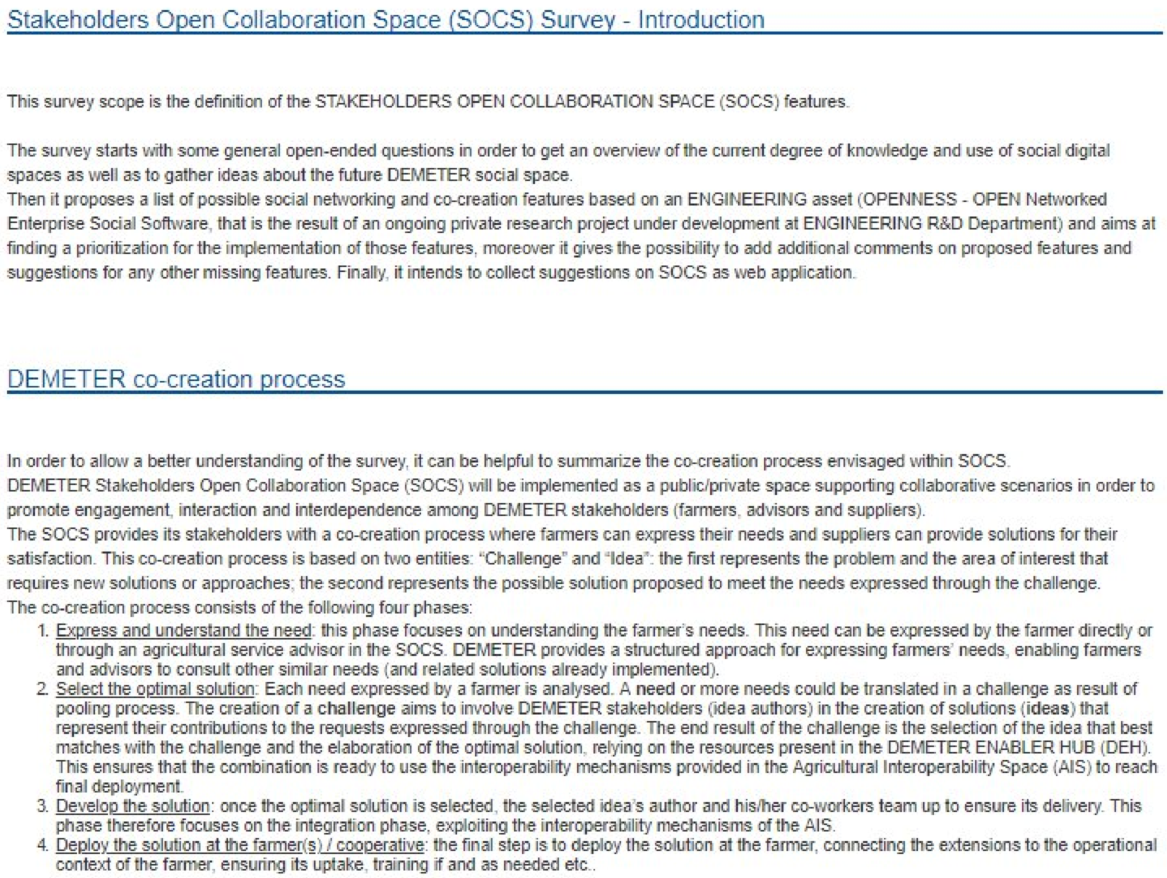
Annex B Survey SOCS

This annex provides the SOCS survey structure as screenshots from the online survey-management system “EU Survey”. The survey starts with an introductory section explaining the survey’s scope and then describing the co-creation process envisaged within SOCS. It includes some general open-ended questions to get an overview of the current degree of knowledge and use of social digital spaces as well as to gather ideas about the future DEMETER social space.

It propose a list of possible social networking and co-creation features based on the ENGINEERING asset and aims at finding a prioritization for the implementation of those features, moreover it gives the possibility to add additional comments on proposed features and suggestions for any other missing features. It intends to collect suggestions on SOCS as web application.

The survey was structured in the following six sections:

1. Introduction and explanatory section



Stakeholders Open Collaboration Space (SOCS) Survey - Introduction

This survey scope is the definition of the STAKEHOLDERS OPEN COLLABORATION SPACE (SOCS) features.

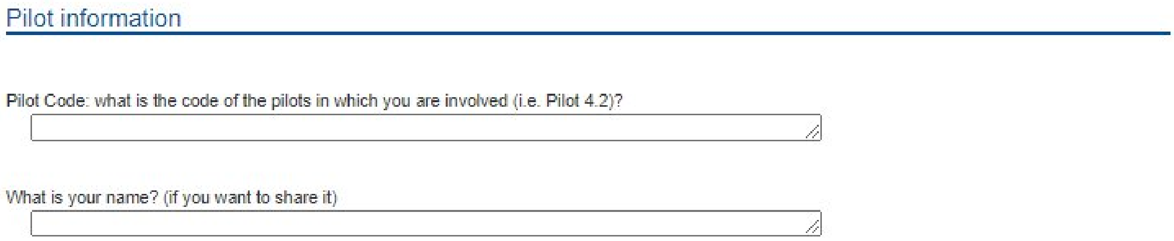
The survey starts with some general open-ended questions in order to get an overview of the current degree of knowledge and use of social digital spaces as well as to gather ideas about the future DEMETER social space.
Then it proposes a list of possible social networking and co-creation features based on an ENGINEERING asset (OPENNESS - OPEN Networked Enterprise Social Software, that is the result of an ongoing private research project under development at ENGINEERING R&D Department) and aims at finding a prioritization for the implementation of those features, moreover it gives the possibility to add additional comments on proposed features and suggestions for any other missing features. Finally, it intends to collect suggestions on SOCS as web application.

DEMETER co-creation process

In order to allow a better understanding of the survey, it can be helpful to summarize the co-creation process envisaged within SOCS.
DEMETER Stakeholders Open Collaboration Space (SOCS) will be implemented as a public/private space supporting collaborative scenarios in order to promote engagement, interaction and interdependence among DEMETER stakeholders (farmers, advisors and suppliers).
The SOCS provides its stakeholders with a co-creation process where farmers can express their needs and suppliers can provide solutions for their satisfaction. This co-creation process is based on two entities: “Challenge” and “Idea”: the first represents the problem and the area of interest that requires new solutions or approaches; the second represents the possible solution proposed to meet the needs expressed through the challenge.
The co-creation process consists of the following four phases:

1. Express and understand the need: this phase focuses on understanding the farmer’s needs. This need can be expressed by the farmer directly or through an agricultural service advisor in the SOCS. DEMETER provides a structured approach for expressing farmers’ needs, enabling farmers and advisors to consult other similar needs (and related solutions already implemented).
2. Select the optimal solution: Each need expressed by a farmer is analysed. A need or more needs could be translated in a challenge as result of pooling process. The creation of a challenge aims to involve DEMETER stakeholders (idea authors) in the creation of solutions (ideas) that represent their contributions to the requests expressed through the challenge. The end result of the challenge is the selection of the idea that best matches with the challenge and the elaboration of the optimal solution, relying on the resources present in the DEMETER ENABLER HUB (DEH). This ensures that the combination is ready to use the interoperability mechanisms provided in the Agricultural Interoperability Space (AIS) to reach final deployment.
3. Develop the solution: once the optimal solution is selected, the selected idea’s author and his/her co-workers team up to ensure its delivery. This phase therefore focuses on the integration phase, exploiting the interoperability mechanisms of the AIS.
4. Deploy the solution at the farmer(s) / cooperative: the final step is to deploy the solution at the farmer, connecting the extensions to the operational context of the farmer, ensuring its uptake, training if and as needed etc..

2. Respondent identification

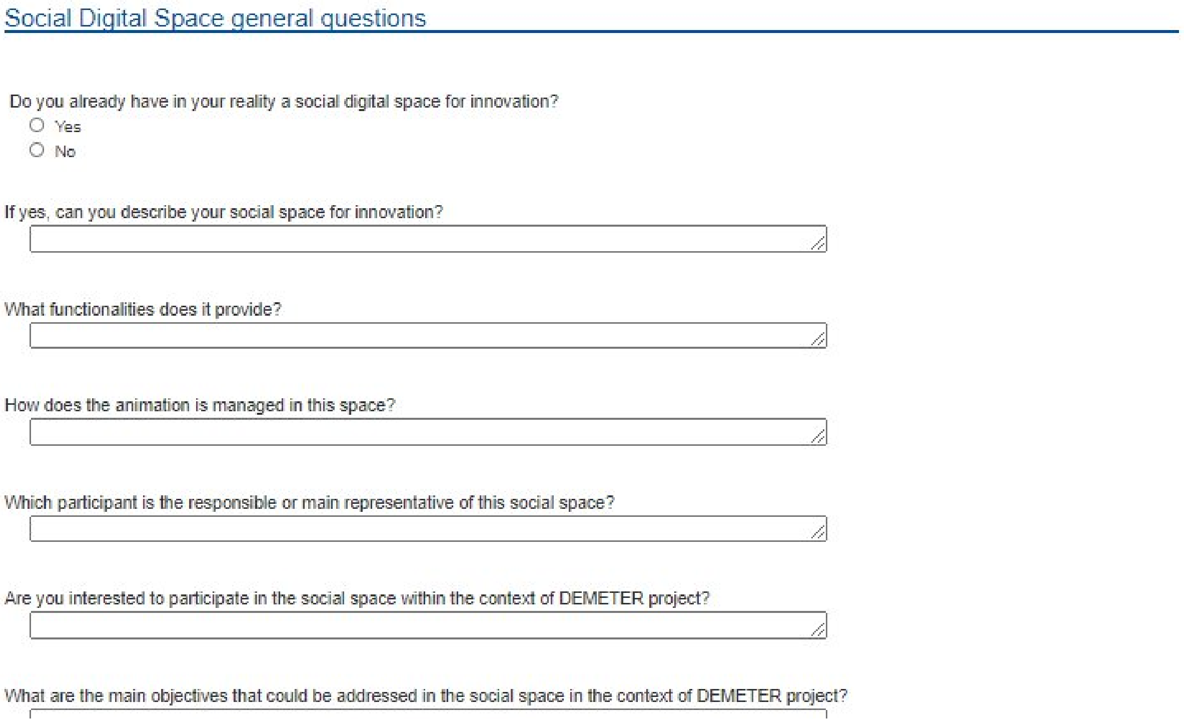


Pilot information

Pilot Code: what is the code of the pilots in which you are involved (i.e. Pilot 4.2)?

What is your name? (if you want to share it)

3. Social Digital Space general questions: open questions aiming at getting an overview of the current degree of knowledge and use of social digital spaces as well as to gather feedback about the willing to use the future DEMETER social space.



Social Digital Space general questions

Do you already have in your reality a social digital space for innovation?

○ Yes

○ No

If yes, can you describe your social space for innovation?

What functionalities does it provide?

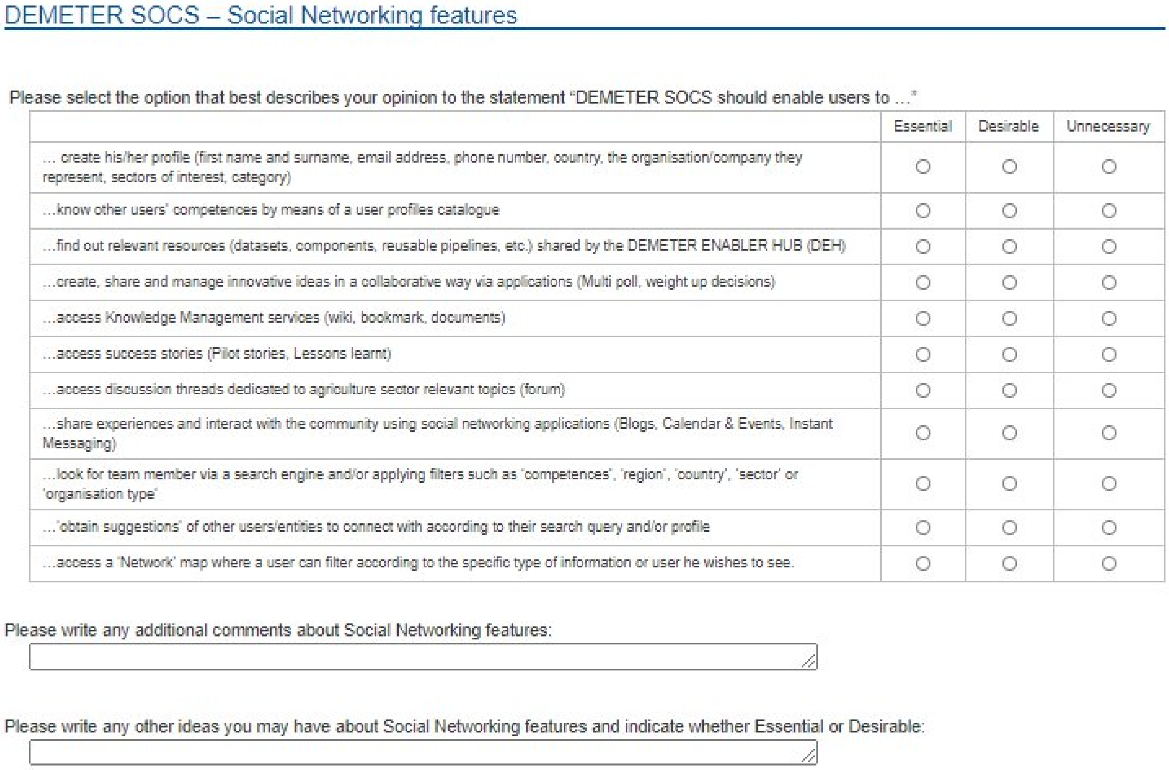
How does the animation is managed in this space?

Which participant is the responsible or main representative of this social space?

Are you interested to participate in the social space within the context of DEMETER project?

What are the main objectives that could be addressed in the social space in the context of DEMETER project?

4. DEMETER SOCS – Social Networking features: identification of which the social networking features should be, by selecting one of the options (“essential”, “desirable” or “unnecessary”) for each of the following propositions.



DEMETER SOCS – Social Networking features

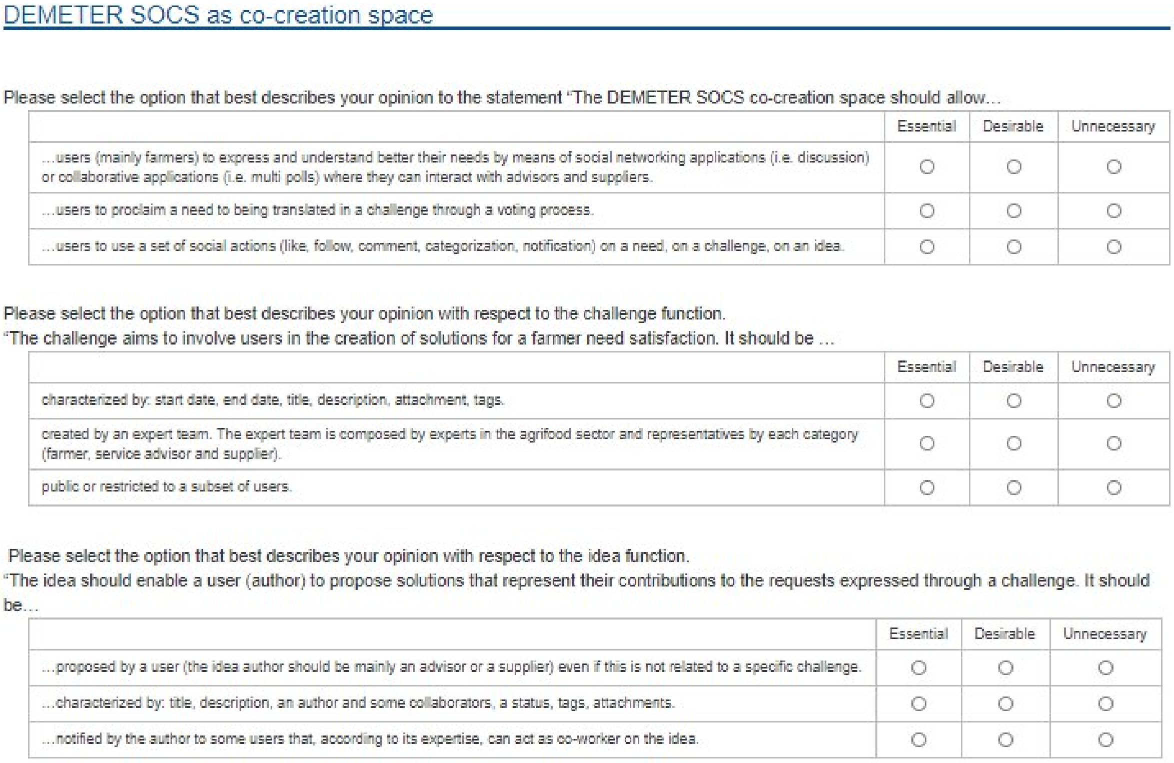
Please select the option that best describes your opinion to the statement "DEMETER SOCS should enable users to …"

	Essential	Desirable	Unnecessary
… create his/her profile (first name and surname, email address, phone number, country, the organisation/company they represent, sectors of interest, category)	○	○	○
…know other users' competences by means of a user profiles catalogue	○	○	○
…find out relevant resources (datasets, components, reusable pipelines, etc.) shared by the DEMETER ENABLER HUB (DEH)	○	○	○
…create, share and manage innovative ideas in a collaborative way via applications (Multi poll, weight up decisions)	○	○	○
…access Knowledge Management services (wiki, bookmark, documents)	○	○	○
…access success stories (Pilot stories, Lessons learnt)	○	○	○
…access discussion threads dedicated to agriculture sector relevant topics (forum)	○	○	○
…share experiences and interact with the community using social networking applications (Blogs, Calendar & Events, Instant Messaging)	○	○	○
…look for team member via a search engine and/or applying filters such as 'competences', 'region', 'country', 'sector' or 'organisation type'	○	○	○
…'obtain suggestions' of other users/entities to connect with according to their search query and/or profile	○	○	○
…access a 'Network' map where a user can filter according to the specific type of information or user he wishes to see.	○	○	○

Please write any additional comments about Social Networking features:

Please write any other ideas you may have about Social Networking features and indicate whether Essential or Desirable:

5. DEMETER SOCS as co-creation space: identification of which the co-creation space characteristics should be, by selecting one of the options ("essential", "desirable" or "unnecessary") for each of the following propositions related to the co-creation space, the challenge function, the idea function and the idea evaluation process.



DEMETER SOCS as co-creation space

Please select the option that best describes your opinion to the statement "The DEMETER SOCS co-creation space should allow…

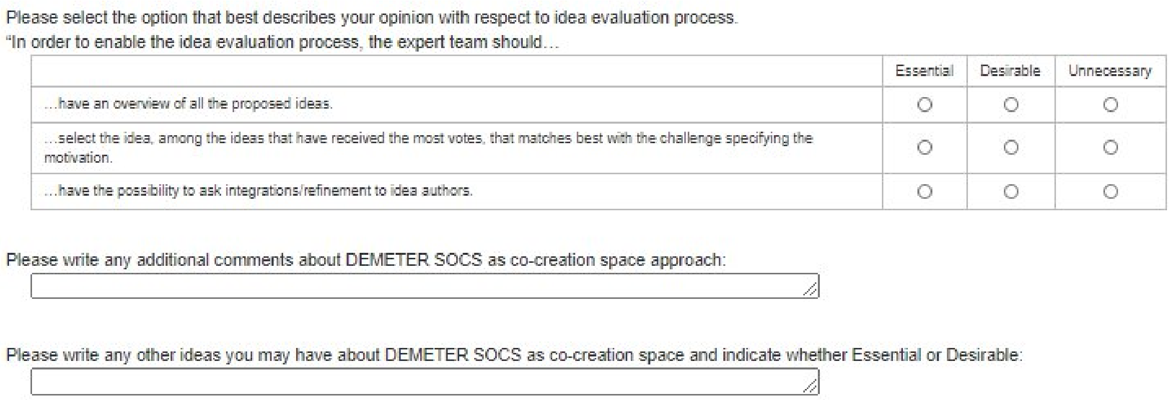
	Essential	Desirable	Unnecessary
…users (mainly farmers) to express and understand better their needs by means of social networking applications (i.e. discussion) or collaborative applications (i.e. multi polls) where they can interact with advisors and suppliers.	○	○	○
…users to proclaim a need to being translated in a challenge through a voting process.	○	○	○
…users to use a set of social actions (like, follow, comment, categorization, notification) on a need, on a challenge, on an idea.	○	○	○

Please select the option that best describes your opinion with respect to the challenge function.
"The challenge aims to involve users in the creation of solutions for a farmer need satisfaction. It should be …

	Essential	Desirable	Unnecessary
characterized by: start date, end date, title, description, attachment, tags.	○	○	○
created by an expert team. The expert team is composed by experts in the agrifood sector and representatives by each category (farmer, service advisor and supplier).	○	○	○
public or restricted to a subset of users.	○	○	○

Please select the option that best describes your opinion with respect to the idea function.
"The idea should enable a user (author) to propose solutions that represent their contributions to the requests expressed through a challenge. It should be…

	Essential	Desirable	Unnecessary
…proposed by a user (the idea author should be mainly an advisor or a supplier) even if this is not related to a specific challenge.	○	○	○
…characterized by: title, description, an author and some collaborators, a status, tags, attachments.	○	○	○
…notified by the author to some users that, according to its expertise, can act as co-worker on the idea.	○	○	○



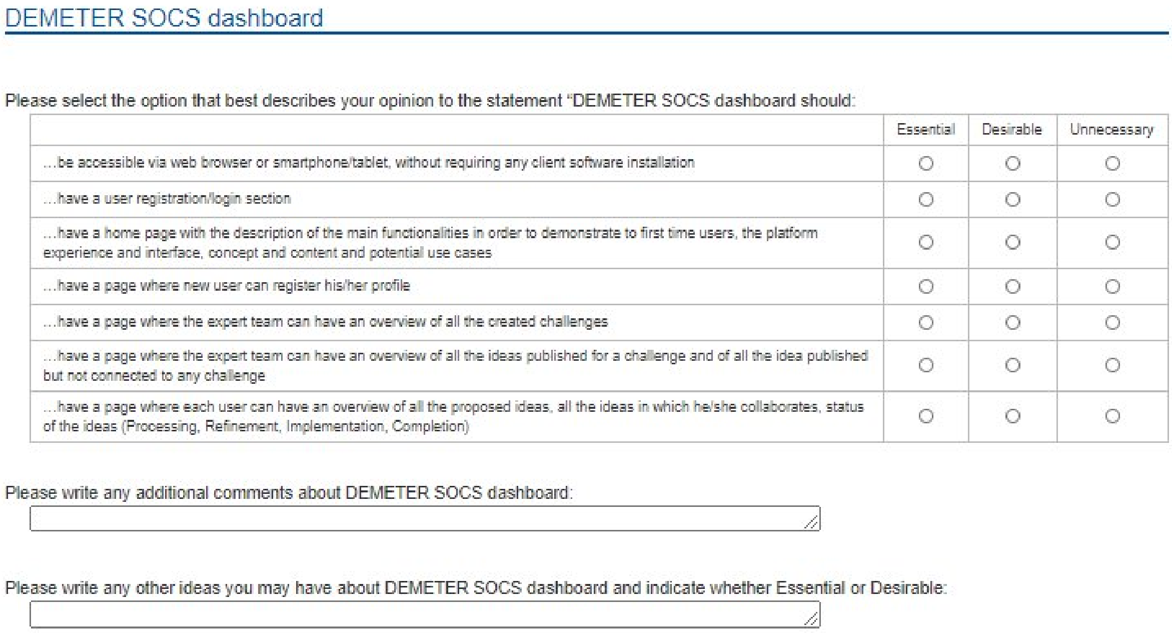
Please select the option that best describes your opinion with respect to idea evaluation process.
"In order to enable the idea evaluation process, the expert team should…

	Essential	Desirable	Unnecessary
…have an overview of all the proposed ideas.	○	○	○
…select the idea, among the ideas that have received the most votes, that matches best with the challenge specifying the motivation.	○	○	○
…have the possibility to ask integrations/refinement to idea authors.	○	○	○

Please write any additional comments about DEMETER SOCS as co-creation space approach:

Please write any other ideas you may have about DEMETER SOCS as co-creation space and indicate whether Essential or Desirable:

6. DEMETER SOCS DASHBOARD: identification of which the dashboard features should be, by selecting one of the options ("essential", "desirable" or "unnecessary") for each of the following propositions.



DEMETER SOCS dashboard

Please select the option that best describes your opinion to the statement "DEMETER SOCS dashboard should:

	Essential	Desirable	Unnecessary
…be accessible via web browser or smartphone/tablet, without requiring any client software installation	○	○	○
…have a user registration/login section	○	○	○
…have a home page with the description of the main functionalities in order to demonstrate to first time users, the platform experience and interface, concept and content and potential use cases	○	○	○
…have a page where new user can register his/her profile	○	○	○
…have a page where the expert team can have an overview of all the created challenges	○	○	○
…have a page where the expert team can have an overview of all the ideas published for a challenge and of all the idea published but not connected to any challenge	○	○	○
…have a page where each user can have an overview of all the proposed ideas, all the ideas in which he/she collaborates, status of the ideas (Processing, Refinement, Implementation, Completion)	○	○	○

Please write any additional comments about DEMETER SOCS dashboard:

Please write any other ideas you may have about DEMETER SOCS dashboard and indicate whether Essential or Desirable: