D6.1 AGRICORE Architecture and Interfaces

agricore

Deliverable Number	D6.1
Lead Beneficiary	IDE
Authors	Pablo Báez-González (IDE), Carlos Leyva (IDE), Alejandro Trujillo (IDE), Juan Carlos Castillo (AAT), Massimo Gioia (AAT), Mercedes Pichardo (AAT)
Work package	WP6
Delivery Date	M23* (M18)
Dissemination Level	Public

www.agricore-project.eu

The Agricore project has received funding from the European Union's Horizon 2020 research and innovation programme under the Grant Agreement No. 816078

Document Information

Project title	Agent-based support tool for the development of agriculture policies
Project acronym	AGRICORE
Project call	H2020-RUR-04-2018-2019
Grant number	816078
Project duration	1.09.2019-31.8.2023 (48 months)

Version History

Version	Description	Organisation	Date
0.1	Deliverable Template	IDE	02/02/2021
0.2	Re-structuration of architecture	IDE	17/05/2021
0.4	GPU-HPC analysis incorporated	IDE	28/05/2021
0.5	Definition of Modules included	IDE-AAT	03/06/2021
0.7	Definition of Interfaces included	IDE	05/07/2021
0.9	Inclusion of EPICS as Annex	IDE	15/07/2021
1.0	Revision, finalisation and exportation	IDE	30/07/2021

Executive Summary

This deliverable reports the advances done in the definition of the AGRICORE suite architecture and its interfaces. It provides the most updated information regarding how the whole suite is built and how each of its elements communicates with the others, this is, the protocols and communication schemes used. According to this, D6.1 Agricore Architecture and Interfaces is defined as a live document, which will be updated in parallel with the development work so as to always reflect the current information regarding the platform design.

Abbreviations

Abbreviation	Full name
ABM	Agent-based Model
ABM-e	Agent-based Model simulation Engine
API	Application Programming Interface
ARDIT	Agricultural Research Data Index Tool
CSS	Cascading Style Sheets
DAPR	Distributed Application Runtime
DEM	Data Extraction Module
DFM	Data Fusion Module
GHG	Greenhouse effect Gases
gRPC	Google Remote Procedure Calls
HDFS	Hadoop Distributed File System
HTTP	HyperText Transfer Protocol
HTML	HyperText Markup Language
IAMs	Impact Assessment Modules
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LDAP	Lightweight Directory Access Protocol
mTLS	Mutual TLS authentication
REST	Representational state transfer
SPG	Synthetic Population Generator
SQL	Structured Query Language
YARN	Yet Another Resource Negotiator

List of Figures

Figure 1 Initial Modular Architecture designed for the AGRICORE Proposal 12
Figure 2 AGRICORE Workflow for Policy Evaluation 13
Figure 3 AGRICORE Updated Modular Architecture 14
Figure 4 AGRICORE decomposition in three individual sub-architectures 17
Figure 5 Joint architecture of ARDIT and the DWH 18
Figure 6 Comparison of Dapr with service meshes 19
Figure 7 Subset of AGRICORE modules that are communicated through DAPR 20
Figure 8 Conceptual deployment diagram for container-based modules 20
Figure 9 Internal containerisation diagram of each module at microservice-level (ABM Module has been selected as example) 21
Figure 10 Modular approach used in CAPRI 22
Figure 11 Module communication traceability matrix 37
Figure 12 Memetic EMAS algorithm with classic (left) and hybrid (right) architecture. 49
Figure 13 Simulation time using FLAME for bacterial colonies. 50
Figure 14 Runtime of CPU backend by number of cores (left) and comparison of CPU/GPU hardware (right). 51
Figure 15 Simulation times for fixed (left) and variable (right) agents number. 52
Figure 16 AG.D1.FR.001. Provide a publicly accessible index of agriculture data sources 59
Figure 17 AG.D1.FR.002. Available for all stakeholders 59
Figure 18 AG.D1.FR.003. Store relevant information of the data sources 60
Figure 19 AG.D1.FR.004. Researchers will be able to extend its scope with additional data sources 60
Figure 20 AG.D1.FR.005. Semantic search will be allowed 61
Figure 21 AG.D2.FR.001. Centralise the information exchange within the AGRICORE IT architecture 61
Figure 22 AG.D2.FR.002. Use a combination of SQL and non-SQL databases massive parallel programming (MMP) technology and/or Hadoop/Spark 61
Figure 23 AG.D2.FR.003. Provide high-performance analysis capabilities to the DWH 62
Figure 24 AG.D2.FR.004. Easy-to-manage access permissions 62
Figure 25 AG.D2.FR.005. Separate critical/private information from information suited to be made public 62
Figure 26 AG.D2.FR.006. Support both private and public cloud infrastructure deployment 63
Figure 27 AG.D3.FR.001. Data of interest extraction 63
Figure 28 AG.D3.FR.002. Distribution curves generation 63
Figure 29 AG.D3.FR.003. Optimised data extraction operations 64
Figure 30 AG.D3.FR.004. Data output stored in DHW 64
Figure 31 AG.D4.FR.001. Infer the underlying joint probability distribution by means of statistical inference methods 64
Figure 32 AG.D4.FR.004. Data output stored in DHW 65
Figure 33 AG.D5.FR.001. Synthetic Reconstruction method 65
Figure 34 AG.D5.FR.002. Match the distribution of the agents' population of interest taking account of the joint probability distributions 65
Figure 35 AG.D5.FR.003. Receives input from the Data fusion module 66
Figure 36 AG.D5.FR.004. Data output stored in the DHW 66
Figure 37 AG.D6.FR.001. Simulate the evolution of the ABM population 66
Figure 38 AG.D6.FR.002. Manage the interactions required with the external modules 67
Figure 39 AG.D6.FR.003. Include the farm ABM model 67
Figure 40 AG.D6.NFR.001. Fully object-oriented implementation 67
Figure 41 AG.D6.NFR.002. Allow a set of high performance computing features 68
Figure 42 AG.D7.FR.001. Land Module Market 68
Figure 43 AG.D8.FR.001. Product Markets module 69
Figure 44 AG.D9.FR.001. Production Factors Market Module 69
Figure 45 AG.D10.FR.001. Compute the main KPIs related to the environmental and climatic impact assessment for a given set of ABM results 69

Figure 46 AG.D11.FR.001. Compute the main KPIs related to the socio-economic impact assessment for a given set of ABM results. 70
Figure 47 AG.D12.FR.001. Compute the main KPIs related to the Ecosystem Services impact assessment for a given set of ABM results. 70
Figure 48 AG.D13.FR.001. Provide access to climate models to obtain weather and climatic data 71
Figure 49 AG.D14.FR.001. Enable simulation of a wide variety of agricultural policy instruments 71
Figure 50 AG.D15.FR.001. Provide access to biophysical models to the AGRICORE tool 72
Figure 51 AG.D15.FR.002. Include plant, weather, stress, soil and agriculture management 72
Figure 52 AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite 73
Figure 53 AG.D16.FR.001-1. User Access Management 73
Figure 54 AG.D16.FR.001-1-2. User Registration 74
Figure 55 AG.D16.FR.001-2. Simulation Setup 74
Figure 56 AG.D16.FR.001-2. Simulation Setup 75
Figure 57 AG.D16.FR.001-2-1. Simulation Setup Page 75
Figure 58 AG.D16.FR.001-2-2. Synthetic Population Selection 76
Figure 59 AG.D16.FR.001-2-3. Selection of Policies to be included in the simulation 76
Figure 60 AG.D16.FR.001-2-4. Selection of Simulation Period 76
Figure 61 AG.D16.FR.001-2-5. Selection of Solver for the Simulation 77
Figure 62 AG.D16.FR.001-2-6. Selection of default KPIs to be computed 77
Figure 63 AG.D16.FR.001-3. Simulation Module 77
Figure 64 AG.D16.FR.001-4. Visualisation Module 78
Figure 65 AG.D16.FR.001-5. Visualisation Module 78
Figure 66 AG.D16.FR.001-5-1. Synthetic Population Generator Configurator 78
Figure 67 AG.D16.FR.001-5-2. Selection of Datasources for SPG 79
Figure 68 AG.D16.FR.001-5-3. Synthetic Population Generator (Down)Loader 79
Figure 69 AG.D16.FR.001-6. AGRICORE Suite Main Page 79
Figure 70 AG.D16.NFR.001. Developed as a cross-platform desktop application web technologies 80

List of Tables

Table 1 Runtime and strong scalability with 512, 1024 and 2048 nodes in K computer. 52

Table of Contents

1 **Introduction** 8

2 **AGRICORE Architecture** 9
2.1 Overall Goal 9
2.2 Initial Architecture 10
2.3 Updated Architecture 14
2.3.1 Global architecture principles 15
2.3.2 Modules grouping 16
2.3.3 Detailed architectures 17

3 **AGRICORE Modules** 22
3.1 Modular Approach 22
3.2 Updated List of Modules 24
3.3 Module Description 24
3.3.1 D1. Agricultural Research Data Index Tool (ARDIT) 24
3.3.2 D2. Data Warehouse (DWH) 24
3.3.3 D3. Data Extraction Module (DEM) 26
3.3.4 D4. Data Fusion Module (DFM) 27
3.3.5 D5. Synthetic Population Generator (SPG) 27
3.3.6 D6. Agent-based Model Simulation Engine (ABM-e) 28
3.3.7 D7, D8, D9 - Multi-Market Modules 31
3.3.8 D10, D11, D12 - Impact Assessment Modules 32
3.3.9 D13. Climate Model Connectors 33
3.3.10 D14. Policy Environment Module 33
3.3.11 D15. Biophysical Model Connectors 34
3.3.12 D16. AGRICORE Interface Module 35

4 **Interfaces** 37
4.1 Introduction 37
4.2 Interface Matrix 37
4.3 Interface Descriptions 38
4.3.1 ARDIT (D1) 38
4.3.2 DWH (D2) ↔ DEM (D3) 38
4.3.3 DWH (D2) ↔ DFM (D4) 38
4.3.4 DWH (D2) ↔ SPG (D5) 39
4.3.5 DWH (D2) ↔ ABM-e (D6) 39
4.3.6 ABM-e (D6) 39
4.3.7 Land Market Module (D7) 40
4.3.8 Product Markets Module (D8) 41
4.3.9 Production Factors Markets Module (D9) 42
4.3.10 Environmental/Climate IAM (D10) 43
4.3.11 Socio-economic IAM (D11) 43
4.3.12 Ecosystem Services IAM (D12) 44
4.3.13 Climate Models Connectors (D13) 45
4.3.14 Policy Environment Module (D14) 45
4.3.15 Biophysical Models Connectors (D15) 46
4.3.16 AGRICORE Interface Module (D16) 46

5 **Development Monitoring** 47
5.1 Open Source Approach 47
5.2 Usage of GitLab 47

6 **Optimisation of the agent-based simulation approach** 49
6.1 Current applications of parallel (HPC and/or GPU-based) simulation of agent-based systems 49
6.2 Current suitability for the AGRICORE approach 53

7 **Conclusions** 55

8 **References** 56

Annex A: List of Requirements (Gitlab Epics) 59

1 Introduction

The overall objective of this report is to describe the advances done in the AGRICORE Suite architecture definition from the one provided at the proposal stage. For this purpose, *Section 1 - AGRICORE architecture* delves into the overall solution goal and needs of such solution and provides a summary of the originally defined architecture. Moreover, this section also provides a high-level description of the updated architecture for the whole solution. Then, *Section 2 - AGRICORE modules*, describes the modules the proposed architecture is composed of. After this, *Section 3 - Interfaces*, details how these modules communicate across them, providing the initial specifications regarding protocol, languages and procedures used for such communication. Next, *Section 4 - Development monitoring*, delves into the procedures and methodologies defined for keeping track of the suite and modules development, in order to ensure the quality of the produced elements as well as their integrability. Then, *Section 5 - Optimisation of the agent-based simulation approach* explores the potential adaptation of the simulation approach to GPU or HPC methodologies to speed up the simulation. Finally, some initial conclusions are provided.

It is important to remark that this deliverable represents the first version of the upgraded AGRICORE Suite. However, this document has been marked as a living document, which means that it might get updated within the project execution to reflect any design change (or further specification) done to the architecture.

2 AGRICORE Architecture

2.1 Overall Goal

The overall objective of the AGRICORE suite architecture is indeed to provide a software basis to allow the implementation of its elements, ensuring that the result reaches the expected functionality. This apparently simple goal is indeed a quite relevant challenge as the functionality of the proposed solution is quite advanced and thus poses significant technical needs and requirements. As already introduced in other project documents, the overall goal of the AGRICORE project - and therefore of the proposed AGRICORE suite - is to develop a new generation of ABM tool
taking advantage of the latest progress in computational science and ICT (including advances in big data, artificial intelligence algorithms, mathematical solvers and cloud computing services) as a means to overcome the challenges that are still hampering their capacity for improving new policies design and for performing the related socio-economic and environmental assessments at various geographic scales – from regional to global.

This overarching goal establishes different requirements for the overall solution at different levels. The proposed solution is quite particular as it is, at the same time, a compute-intensive and a data-intensive application, depending on the step of its usage workflow we are at each moment. This means that the proposed architecture should provide those features of compute-intensive applications (as parallelization, signalling and scheduling of asynchronous tasks, support long-execution jobs) and at the same time, provide all the features key for data-intensive applications (although also useful in compute-intensive ones) as scalability, high-performance and easy maintenance.

Regarding the high-performance computing needs, they are clearly justified by the conceptual approach followed in the project. Indeed, the overall suite has the ultimate objective of being capable of simulating the evolution of all farms (> 10 million) in Europe across a given period (e.g. 4 years). Each farm, at the same time, is an autonomous decision-making entity that takes into account its situation, its surroundings and the environment to decide its next actions. Moreover, all farms interact with common elements as the product or the land markets. On top of that, biophysical models are used to predict the growth and the impact of the agricultural management decisions of the farms. As a result, the number of computations to be made is quite high, especially when considering the cross-Europe scenario.

Another key need for the AGRICORE architecture is its modularity. Modular design was already defined in 1985 [1] as a technique of decomposing into modules a software development, with the goal of reducing its overall implementation cost by allowing modules to be designed and revised independently. Specific goals of the module decomposition are as follows.

1. Each module's structure should be simple enough to be understood fully.
2. It should be possible to change the implementation of one module without knowledge of the implementation of other modules and without affecting the behaviour of other modules.
3. The case of making a change in the design should bear a reasonable relationship to the likelihood of the change being needed. It should be possible to make likely changes without changing any module interfaces; less likely changes may involve interface changes, but only for modules that are small and not widely used. Only very unlikely changes should require changes in the interfaces of widely used modules.
4. It should be possible to make a major software change as a set of independent changes to individual modules, i.e. except for interface changes, programmers changing the individual

modules should not need to communicate. If the interfaces of the modules are not revised, it should be possible to run and test any combination of old and new module versions.

In the AGRICORE suite, the modular design is a must for several reasons. First, the modelling and evaluation of policy impact on the frame of the Common Agriculture Policy (CAP) is a complex task combining efforts from several researchers and teams including JRC centres and Universities across Europe. A key way to promote adoption and ensure the survival of the proposed solution is to allow researchers to use the elements they find most valuable and enable them to substitute each of the pieces with new developments that provide advanced features (e.g. implementing new policy types, considering further aspects as transport targeting Farm To Fork strategies, etc.). Indeed, as noted in [2] and originally identified in [3] identified the prevailing project-related funding mechanism as a core reason why models are rarely maintained and extended. Enabling the partial upgrade and replacement of elements within an overall solution is indeed a way to promote that, even in a project-related funding scheme, the overall surviving of the previous developments is maintained.

Moreover, the AGRICORE project promotes the idea of enabling the overall suite to be used in all types of scenarios, including single nodes, private, public and even hybrid clouds. Indeed, even the originally proposed architecture made use of the concept of modular microservices interconnection, with matches a distributed approach for the global AGRICORE suite architecture. This target to further promote collaboration across different teams but also ensuring that the simulation of scenarios can be done in reasonable times, even when the computing needs exceed the capabilities of single-node deployments.

In the next subsection, the original approach for implementing the AGRICORE project results will be explained, focusing on the proposed architecture approach. After this, an updated description of the AGRICORE results is provided, delving in more technical details about its implementation.

2.2 Initial Architecture

The AGRICORE tool was conceived from the outset to be a highly modular and customizable package, released as open-source so that public institutions and academic organizations can transparently update and improve the tool as new needs arise. The partners designed the IT architecture depicted in Figure 1 below, where the main modules and their interconnections were tentatively defined. To ensure modularity, the partners committed to improving this architecture within the project. The specification of the interfaces between the different modules and the design and selection of clear ontologies was another point considered fundamental in the proposal, together with the use of standardized communication protocols to ensure maximum compatibility.

The modules initially contemplated within the modular architecture of the AGRICORE suite were the following:

- **D.1 - European Data-source Index for Agriculture Policy Assessment,** conceived as a virtual catalogue that allows researchers, policymakers and other stakeholders to both include and index agricultural data not only at the level of theme or topic addressed but down to the level of the specific variables, their physical units and the units of analysis to which they refer.
- **D.2 - Data Warehouse (DWH)**, which is the set of hardware and software elements that allow managing (load, extract, transform) those data flows that require storage. This includes, but is not limited to: variables for SPG, intermediate datasets derived from the data extraction process, definitions of probability distributions, synthetic populations, simulation results, etc.

- **D.3 - Big Data Extraction Module (DEM):** The user building the synthetic population must select, for each of the Attributes of Interest contemplated in the agents' structure, an existing variable in an accessible dataset to be used to assign random values to each one of the agents' attributes. The DEM performs the necessary ETL operations to obtain these variables from the corresponding data sources and assembles them forming one or several aggregated datasets that are stored in the private part of the user's DWH.
- **D.4 - Big Data Fusion Module (DFM)**, which, from the aggregated datasets obtained by the DEM and stored in the DWH, detects the possible existence of correlations and dependencies between attributes/variables and, in any case, obtains the probability distributions (individual, joint or conditional) of each variable. The formulations of these probability functions are stored in the DWH for later use during the generation of random values in the attributes of the agents.
- **D.5 - Synthetic Population Generator (SPG)**, which builds synthetic population by creating as many agents as specified by the user. This creation is based on using the probability functions obtained by the DFM to assign values to the attributes of each agent. Once all the attributes of all the agents have been populated, it is checked that the global probability distributions (for the whole population) of the attributes of interest in the synthetic population and in the real population do not differ above a predetermined value. Otherwise, the attribute assignment is repeated for a number of agents iteratively until the required fit is achieved.
- **D.6 - ABM Simulation Module**, whose function is to allow the simulation of the evolution of the agents on the basis of the agroeconomic management actions they take. These actions, determined for each agent and each campaign, are the result of an optimization process that represents the Holding Manager's decision making. Sometimes, these decisions cannot be fully implemented because they depend on the competitive interaction of some agents with others through the external market modules. In any case, the effect of these actions and interactions is reflected in the evolution of the state of each agent, given by the dynamic equations of the model.
- **D.7 - External Interface Module**, that would serve as a central point to relate the set of external auxiliary modules (D.8, D.9 and D.10) with the Agent-based Simulation Module (D.6), but also as a means to facilitate the incorporation of additional external modules by other researchers.
- **D.8 - Model Interaction Modules** considered the interactions of the agents with the rest of the agricultural structures and the connection with biophysical models both to mimic the manager's agronomic knowledge and to compute the effect of his decisions on the productivity of each agricultural enterprise (crop or livestock) within the holding.
- **D.9 - Impact Assessment Modules (IAMs)** simulate what would be the overall effect of the decisions taken by the different agents throughout the simulation. For this purpose, the results of the simulation (sum of actions) are aggregated and passed as arguments to the external impact assessment modules/models, which compute the effect that these actions would have i) on the environment (gas emissions, pollution of aquifers, variation in biodiversity, etc.), ii) on the socio-economic situation of the rural environment (improvement of the profitability of farms, effect on the labour market, incorporation of young people and women into the agricultural sector, etc.) and iii) on the capacity of ecosystems to provide services to humans (support, regulation, cultural, tourism, etc.).
- **D.10 - Policy Environment Module** allows to incorporate in the simulation of the agents one or several policy measures whose effect is to be calculated. This is done by transposing each of the simulated policy instruments by means of structural or parametric modifications to the objective function governing the optimisation of one or more groups of agents.

- **D.11 - AGRICORE Interface Module** is the set of screens, controls and mechanisms for inputting and loading information. Together, they allow the user to interact with the rest of the software elements of the AGRICORE suite.

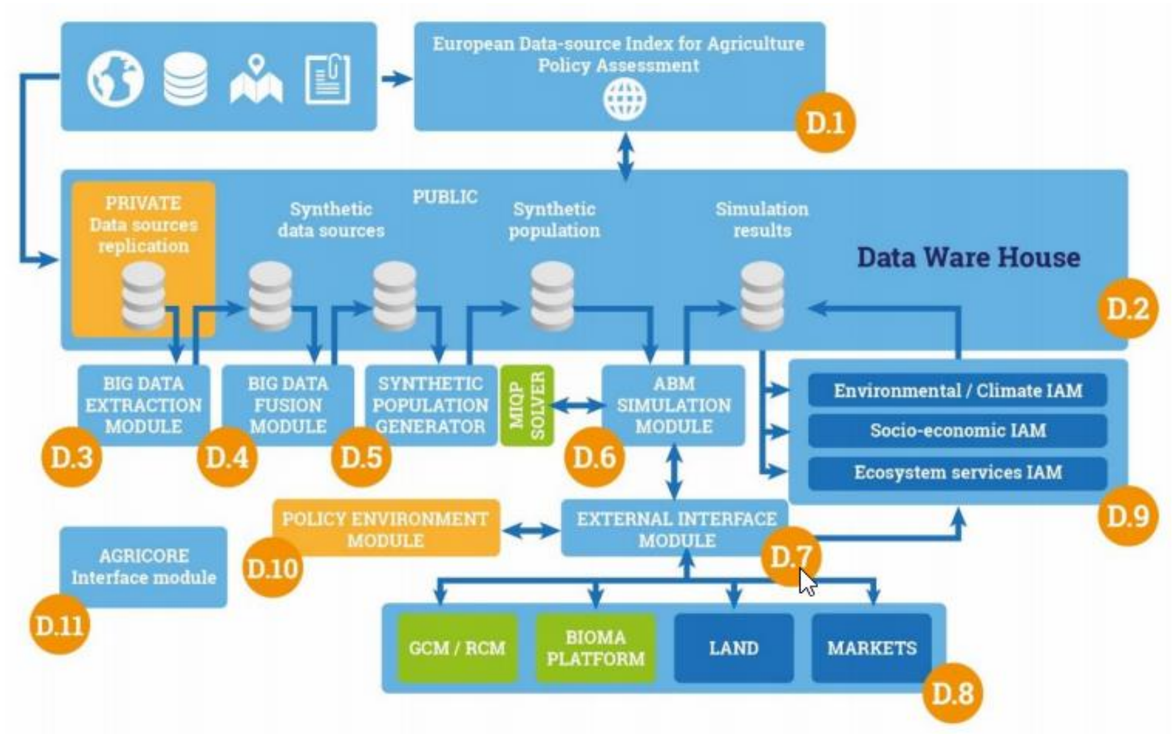

European Data-source Index for Agriculture Policy Assessment
D.1
PRIVATE Data sources replication
Synthetic data sources
PUBLIC
Synthetic population
Simulation results
Data Ware House
D.2
BIG DATA EXTRACTION MODULE
BIG DATA FUSION MODULE
SYNTHETIC POPULATION GENERATOR
MIQP SOLVER
ABM SIMULATION MODULE
Environmental / Climate IAM
Socio-economic IAM
Ecosystem services IAM
D.3
D.4
D.5
D.6
D.9
POLICY ENVIRONMENT MODULE
EXTERNAL INTERFACE MODULE
D.7
AGRICORE Interface module
D.10
D.11
GCM / RCM
BIOMA PLATFORM
LAND
MARKETS
D.8

Figure 1 Initial Modular Architecture designed for the AGRICORE Proposal

This architecture aimed to support the original workflow conceived for the application of the AGRICORE suite for policy impact assessment:

1. **Data gathering:** Depending on the policy instrument whose impact is to be assessed, it is necessary to detect and access data that enable the assignation of values to the set of attributes of interest that make up each of the agents. This information must be statistically representative of the set of real farm holdings to be synthesised, precisely those that are affected by the policy measure under study in terms of their typology (technical-economic orientation), size (economic dimension) and location (geographical and administrative scope). It is also necessary to obtain all information on the policy measure itself: requirements to be met by potential beneficiaries (natural or legal persons), requirements to be met by potential beneficiary farms, articulation of the measure (monetary amounts, the timing of payments, etc.) and restrictions of the policy programme associated with the measure (total budget, the maximum number of eligible holdings, etc.).
2. **Synthetic Population Generation:** Using all the information obtained, an advanced user can construct the synthetic population of agents that constitutes the simulation subject. While the DEM, DFM and SPG modules, with their respective graphical interfaces, can assist in the process, the construction of synthetic populations is, for the time being, a complex procedure that requires advanced statistical knowledge (Bayesian Networks) and programming skills.

3. **Connection with Biophysical Models and Solvers:** To enable the execution of the simulation, communication mechanisms must be in place to allow the agents to interact with the solver that computes their agro-economic decisions at each simulation step (each simulated agricultural season). There must also be an interconnection mechanism between the agents and the biophysical models necessary to simulate the evolution of the different enterprises.
4. **Simulation Set-up:** In addition to the above interconnections (solver and biophysical models), the user must configure the remaining simulation parameters through the AGRICORE graphical interface. These are the synthetic population to be used, the policy measure(s) to be tested, the base year of the simulation and its duration in agricultural campaigns, some KPIs to be computed in a predefined way at the end of the simulation.
5. **Simulation execution:** Once configured, the simulation is launched via the AGRICORE graphical interface, from which it is also possible to follow its progress. For internal execution, the GPU and HPC parallelisation techniques presented in the second to last section of this deliverable can be used.
6. **Calls to IAMs:** Those KPIs predefined during the simulation set-up are automatically calculated by the AGRICORE suite through calls to the corresponding IAMs. Once calculated, they are automatically displayed in the visualisation module. However, the user can always operate with the simulation results, conveniently tabulated, and the IAMs to obtain additional KPIs as needed.
7. **Results visualisation and analysis:** The different KPI evolution graphs can be visualised and/or downloaded using the visualisation module. From the analysis of these graphs, the evaluation of the policy measure under analysis can be extracted and, eventually, iteratively redesigned until the results predicted by the model meet the objectives that the policy-maker desires.

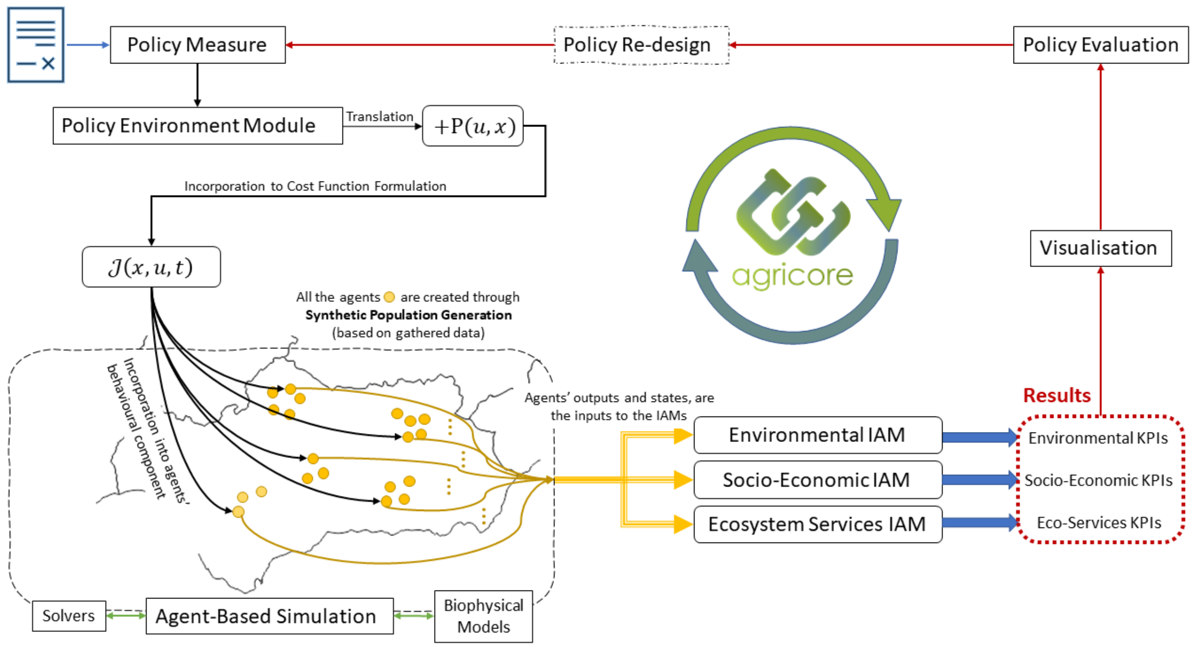


Figure 2 AGRICORE Workflow for Policy Evaluation

2.3 Updated Architecture

During the first stage of the AGRICORE project, the consortium members have been discussing and iterating across different definitions of all the suite modules, as well as of the workflows to be applied (both, by the user and internally during the population generation and the agent simulation processes). As a result of such analyses the architecture initially drafted has significantly changed and further details on how it should be implemented have been defined. In order to facilitate the follow up of the changes performed regarding the modules to be implemented, the next Figure represents the updated architecture in a similar way as it was presented in the project proposal.

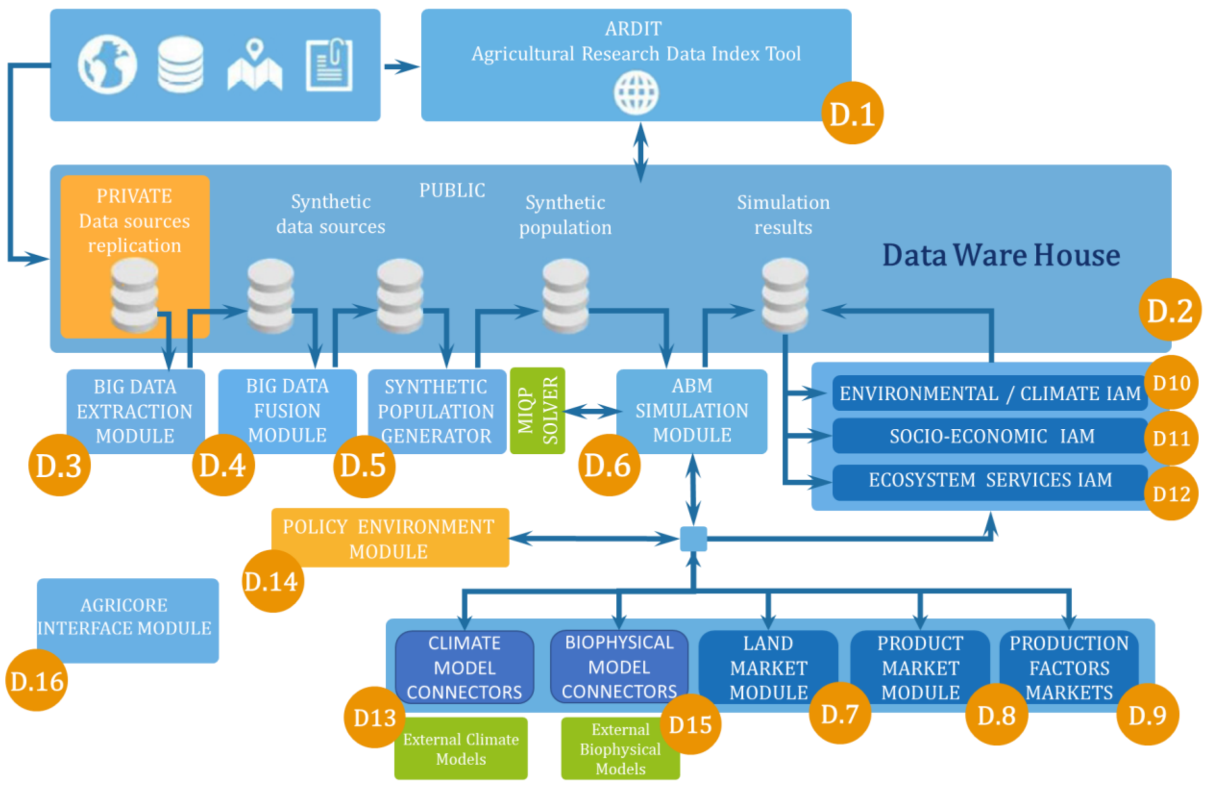


Figure 3 AGRICORE Updated Modular Architecture

The main differences between the two figures (and accordingly with this schematic architecture representation) are:

- The module previously defined as "European Data-Index for Agriculture Policy Assessment" has been renamed to "ARDIT - Agricultural Research Data Index Tool". The change does not affect only the name. Indeed, the expected functionality for this element has been significantly upgraded, including governance schemes for its future maintenance, the inclusion of ETLs (Extraction Transformation Load) scripts to the characterised datasets and the possibility to defined user-private sections.
- Due to the relevance of the developments, every Impact Assessment Module (IAM) is now an independent element.
- This also applies to the market modules, which are now split between "Land Market Module", "Product Market Module" and "Production Factors Market Module".

- The BIOMA connection module has been replaced by a generic Biophysical Model Connector due to the impossibility to get access to the latest version of BIOMA (source code) and being unable to establish contact with their developers. Currently, this module aims to enable the connection to different growth models as STICS, WOFOST, DNDC or ECropS.
- The model defined as External Interface Module, which aimed to standardized the way the modules communicate between them has disappeared. The explanation for this will be provided in the next paragraphs when describing in detail the proposed software architecture.

2.3.1 Global architecture principles

As defined at the beginning of this section, the AGRICORE suite expected functionality established several requirements for the definition of its architecture, especially at the level of Scalability, High-performance and Maintainability. Classical ABM-oriented solutions tend to focus on highly integrated (monolithic) approaches in where most of the implementation is done in the same place and language, being the common extension just a set of added functions or linked libraries. However, this approach is difficult to scale up as its implementations rarely include load distribution mechanisms that enable easily upgrading the processing power or distributing the processing needs across different nodes. Similarly, these solutions tend to be highly coupled, using the same languages for all implementations and providing less modularity than desired, as usually including a new functionality requires modifications not only in the modules affected.

Alternatively, the AGRICORE project proposed a distributed architecture, which clear separation between the elements composing it by following a truly modular approach. To do so, 6 mandatory principles have been established, not only for the overall system architecture but also for each one of the elements forming it: 1) Use SOA as basic architecture reference; 2) Use microservices and containerisation technologies; 3) Always provide single node deployment scripts; 4) Support both cloud and on-premise deployment; 5) Don't establish any language requirement to other modules; and 6) For performance demanding elements, provide ways to tackle performance limitations in SOA architectures. These principles should be considered when designing and implementing any combination or a single module. Moreover, these rules should always be considered in addition to regular Code Quality rules (e.g. Produce proper comments and documentation, Don't Repeat Yourself, etc.) and the rules defined in the project's Software Quality Assurance methodology. Each of these additional principles is analysed next:

2.3.1.1 Use SOA as basic architecture reference;

Service Oriente Architecture (SOA) is a style of designing a software solution supporting service orientation. A service can be considered as a discrete software unit that can be invoked or accessed by other elements in the same architecture, providing data to the invoked service and obtaining other information as a result of the requested operation (or a change of state in the underlying model). According to one of the definitions of SOA [4], a service has four main properties:

- It logically represents a repeatable business activity with a specified outcome.
- It is self-contained.
- It is a black box for its consumers, meaning the consumer does not have to be aware of the service's inner workings.
- It may be composed of other services.

If these four properties are compared against the list of specific goals of modular software design approaches, it can be seen that the mere adoption of a SOA strongly promotes the modularity of the solution to be developed. In AGRICORE, SOA will be used for defining each sub-architecture in order to ensure easy future distribution and full modularity of the AGRICORE suite.

2.3.1.2 Use microservices and containerisation technologies

Microservices can be considered an extension or a variant of the SOA. Indeed, microservices have been defined [5] as a modern interpretation of service-oriented architectures used to build distributed software systems. The concept of microservices is a natural evolution of the classical SOA but aiming for more fine-grained interfaces, smaller services and cloud orientation, to cope with the need for elasticity while upscaling applications. Indeed, microservices are strongly linked to DevOps and Continuous Delivery and Deployment [6]. These two software methodologies or concepts propose software creation procedures that combine software development with IT operations for a faster software life cycle. When combined with containerisation technologies (e.g. Docker) and proper development scenario setup, can strongly facilitate testing and developing activities by enabling both tasks to be run in an environment almost identical to the production one.

In AGRICORE, microservices, DevOps and containerisation technologies (in this case, docker), will be integrated across all the developments. This will reinforce the modularity of the solution, enable a distributed deployment and an easy scale-up of operations.

2.3.1.3 Always provide single node deployment scripts

Although most of the project's development will benefit from distributed scenarios - where several computing nodes are available - the AGRICORE should be able to be used in a single computer. This will enable an easier development process but will also benefit the adoption of the results, allowing researchers to not depend on a complex and powerful set of machines to use the application.

2.3.1.4 Enable hybrid deployments

On the contrary, all developments should also support fully distributed approaches, either using public cloud services or private ones. Moreover, a hybrid approach should also be supported, in which most of the application is deployed in a local private cloud but also public cloud services are occasionally used for computing-intensive tasks.

2.3.1.5 Don't establish any language requirement to other modules

Following SOA and microservices principles, the different (micro)services being developed should not be tied to any specific language implementation. This is especially relevant for this application as several researchers in the Agricultural Policy field are already used to specific languages and forcing them to adopt new languages or frameworks would strongly hinder the adoption of the project results.

2.3.1.6 For performance demanding elements, provide ways to tackle performance limitations in SOA architectures

SOA and microservices are great for distributed applications. However, they pose some limitations for highly demanding computing tasks (as for instance, the simulation of big populations in a distributed ABM approach). Accordingly, each one of the individual architectures defined should provide means for tackling such limitations.

2.3.2 Modules grouping

When analysing the new list of modules to be developed, a logical relation between some of them has been identified. The next Figure depicts the same services as already presented but split them into three independent architectures.

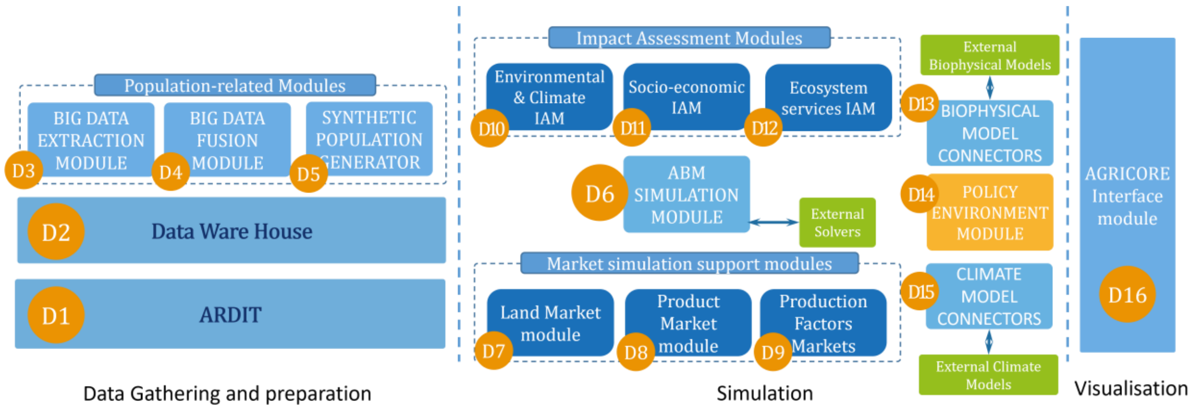

Population-related Modules
BIG DATA EXTRACTION MODULE
BIG DATA FUSION MODULE
SYNTHETIC POPULATION GENERATOR
D3
D4
D5
D2
Data Ware House
D1
ARDIT
Data Gathering and preparation
Impact Assessment Modules
Environmental & Climate IAM
Socio-economic IAM
Ecosystem services IAM
D10
D11
D12
D6
ABM SIMULATION MODULE
External Solvers
Market simulation support modules
Land Market module
Product Market module
Production Factors Markets
D7
D8
D9
Simulation
External Biophysical Models
D13
BIOPHYSICAL MODEL CONNECTORS
D14
POLICY ENVIRONMENT MODULE
D15
CLIMATE MODEL CONNECTORS
External Climate Models
AGRICORE Interface module
D16
Visualisation

Figure 4 AGRICORE decomposition in three individual sub-architectures

In these schemes, modules D1 to D5 have been grouped in a separate architecture which includes ARDIT, the DWH and the modules related to the generation of populations. Moreover, modules D6 to D15 have been also grouped in a Simulation architecture while D16 is in a separated one. However, these three architectures are still connected between them. Indeed, the simulation architecture will consume information from the Data Gathering and Preparation one by obtaining (and pouring) data from the DWH. Similarly, the interface module will gather information from the DWH but will also be used to configure the simulation services and to read its results.

2.3.3 Detailed architectures

Each one of these architectures has been further defined. These detailed architectures are described next.

2.3.3.1 Data Gathering and preparation architecture

Data gathering and preparation architecture includes mainly the connection between the ARDIT and the DWH tools for the storage of potentially useful data sources that feed the synthetic populations with data. This architecture has been designed to meet the main objectives of the AGRICORE project in terms of designing fully distributed solutions, which can be used in public cloud services or private ones, and in addition, adding the possibility to download and install the tools locally by researchers, on machines with less resources or processing capabilities. In this way, two approaches to the architecture between ARDIT and the DWH have been designed:

- Global architecture: includes the use of a publicly available version of ARDIT (Global Indexer) on the web. Users will be able to perform searches on data sources based on multiple attributes and parameters, new data source characterisations will be stored there and also, users will be allowed to store them in a DWH exploited in cloud. However, the access and use of this DWH will be restricted to authorised users only, as it will be use for project purposes.
- Local architecture: this approach allows users to download and install the complete architecture in private environments in a simple way, enabling researchers, developers and anyone interested in the project to use and test the tools on their machines. In this case, the ARDIT indexer will not allow new characterisations to be made and will be limited to simply loading into the DWH, data sources available on the user's machines. In addition, local indexers will be able to subscribe to the Global Indexer in order to update their collections of characterised data sources. In other words, if a new new data source is indexed in the Global Indexer, users will be able to synchronise their local databases with the global one. So, users will have that data source available to be loaded into their data warehouses.

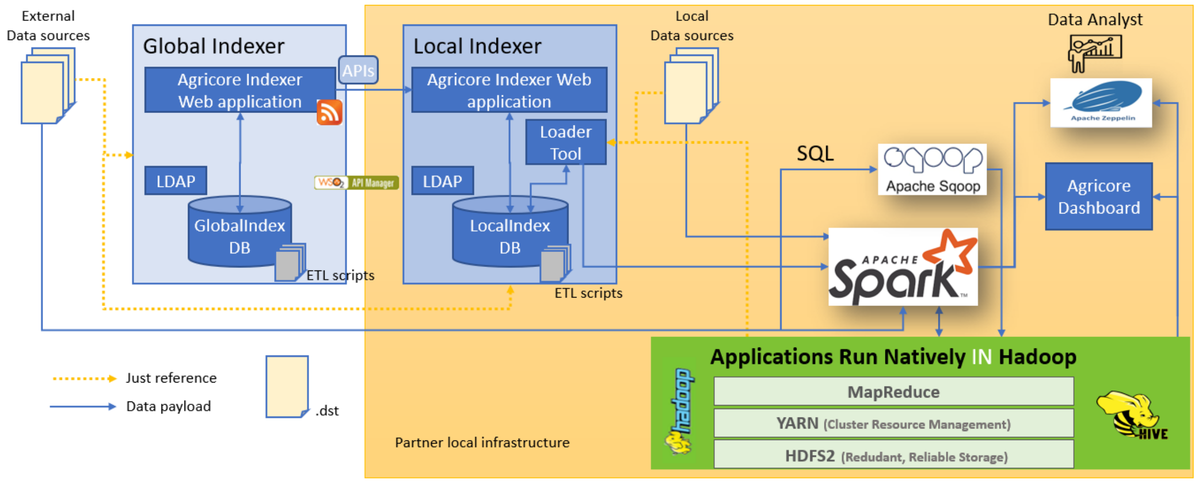


Figure 5 Joint architecture of ARDIT and the DWH

2.3.3.2 Simulation Architecture

The Simulation Architecture includes all the modules that are used during the simulation of the evolution of the population. Due to the agent-based nature of the solution and the complex decision-making process done by each farmer, this architecture can be safely considered as a high demand one in terms of computing power. Indeed, running the number of iterations required to simulate a considerable number of farms poses several constraints to the proposed SOA(microservice) distributed approach, as all the limitations of such basic architecture are in conflict with these modules needs. In order to tackle this issue. For instance, the distributed approach would imply a high number of calls between all the services involved which, even with a minor latency between services, would surely impact the performance. Accordingly, several measures need to be put in place to tackle this effect. Among them:

- Each service should always allow the parallel execution of itself. This means that, for instance, the product market module would need to be able to be called for a single product, for two of them, or for any set of products to be analysed. It is the responsibility of the service to then parallelised such requests in a proper way.
- The architecture should provide common ways for regular tasks, avoiding overlapping and duplication of functionalities. This includes state management, subscribing to events, chaining services and signalling between them.
- Every module should offer a well-defined interface (as it is common in any service-oriented scheme). However, in order to maximise performance, this interface should be offered using a high-performant protocol, as for instance gRPC.
- Services should be easily invokable, avoiding the user needing to deal with service discovery and name resolution.
- All the services should use a common way to log events or at least to enable the debug and tracing of the application execution to facilitate development and maintenance tasks.

According to these needs, the AGRICORE development team has analysed several possibilities. Initially, the approach proposed consisted of the inclusion of a service mesh. A service mesh is a dedicated layer for easing service-to-service communications. They usually facilitate deployment and tackle all network-related aspects of distributed deployments, like service discovery and name resolution. However, this initial approach was not enough as it would require the additional elements to provide all the related functionality, as signalling between services, event subscription, state management, etc. Accordingly, the AGRICORE team evaluated different possibilities which included the combination of a service mesh with additional solutions and as a result, DAPR was selected, which is an open-source project led by Microsoft. DAPR (Distributed Application Runtime) is a portable, event-driven runtime that makes it easy for any developer to build resilient, stateless and stateful applications that run on the cloud and edge and embraces the diversity of languages and developer frameworks [7]. The overall comparison between DAPR and a service mesh is depicted in the next figure:

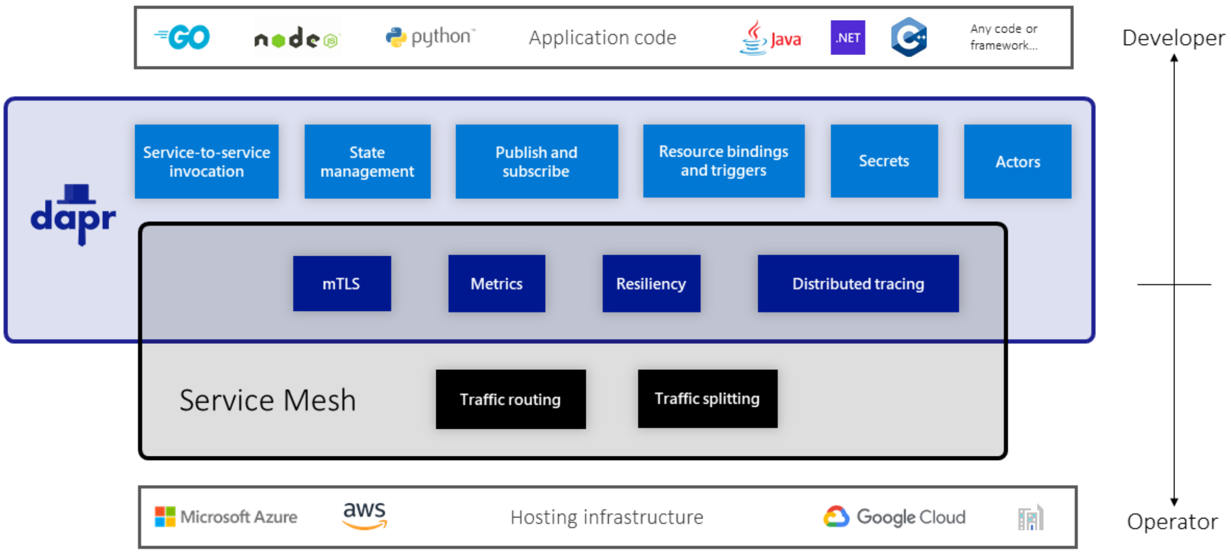


Figure 6 Comparison of Dapr with service meshes

As it can be seen, DAPR provides some unique features as Service-to-service invocation, state management, publish and subscribe, resource binding and triggering, secret management and actors definition. It also provides some overlapping functionality with service mesh as mTLS, Metrics, Resiliency and Distributed tracing. It doesn't provide traffic routing or splitting but yet, it support load balancing through mDNS and easy deployment in a docker environment and public/private hosting infrastructures.

The inclusion of DAPR enables providing almost all the required core functionalities to comply with the modular approach as well as with the microservice oriented and the distributed deployment approach. Accordingly, DAPR has been adopted as a core element of the Simulation architecture.

DAPR works by including a small sidecar container implementation, providing all the runtime functionality through them. This approach is quite adequate as enables a separate configuration of the DAPR solution for all the elements and enable each module developer to somehow ignore its existence and focus on its module implementation. It also facilitates the invocation (synchronous or asynchronous) of other services and the management of the application status in a centralised (but yet distributed) way. By adopting DAPR, the Simulation architecture is then proposed as described in the next figure:

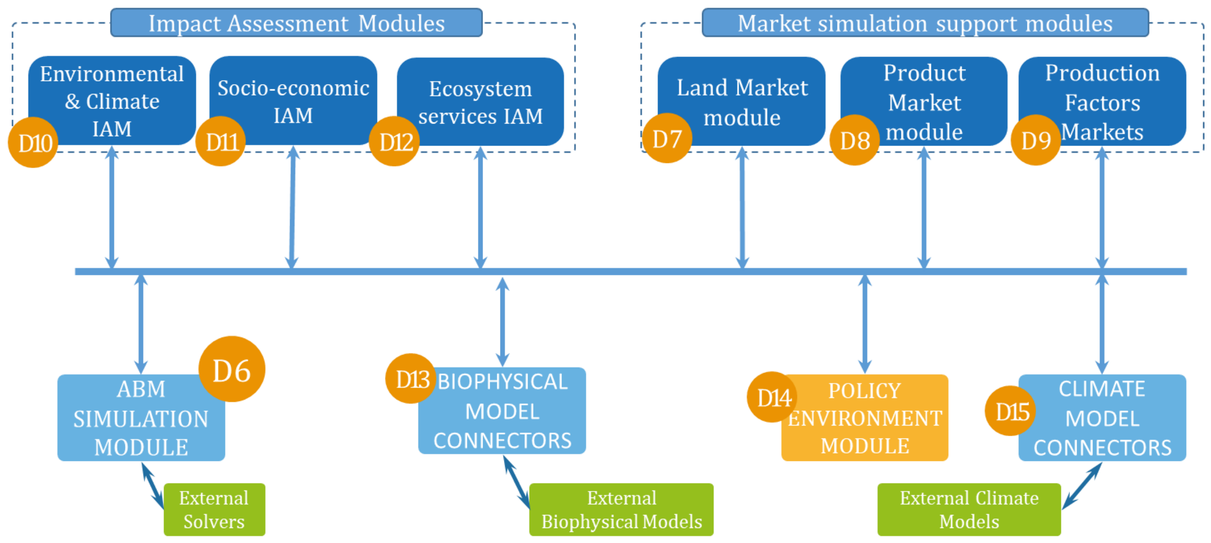

Impact Assessment Modules
Environmental & Climate IAM
D10
Socio-economic IAM
D11
Ecosystem services IAM
D12
Market simulation support modules
Land Market module
D7
Product Market module
D8
Production Factors Markets
D9
ABM SIMULATION MODULE
D6
D13
BIOPHYSICAL MODEL CONNECTORS
D14
POLICY ENVIRONMENT MODULE
D15
CLIMATE MODEL CONNECTORS
External Solvers
External Biophysical Models
External Climate Models

Figure 7 Subset of AGRICORE modules that are communicated through DAPR

As it can be seen, all services (modules) are interconnected (through DAPR)and can consume each other functionalities. In a global scheme, this architecture can be deployed as depicted in the next Figure.

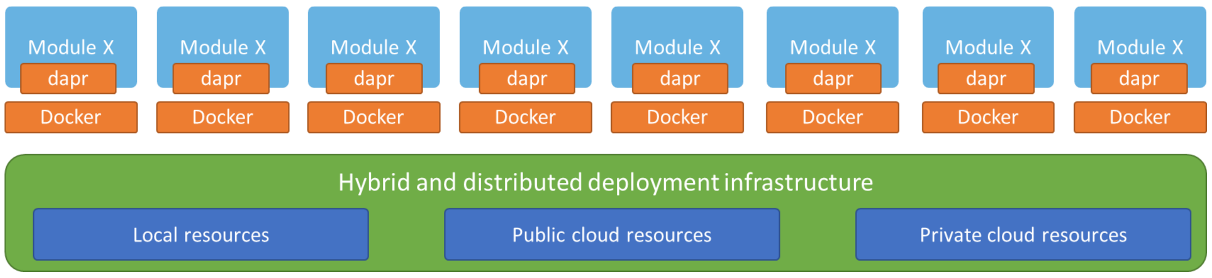


Figure 8 Conceptual deployment diagram for container-based modules

Moreover, each one of the modules implemented as a service included in this architecture has a similar implementation. Each module is composed of a custom service implementation which, as already mentioned, should expose a gRPC interface, which is connected to a sidecar container for DAPR integration. The custom service should also be encapsulated in a docker container. The custom service can be implemented in any language, although its implementation and integration is easier if it is done in any of the DAPR supported languages (C++, Go, Java, JavaScript, Python, Rust, .NET and PHP).

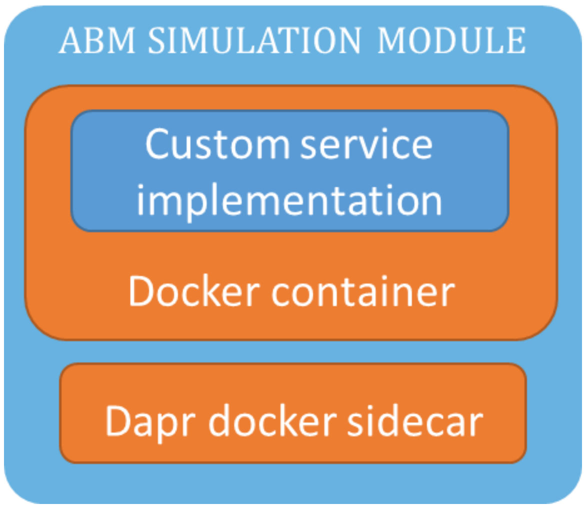


Figure 9 Internal containerisation diagram of each module at microservice-level (ABM Module has been selected as example)

3 AGRICORE Modules

3.1 Modular Approach

Modular software is a design approach, which is based on the separation of the software in different stand-alone modules, allowing them to be developed, tested and maintained independently. Each module is composed of different functionalities, common or specific, and different components. In this way, the modular development allows the component re-utilisation, reusable code, improves efficiency, it is easy to debug, modify and maintain and reduces costs [8][9][10].

To mention a few disadvantages, having a large number of small modules can affect performance due to high communication overhead, which might require additional memory and slow down execution time due to the delay of such communications. In addition, maintainability becomes more complex due to more interfaces, more documentation generated, etc.

CAPRI is a European Commission project for decision making related to the Common Agricultural Policy on the basis of quantitative scientific analysis. It uses modular software, consisting of interconnected modules that provide different types of data, on markets, policies, etc. It also has another exploitation module consisting of a graphical interface developed in JAVA, where graphs, maps or tables can be viewed.

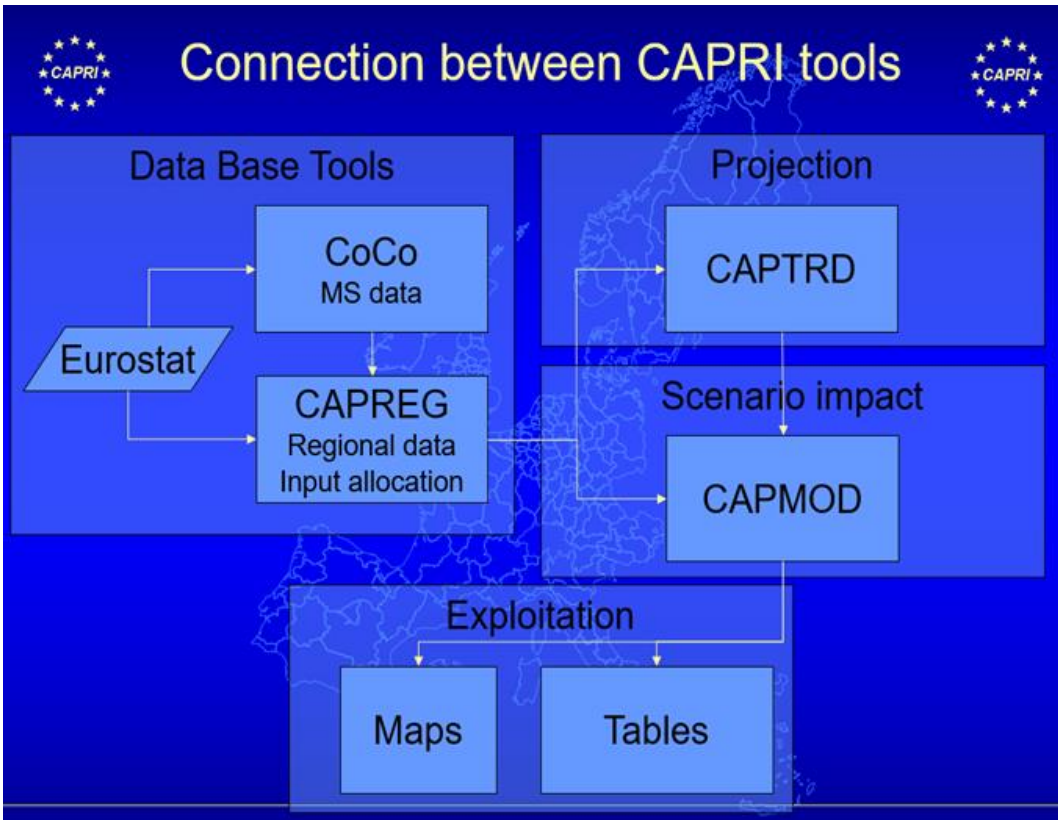
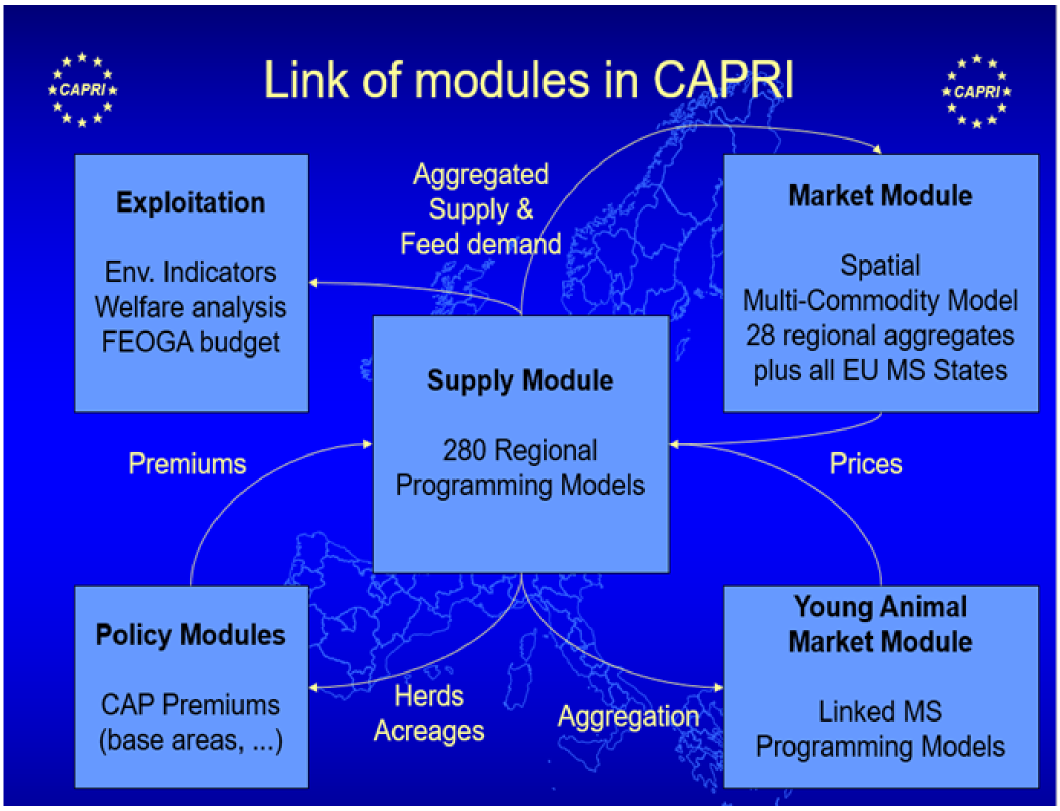


Figure 10 Modular approach used in CAPRI



Other EU agricultural policy projects using modular development include:

- **Fast:** the FaST digital service platform will make available capabilities for agriculture, environment and sustainability to EU farmers, Member State Paying Agencies, farm advisors and developers of digital solutions.
- **BESTMAP:** develop a new modelling framework using insights from behavioural theory, linking existing economic modelling with individual-farm Agent-Based Models.
- **Mind Step:** European research project aiming to improve the exploitation of available agricultural and biophysical data and will include the individual decision making (IDM) unit in policy models.
- **iMap:** a modelling platform that includes the main partial equilibrium models (AGLINK, CAPRI, ESIM, AGMEMOD) and general equilibrium models (GLOBE, GTAP) to analyse similar policy questions with different tools, which allows the comparison of results in order to substantiate the findings.

3.2 Updated List of Modules

3.3 Module Description

3.3.1 D1. Agricultural Research Data Index Tool (ARDIT)

3.3.1.1 Main goal

The main objective of ARDIT is to provide a publicly accessible index of agricultural data sources. This includes on the one hand the query side, aimed at detecting data sources that include specific variables with a given geographical scope and spatial/statistical resolution. On the other hand, it should also allow the user to include new data sources, following a characterisation procedure that, once completed, allows the registration of the new data source item in the index.

3.3.1.2 Current implementation strategy

ARDIT is a browser-based web application, built using Spring framework for the server-side code in Java, and Angular framework to build a web client using TypeScript, HTML and CSS. The server-side code is built under a REST API, hence the separation between the server and client-side. For communication between both sides, HTTP requests are used to transfer data in JSON format. In addition, the platform uses PostgreSQL as the database manager, Docker to automate the deployment of the application in all the different software environments and LDAP (Lightweight Directory Access Protocol) for the authentication on the platform and for storing user credentials.

3.3.1.3 Inputs / outputs

- From the ARDIT User Interface: Characterisation of new or edited data sources, following a tagged and predefined format according to the AGRICORE DCAP 2.0 ontology.
- To the AGRICORE Interface: List of Dataset-Dataset_Variables available to assign values to each of the attributes of interest of all agents in a synthetic population given a user predefined spatial, temporal and financial-economic scope.
- To the DWH: The corresponding ETLs for the DWH to be able to obtain the data.

3.3.1.4 Module Requirements

Module D1 Requirements are listed in Annex A.

3.3.2 D2. Data Warehouse (DWH)

3.3.2.1 Main goal

As its name suggests, this module constitutes the centralized point for the storage and exchange of non-volatile data within AGRICORE's IT architecture. It has a public part, for open data, and a private part for the storage of data with access restrictions. The DWH has a particularly relevant role in the process of building synthetic populations, since it constitutes the warehouse for the raw data of all the datasets that are necessary to fill in the attributes of the agents (and that the Big Data Extraction Module (D.3) must extract), as well as for the intermediate data derived from the Big Data Fusion (D.4) process (definitions of the simple and compound probability distributions, the definition of the Bayesian Network, etc.). Finally, once the synthetic agent population has been constructed (D.5), its representation in file format is also stored in the DWH.

Additionally, using ETLs scripts provided by the user, the DWH connects to the ARDIT (D.1) to download and adapt the necessary data from the corresponding data sources. Simulation results, both intermediate and final, are also stored in the DWH. These results can come directly from the ABM simulation engine (D.6) or be the result of an additional calculation in the impact assessment modules (D.9).

Finally, and although not explicitly represented in the Figure (Updated Architecture), the DWH connects to the AGRICORE graphical interface (D.12) to allow interactive configuration of the simulation, visualization of the data, and uploading and downloading of intermediate files.

3.3.2.2 Current implementation strategy

The DWH provides distributed data processing using massive parallel programming technologies such as Hadoop and Spark, combined with a set of Big-Data-based technologies for data ingestion, processing and storage. The solution proposed for the DWH combines certain tools from the Hadoop framework, such as the HDFS, YARN or MapReduce, along with Spark, to implement the distributed storage system. In addition, other technologies have been added such as Hive, which allows queries in SQL-like format on the data stored in the HDFS, or Sqoop, a tool that allows external relational databases to be dumped into the distributed system, or vice versa, exporting data stored in the DWH to external relational databases. Finally, Apache Zeppelin has been added as the main tool for interacting with the DWH. Zeppelin is a web-based notebook that allows to launch queries, perform different operations and display data in charts, graphs or plots using a programming language like Python, Scala or SparkSQL, among others.

3.3.2.3 Inputs / outputs

- From the ARDIT: The addresses (URIs and URLs) that allow the DWH to access and download (via distribution or data service) the dataset variables selected by the user to assign values to the attributes of interest. Also, the ETLs stored in the private or public side of the ARDIT.
- To Big Data Extraction Module: the different variables required, encoded and packaged in one of the formats supported by DEM.
- From DEM/To DFM: A file/stream with the aggregation of the data of all variables, encoded and packaged in the format supported by the Data Fusion Module.
- From DFM/to SPG: One or several files containing: i) the order in which the values of those attributes that are correlated should be generated, as well as the list of those that are completely independent of the others and could be generated individually and in parallel, and ii) the probability distributions necessary to generate those attributes, encoded as mathematical formulae or as marginal tables.
- From SPG/to ABM Simulation Module: The file containing the resulting synthetic population (all agents and their attribute values). This file can be very large, so alternatives for its generation and fragmented storage should be studied.
- To IAMs: Simulation results, encoded and packaged in the required format, which are passed as arguments to the corresponding IAM.
- From IAMs: Key Performance Indicators resulting from the execution of the models contained in each IAM using all or part of the simulation results as input arguments.
- From/to AGRICORE Interface:

3.3.2.4 Module Requirements

Module D2 Requirements are listed in Annex A.

3.3.3 D3. Data Extraction Module (DEM)

3.3.3.1 Main goal

This module consists of a set of tools that allow end-users to manage the metadata information and extract/process data stored in the DWH even if they do not have advanced knowledge of data engineering. This metadata help users to know what data is stored in the DWH, and what variables are found in these data files, which are stored in a distributed way in the DWH. For this purpose, a graphical interface is provided for the visualization of the metadata related to the data stored in the DWH. Additionally, users are given the possibility to create new datasets from existing ones in a graphical and intuitive way, without the need to master the native low-level programming big data processing tools of the DWH. The storage location of the generated datasets will be determined by the end-user, being possible to store it in the DWH, in the cloud or in the user's own computer. Some of the operations available through this module are:

1. Parallelly access each of the datasets (they can be stored in the public part of the DWH, or the private part of the DWH, or even require downloading through a data service).
2. Filter the dataset to select only the fields corresponding to the required variables.
3. Extract the data and if necessary format it for further processing.
4. Aggregate the data from all accessed datasets and store them in another dedicated area in the private part of the DWH.

These functionalities should be available for variables with different data types: numeric, plain text, geo-referenced coordinates, etc.

3.3.3.2 Current implementation strategy

DWH contains a large amount of data stored in a distributed manner. Each data comes from a data source, which has been identified from ARDIT. This module enables to store and visualise all these metadata that allow to identify the data source with the content that contains it (variables, time horizon of this data, ...). This metadata will be managed by some tool that could be a database or even a simple text file and will be updated when a new data source is added to the DWH. This module will provide a graphical interface that allows the user to visualise all these metadata in an easy and intuitive way, giving the possibility to create new datasets from the existing ones in the DWH without the need of having data engineering knowledge. The computational load required to create these datasets will be supported by the DWH, as this module serves to execute requests on the DWH in the easiest and most intuitive way.

In terms of the implementation strategy, it is proposed that DEM be a microservice decoupled from the data warehouse and encapsulated in the form of a container. This microservice should have three functionalities: 1) update the metadata of the information every time a new data source is entered in the DWH, 2) display through a graphical interface all the metadata about the data stored in the DWH, 3) combine the existing data in the DWH using this graphical interface, without the need to know how the BigData processing tools used in the DWH work.

3.3.3.3 Inputs / outputs

- From DWH: Information about the imported data sources and the variables or other useful information that they comprise.
- From AGRICORE Interface (maybe via consultation to ARDIT to know which variable within the selected dataset): List of Dataset-Dataset_Variables selected by the user to cover all the attributes of interest of the agents.
- To DWH: A file/stream with the aggregation of the data of all variables, encoded and packaged in the format supported by the Data Fusion Module (.csv).

3.3.3.4 Module Requirements
Module D3 Requirements are listed in Annex A.

3.3.4 D4. Data Fusion Module (DFM)

3.3.4.1 Main goal
The objective of this module is to obtain the mathematical artefacts that allow the SPG to generate the (pseudo)random values that are assigned to each of the agents' attributes. To this end, given the aggregation of data corresponding to the relevant variables extracted from the DWH by the DEM, the DFM obtains the probability distributions of each variable. Additionally, it can also obtain the joint probability distributions (marginal or conditional distributions) of those groups of variables between which statistical correlation is detected. Both automated parametric and non-parametric fitting methods (Kernel Density Estimation) are considered for the adjustment of these distribution functions.

3.3.4.2 Current implementation strategy
At a conceptual level, the strategy that has been defined is the integration of the process of obtaining probability functions within a broader process that seeks to obtain the Bayesian Network (BN) of the aggregate of variables. A Bayesian Network is a graphic mechanism (although translatable into text format) that allows representing a group of variables (through nodes) and the dependence relationships between them (by means of lines with arrows). Thanks to this, not only those joint probability distributions to be adjusted are detected, but also the sequence in which the values of those attributes that show interdependence must be generated (starting with the ancestor variables and continuing with their descendants). BNs are learned utilizing the hybrid Forward Early Dropping Hill Climbing (FEDHC) algorithm [11].

At the technical implementation level, the scripts that perform the BN extraction are written in R language using the *pchc* package [12].

3.3.4.3 Inputs / outputs
- From DWH: A file/stream with the aggregation of the data of all variables, encoded and packaged in the format supported by the Data Fusion Module.
- To DWH: One or several files containing: i) the description of the generated Bayesian Network (in textual format) ii) the order in which the values of those attributes that are correlated should be generated, as well as the list of those that are completely independent of the others and could be generated individually and in parallel, and iii) the probability distributions necessary to generate those attributes, encoded as mathematical formulae or as marginal tables.

3.3.4.4 Module Requirements
Module D4 Requirements are listed in Annex A.

3.3.5 D5. Synthetic Population Generator (SPG)

3.3.5.1 Main goal
Once the probability distribution functions of all attributes are available, as well as the order in which they should be generated, the objective of this module is to create the global population of agents of the requested size with all their assigned attributes, and to make sure that this synthetic population accurately reflects (with some degree of fit) the real population. The latter is ensured by using the Synthetic Reconstruction (SR) technique, in which iteratively, the fit of the probability distribution functions of the attributes of the synthetic population with the probability distribution functions of the same attributes of the real population is checked; If the

difference is greater than a certain acceptable threshold, a number of agents are 'reconstructed' (re-generated) (either randomly or those whose attributes are further away from the normal value of the real population) until the synthetic population is within the acceptable range of fit.

3.3.5.2 Current implementation strategy

At the implementation level, the SPG will be a Python package composed of:

- A main script that generates as many empty agents as the required population size. It will then create a number of parallel instances of the value assignment script.
- Such a script, taking as argument the attribute generation sequence given by the BN, assigns in the same order a random value for each attribute based on the probability distribution (individual, joint, conditional) drawn for it. The set of these distributions is also an input argument.
- A statistical accuracy check script that calculates the resulting probability distributions in the synthetic population for each attribute, and compares them with the analogous distributions in the real population.
- A script that, in case the difference is greater than tolerable according to a pre-set threshold, selects a set of agents whose attributes will be re-populated. The process is repeated until it falls below the threshold.

3.3.5.3 Inputs / outputs

- From DWH: One or several files containing: i) the order in which the values of those attributes that are correlated should be generated, as well as the list of those that are completely independent of the others and could be generated individually and in parallel, and ii) the probability distributions necessary to generate those attributes, encoded as mathematical formulae or as marginal tables.
- From AGRICORE Interface: The required Goodness of Fit for the SP.
- To DWH: The file containing the resulting synthetic population (all agents and their attribute values). This file can be very large, so alternatives for its generation and fragmented storage should be studied.

3.3.5.4 Module Requirements

Module D5 Requirements are listed in Annex A.

3.3.6 D6. Agent-based Model Simulation Engine (ABM-e)

3.3.6.1 Main goal

This module will simulate the evolution of the ABM population according to the AGRICORE agent dynamics previously discussed in this section. The implementation of this module will be fully object-oriented and the agents will be instantiated according to the synthetic population already generated in the preceding modules. On runtime, this module will connect to a mathematical solver in order to perform the iterations needed to simulate the evolution of the agents and it will also manage the interactions required with the external modules (land, markets, environment, etc). Given that the simulation of an EU-scale case study would require the computation of a very large number of agents, this module will allow a set of high-performance computing (HPC) features, including:

- The exploitation of parallel processing and cloud computing environment capabilities, taking advantage of the cost-competitiveness of today’s cloud computing services which provides affordable high computational power.

- The execution in GPU-based (Graphics Processing Units) HPC architectures, which is a possibility to be explored during the architecture design stage. In fact, while CPUs typically consists of up to dozens of cores (typical ones currently below 10), the GPU consists of hundreds of smaller cores which provide a massively parallel architecture and which has found many applications in computer science problems.
- The use of the latest releases of best rated off-the-shelf mathematical solvers. Attending to the partners' experience, the IBM CPLEX MIQCP solver seems the best selection in terms of performance given the specific characteristics of the problem to be solved. Nevertheless, other possibilities will be considered as well, such as the GUROBI solver.
- The implementation of warm-start techniques, which consists of optimised calls to the solver taking advantage of tentative solutions already available from other similar agents. This measure has been previously tested by data scientists within the IDE and AAT partner teams in other ABM applications with remarkable success.

3.3.6.2 Current implementation strategy

The ABM simulation engine is formed by the set of agents and the interfaces that communicate them with the modules they need to run it. Each agent is an instance of the Farm Holding class, which in turn may contain one or more objects of auxiliary classes (LandPlot, FarmManager, FarmOwner, CropEnterprise, LivestockEnterprise, etc.).

The methods of each FarmHolding agent will include (method names are tentative):
- *campaignPlanning*, which constructs the optimisation problem and passes it as an argument in a solver call.

- *campaignSimulation*, which simulates a campaign (calculates production and disturbances) using the actions decided by each agent in the previous method and agro-climatic conditions obtained through the Climate Model Connector. Outputs are calculated using the connector to biophysical models and also simulating the interaction among agents through the multi-market modules.
- *postCampaignEvolution*, which updates the state of each agent using the actions implemented and the production/profitability obtained. The evolution of the state of the supra-agent ecosystem is also simulated. Both make use of the external biophysical models through the dedicated connector.

3.3.6.3 Inputs / outputs

- From DWH: The file containing the resulting synthetic population (all agents and their attribute values). This file can be very large, so alternatives for its distributed processing and fragmented volatile storage should be studied.
- From Policy Environment Module: The list of the instrument(s) and policy measures that the user wishes to simulate, so that the ABM can translate them and incorporate them by modifying the objective function of all or some of the agents (alternatively, if the translation were done by the PEM instead, the ABM-e would directly receive the cost function modified to include the mathematical transliteration of these measures to be simulated). This input is received only once, before starting the simulation, as it is actually part of its set-up process.
- From/to Biophysical Models (throughout Biophysical Model Connectors): The ABM-e shall exchange information with the BM throughout the simulation run. At the beginning of each simulation step (at the start of each new agricultural season) queries are performed as part of the optimisation calculations that mimic the decision-making process of each Holding Manager. At the end of each simulation step, queries are performed as part of the dynamic state update to compute the effect of the actions taken by the agents on the biophysical environment (soils, available water, variation of livestock and wildlife, etc.).

- To DWH: The simulation results consist of:

  - The state of all agents after each of the simulation steps (value of their attributes in all k).

  - The agro management actions taken at each optimisation/control instant.

  - The log of operations and/or prices resulting from the auxiliary markets (LM and PM).

  - Others (to be delimited).

  All of them must be stored at the end of the simulation and as a previous step to the launching of the visualisation. The results archive can be even larger than the SP archive, so it is even more important to study alternatives for fragmented storage and processing.

- To IAMs: The IAMs need a part of the simulation results of the agents as input for the calculation of the KPIs. Specifically, in the intermediate simulation steps, KPIs can be calculated after each simulated agricultural campaign and dynamically displayed in the simulation interface.

- From/to Solver(s): The solvers will receive, for each optimisation instant and in the format they accept, the set-up of the optimisation problem corresponding to each agent (its objective function, with the updated values of its parameters and the constraints with their limit values). The solver will return, for each agent, the optimal solution to the optimisation problem posed, which in turn constitutes the set of agri-management decisions to be applied to the Holding in the next simulation interval. In the event that many agents of a similar type have to solve very similar optimisation problems, and after solving each one of them, the solutions obtained for the previous ones can be added to the optimisation set-up of the following ones, so that the solver starts searching from them. This technique is known as warm-start, and it considerably reduces the problem-solving times of each individual agent and thus the overall simulation time.

- From/to Multi-Market Modules: Throughout the simulation, and at the designated times, the ABM-e is connected to the multi-market modules to compute the effect of agents' buying/selling intentions at the overall market level. The process of land exchange (by buying and selling or renting and leasing) is carried out at the beginning of each simulation step, before the calls to the solver, so that as soon as they occur, the agent already knows how much land productive factor he/she has. The call to the product market, on the other hand, occurs at the end of each simulation step, which is when the biophysical model has returned the simulated production result corresponding to the combination of agro-management decisions and bio-climatic conditions.

- To AGRICORE Interface: In all cases, constant communication is maintained to indicate the status of the simulation (active, completed, error) and the degree of progress (percentage of the simulation completed and/or estimated time remaining). Additionally, in those cases where the user requires it, KPIs calculated on the fly can be sent to be displayed by the graphical visualisation sub-module of the AGRICORE interface.

3.3.6.4 Module Requirements

Module D6 Requirements are listed in Annex A.

3.3.7 D7, D8, D9 - Multi-Market Modules

3.3.7.1 Main goal

This set will account for the interactions of the agents with the rest of the agricultural structures, and will initially include the next modules:

- Land Market, which will model the use and transfer of land resources among agents. In this module, agents will be able to exchange their land plot (parcel) objects, either permanently (buy-sell market) or temporarily (lease-rent market).
- Product Market, which will implement a model to simulate the dynamics of production market prices and will also include additional market dynamics such as manure, fodder, and young animals.
- Production Factors Markets: This market will model the variations in the price of factors of production produced by the aggregate effect of the agro-management decisions of all agents at the national level or above.

3.3.7.2 Current implementation strategy

Land exchange markets will be simulated through zoned double auctions (DA). That is, potential sellers/landlords, include all available plots of land in a given geographical vicinity in the same market session, in which all potential buyers/lessors for whom it is profitable (due to their proximity to the rest of their land) to buy/lease in that area also participate. It remains to be determined whether these zoned markets will be simulated with continuous double auctions (CDAs), in which participants continuously update their offers (bids and asks) throughout the session and transactions occur individually but continuously, or with a succession of discrete double auctions (DDAs), in which participants update their offers (bids and asks) after each discrete session, in which several transactions may occur. In any case, each agent will make use of a dedicated Trading Agent, which is an instance of a Python class that comprises all the non-native trading functionalities of the agents.

Product Markets will be simulated taking advantage of existing CGE models. Global/national/regional supply is calculated incorporating the aggregated production of agents within the ABM-e; non-European (imported) supply and global demand are generated interpolating from pre-existing forecasts.

3.3.7.3 Inputs / outputs

- From ABM Simulation Engine (throughout Inter Module Communication Module):
    - List of participating agents in the Land Markets containing, for each: intention to buy/sell - land requirements - maximum budget/minimum valuation. This information (and perhaps some additional information such as the agent's risk aversion coefficient) will be used by the virtual trading agents (one for each agent) to update the offers during the session.
    - Individual production amounts for all agents and all enterprises (crop derived products and/or livestock derived products).
    - Production factors required by all agents (fertilizers, machinery, combustibles, etc.).
- To ABM Simulation Engine (throughout Inter Module Communication Module):
    - List of land parcels that change ownership for the following campaigns or permanently, including the transferring agent and the acquiring agent. This list must also include, for each parcel of land exchanged, the price of the transaction, so that it can be summed up/deducted from the corresponding agents.

  - Averaged campaign prices for each simulated product, considering the aggregation of supply by all simulated agents but also the external supply (importations) and the expected aggregated demand.
  - Averaged campaign prices for each required productive factor.

3.3.7.4 Module Requirements

The requirements of Modules D7, D8 and D9 are listed in Annex A.

3.3.8 D10, D11, D12 - Impact Assessment Modules

3.3.8.1 Main goal

The objective of IAMs is to translate simulation results into impact effects at various levels of interest for multi-sided policy analysis. In this sense, IAMs are completely independent modules of the simulation, and their use is not required in order to carry out the simulation. By contrast, provided that the result files of a completed simulation are available, IAMs can be run at any time thereafter.

So far, the following classification has been established for potentially embeddable IAMs:

- D10. Environmental / Climate IAMs, which will compute main KPIs related to the environmental and climatic impact assessment. This category might include, for example, GHG emission calculation modules, agricultural water pollution modules, land exhaustion modules, etc.
- D11. Socio-economic IAMs will assess the relationship between policy instruments and KPIs related to the integration of agriculture in rural systems. This category might include, for example, agricultural labour effect modules, modules that compute the effects on specific socio-demographic strata (young farmers, female farmers), etc.
- D12. Ecosystem services IAMs, which will model and provide ecosystems services KPIs categorised in four main areas including supporting, provisioning, regulating and cultural services. This category might include, for example, modules that compute effects of organic farming on the populations of endangered species, pollination enhancement calculation modules, modules for the calculation of the increase in agro-tourism etc.

3.3.8.2 Current implementation strategy

The Impact Assessment modules will be composed of Python implementations of existing theoretical models in the literature, as well as connectors to other models already implemented as software.

3.3.8.3 Inputs / outputs

- From DWH/ABM Simulation Module: IAMs use suitably formatted simulation results files as input to produce impact assessment KPIs as output. These results files can be stored in the private part of the user's DWH, or uploaded as a file directly from local or another cloud by the user.
- To DWH/AGRICORE Interface: The resulting KPIs can be downloaded as a file for storage in the DWH or in any other physical location upon request. Additionally, they can also be sent to the visualisation module of the AGRICORE interface for graphical presentation and/or advanced processing.

3.3.8.4 Module Requirements

The requirements of Modules D10, D11 and D12 are listed in Annex A.

3.3.9 D13. Climate Model Connectors

3.3.9.1 Main goal

The objective of this module is to enable the use of climate and weather models in the form of microservices within the AGRICORE suite. Such models can be used, among others, for:

- Generate a trajectory or set of trajectories of annual weather conditions, either for use as scenarios in stochastic optimisation or for use as conditions for the current simulation campaign.
- Compute the effect of agents' actions on weather conditions, either to incorporate them dynamically in long-span simulations (effect of previous simulation steps on current and future simulation steps) or to compute long-term effects (from the end of the simulation onwards).

In principle, the climate models to be connected are coupled atmosphere-ocean General Circulation Models (AOGCMs), such as HadCM3, EdGCM, GFDL CM2.X, ARPEGE-Climat. In these models, non-agricultural emission projections are left fixed and the contribution of GHG emissions from agriculture is varied by extrapolating the simulation results.

3.3.9.2 Current implementation strategy

Still to be defined.

3.3.9.3 Inputs / outputs

- From the ABM-e Simulation Module and/or the IAMs: aggregated simulation results and extracted KPIs by the IAMs to predict the effect of the simulated agricultural activity on the evolution of the climate in the medium and long term.
- To the ABM-e Simulation Module: probable short- and medium-term (1-12 months) weather trajectories, so that they can be used as perturbations for the simulation. These trajectories will possibly include a sequence of expected mean values for temperature, solar radiation, precipitation and wind speed. These trajectories can also be expressed as an average value and a confidence interval.

3.3.9.4 Module Requirements

The requirements of Module D13 are listed in Annex A.

3.3.10 D14. Policy Environment Module

3.3.10.1 Main goal

This module, in its POSITIVE configuration, should allow the incorporation of different policy measures and instruments into the AGRICORE agent simulation. These instruments are translated and incorporated in the form of modifications to the objective function that mimics the decision process of each agent. As a result of the simulation, a series of Key Performance Indicators will be obtained that allow comparing the effect of some measures with that of other alternative measures.

In its NORMATIVE configuration, and given a set of global constraints for the implementation of public agricultural policies (e.g. maximum budget, foreseeable available manpower, etc.), this module should allow, based on ABM simulation, to determine the optimal set of instruments and measures (and the optimal values of their parameters) to maximize a set of objective KPIs in a given time period. For this purpose, the set of potentially selectable policy instruments and measures must be indicated prior to the simulation and included in parameterised form.

For both cases, this module must constitute a catalogue of pre-existing policy instruments (real measures applied in the past or currently in application). It must also allow the modification of these instruments (by varying the values of their parameters). Finally, it must allow the user to create his own fully customized policy from scratch. This module translates the policy schemes of interest into the AGRICORE simulation environment. To that end, it is connected to the ABM simulation module (D.6) in order to introduce the necessary modifications of the agents' model structures as a previous step to the agents' instantiation. This basically consists of converting the policy instrument(s) selected by the user into modifications to be applied to the ABM's objective function which, introduced in each agent, mimics the decision process of its managers. The nature of these modifications may vary according to the type of policy instrument, and affect either the structure of the objective function (e.g. incorporation of additive terms to simulate production premiums, variation of depreciation coefficients, etc.) or its constraints (e.g. limits to the value of production variables to simulate production quotas), or a mixture of both.

3.3.10.2 Current implementation strategy

During the simulation set-up process through the AGRICORE graphical interface, the user can consult a public repository with definitions of policy instruments previously created by other individual users, or by academic organisations or policy-making institutions. Users will be able to save a copy of the definition of those policies of interest to them in their own private repositories. Once in their private repository, they can modify the value of the policies' parameters or leave them in their original definition. Users can also define their own policy instruments in a controlled and tagged format (XML/JSON), store them in their private repository and publish them for inclusion and indexing in the global public repository. Finally, the user can select, for further translation and incorporation into the simulation, one or more of the existing policy instruments in his/her private repository.

In order for the module to provide the policy translation service, work is currently underway to define a format based on tagged language (XML/JSON) to harmonize the formal description of a wide range of regional, national and even EU public policies.

3.3.10.3 Inputs / outputs

- From the AGRICORE Interface: i) the selection of the configuration type (Positive or Normative), ii) the selection of configuration set-up (policy instruments and its parameter values for positive configuration; policy planning constraints and objectives to be achieved with the chosen policy combo for the normative configuration), iii) new/edited policies characterisations.
- To the ABM Simulation Module: Updated version of the objective function to be incorporated in each type of agent, which will be modified according to the combo of policies selected by the user. Modifications can be in the form of additions to the function, hard constraints added to the constraint set, or soft constraints that modify the form of the objective function itself.

3.3.10.4 Module Requirements

The requirements of Module D14 are listed in Annex A.

3.3.11 D15. Biophysical Model Connectors

3.3.11.1 Main goal

The aim of the biophysical modelling module is to incorporate into the simulation the biophysical dynamics of the ecosystem in which the agents being simulated are located. In turn, this objective is manifested in two other objectives:

- To incorporate into the decision process of holding managers the (limited) knowledge they have about how their land/herds behave in the face of different combinations of crops and/or applied production factors. In other words, to integrate how managers predict what the production will be according to the productive factors they use and how they balance maximising the yield of their land while avoiding stressing it.

- Simulate the temporal evolution of the dynamics of the ecosystem, as a result of the aggregate effect of the decisions of all the holdings located in it (agro-management decisions) and of external factors (climatic disturbances, pests, etc.).

3.3.11.2 Current implementation strategy

Still to be defined.

3.3.11.3 Inputs / outputs

- From the ABM-e Simulation Module or other modules: Model queries, passing as arguments the status of the land plot or livestock herd together with a combination of agro management decisions that would be applied on them.
- To the ABM-e Simulation Module or other modules: The models return the effect of these decisions on the future state of the land plot/livestock herd as well as the immediate resulting productivity.

3.3.11.4 Module Requirements

The requirements of Module D15 are listed in Annex A.

3.3.12 D16. AGRICORE Interface Module

3.3.12.1 Main goal

This module will centralise the interaction of the users with the AGRICORE suite, so the development of future modifications, extensions or even alternative interfaces is facilitated.

The interface shall allow all user types at least the following operations:

- Create a user account and manage password recovery.

- Access the AGRICORE suite using their user credentials.

- Connect to ARDIT to include new characterisations or to perform manual queries.

- Prepare a new simulation by defining in a non-sequential way the 5 necessary elements (inclusion of a synthetic population, selection of policies to be simulated, selection of the solver and the biophysical models to which the agents will be connected, selection of KPIs to be automatically calculated and selection of the period to be simulated).

- Launch the simulation once it has been configured. To be able to observe the progress of the simulation, pause it or cancel it.

- Launch the visualisation of results directly at the end of a simulation, or by loading a results file generated in a previous simulation.

Additionally, users should be able to access from the general interface a separate sub-module to assist them in the process of creating new synthetic populations. However, this process may include extensive and fully manual data processing actions not based on the use of the graphical interface, which in practice limits the use of this sub-module to advanced users (with sufficient statistical, econometric and agronomic expertise).

3.3.12.2 Current implementation strategy

AGRICORE's interface implementation follows a complete process of requirements gathering, the design of the interface itself, its validation and subsequent development. The AGRICORE interface will be implemented as a cross-platform desktop application using web technologies, following best design and usability practices. Electron framework [13]will be used to enable the utilisation of the same tools, libraries, plugins or components used for website development. Electron renders web pages, which would normally be displayed in the web browser window, on the windows and menus of the operative system where it runs. It uses Chromium as a rendering engine and Node.js for backend side code, so it allows to build the complete application using only JavaScript, HTML and CSS.

In addition, data visualisation tools, libraries and other tools will be used and combined to display charts, plots, graphics, maps and geo-referenced information.

3.3.12.3 Inputs / outputs

- From the user: All input data necessary for the operation of the various sub-modules offering user interface, graphical or console-based.
- To the user: Confirmation or error messages related to the configuration, in graphical or console text format. Also, output data, displayed in table or graphical format.

3.3.12.4 Module Requirements

The requirements of Module D16 are listed in Annex A.

4 Interfaces

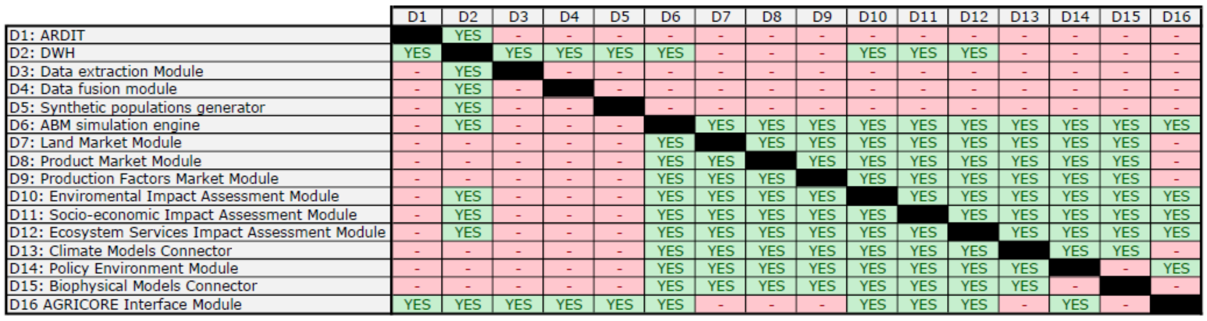
4.1 Introduction

Generally speaking, "an interface is a boundary where, or across which, two or more parts interact, so that one system affects or is affected by a design feature of another system" [14]. Speaking specifically about software engineering, "an interface defines the signature operations of an entity, it also sets the communication boundary between two entities, and generally refers to an abstraction that an asset provides of itself to the outside" [15]. From a requirements point of view, whenever the words "interact" or "affect" need to be included in the wording of a requirement, an interface is involved. Such a requirement is therefore classified as an interface requirement.

There are many types of interfaces. Some of the most common are mechanical interfaces, electrical interfaces, electromagnetic compatibility interfaces, organisational interfaces, data communication interfaces and human-machine interfaces. In the development of the AGRICORE suite, only the latter two appear, although the definition of organisational interfaces may be necessary for the maintenance and governance of the tool after the end of the project. The definition of the human-machine interfaces in AGRICORE corresponds to the requirements of the graphical user interface of the project, which is being addressed within task T4.2 and will be embodied in deliverable 'D4.3 - Validated design for the AGRICORE interface (M29)'. The rest of the data communication interfaces are presented next in this chapter.

4.2 Interface Matrix

The first step in defining interfaces in a multi-modular development is the identification of interfaces. This first step includes an analysis of the suite and how it relates (interacts) with the systems that complement it (external interfaces) and an analysis of the modules that make up the suite and how they interact with each other (internal interfaces). The AGRICORE 'Modules Communication Matrix' is presented below. The existence of communication in this matrix implies the existence of an internal interface, although some of them (D8, D13, D15) also indicate the existence of an external interface with models or systems external to the project.



	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15	D16
D1: ARDIT	■	YES	-	-	-	-	-	-	-	-	-	-	-	-	-	-
D2: DWH	YES	■	YES	YES	YES	YES	-	-	-	YES	YES	YES	-	-	-	-
D3: Data extraction Module	-	YES	■	-	-	-	-	-	-	-	-	-	-	-	-	-
D4: Data fusion module	-	YES	-	■	-	-	-	-	-	-	-	-	-	-	-	-
D5: Synthetic populations generator	-	YES	-	-	■	-	-	-	-	-	-	-	-	-	-	-
D6: ABM simulation engine	-	YES	-	-	-	■	YES	YES	YES	YES	YES	YES	YES	YES	YES	YES
D7: Land Market Module	-	-	-	-	-	YES	■	YES	YES	YES	YES	YES	YES	YES	YES	-
D8: Product Market Module	-	-	-	-	-	YES	YES	■	YES	YES	YES	YES	YES	YES	YES	-
D9: Production Factors Market Module	-	-	-	-	-	YES	YES	YES	■	YES	YES	YES	YES	YES	YES	-
D10: Enviromental Impact Assessment Module	-	YES	-	-	-	YES	YES	YES	YES	■	YES	YES	YES	YES	YES	YES
D11: Socio-economic Impact Assessment Module	-	YES	-	-	-	YES	YES	YES	YES	YES	■	YES	YES	YES	YES	YES
D12: Ecosystem Services Impact Assessment Module	-	YES	-	-	-	YES	YES	YES	YES	YES	YES	■	YES	YES	YES	YES
D13: Climate Models Connector	-	-	-	-	-	YES	YES	YES	YES	YES	YES	YES	■	YES	YES	-
D14: Policy Environment Module	-	-	-	-	-	YES	YES	YES	YES	YES	YES	YES	YES	■	-	YES
D15: Biophysical Models Connector	-	-	-	-	-	YES	YES	YES	YES	YES	YES	YES	YES	-	■	-
D16 AGRICORE Interface Module	YES	YES	YES	YES	YES	YES	-	-	-	YES	YES	YES	-	YES	-	■

Figure 11 Module communication traceability matrix

4.3 Interface Descriptions

In order to define each pre-identified interface, it is necessary to define the characteristics of each system at the interface, the media involved in the interaction and the characteristics of what crosses the interface. When, as in the case of AGRICORE, the different modules are being developed simultaneously (none of them exists yet), the definition of the interfaces becomes more complicated. The definition of the interface between the modules will evolve over time as the design matures. Initially, work is done in the problem space, and the focus is on the "what", not the "how". In terms of the interaction between the two systems, what information do you need to define and what can you define so that the design team can design the interface? Once the design of the modules is complete, the interfaces will be defined in figures, tables and drawings. Prior to design, these definitions must be documented and controlled somewhere, which is what is done below.

For some pairs of modules, the envisaged communication between them is presented; for other modules (those whose microservices will be connected through DAPR) only the services (and the corresponding API that each module exposes) are presented. It should be stressed that all these interface definitions belong to the pre-design phase and are therefore completely tentative.

4.3.1 ARDIT (D1)

```
// The finder of available Datasets to fill agents' Attributes of Interest
service Finder {
    // Detect those datasets that contain information related to a specific
agent's AoI
    rpc FindDatasetsFor (AttributeRequest) returns (ListOfDatasets) {}

// The request message containing the AoI name.
message AttributeRequest{
    string attributeName = 1;
}

// The response message containing the list of (1+) datasets containing at
least one dataset variable that could be used for populating the AoI.
message ListOfDatasets{
    repeated string datasetName = 1 [packed=true];
}
```

4.3.2 DWH (D2) ↔ DEM (D3)

DWH (D2) will store large amounts of data, many of them with different structures and different variables. To be able to identify which variable corresponds to which data source, the DEM (D3) includes a metadata storage tool that allows identifying the data source with the variables in it (e.g. NoSQL database, text file). This tool provides a graphical interface that enables the user to execute orders in the DWH (D2) via DEM (D3) without the need for technical knowledge of the DWH (D2). Thanks to this tool, anyone without data engineering knowledge will be able to interact with the data warehouse in a simple and intuitive way. However, the set of allowed operations, which needs to be defined yet, might be restricted.

4.3.3 DWH (D2) ↔ DFM (D4)

DFM (D4) needs the data stored in DWH (D2) to generate the mathematical artifices that drive the generation of the values to be assigned to each of the agent's attributes during the generation of synthetic populations. To this end, DFM (D4) will interact with DWH (D2) through the exporting tools provided within the DWH (D2), like Sqoop, Spark or Hive. Once the data has been imported into DFM (D4), this module would be able to use it to perform operations as needed. The interaction between the two modules will be detailed in the future, once a common methodology for importing and exporting data is established.

4.3.4 DWH (D2) ↔ SPG (D5)

SPG (D5) needs the data stored in DWH (D2) to generate the synthetic population. To this end, SPG (D5) will interact with DWH (D2) using the exporting tools provided within the DWH (D2), like Sqoop, Spark or Hive. Once the data has been imported into SPG (D5), this module would be able to use it to perform operations as needed. The interaction between the two modules will be detailed in the future, once a common methodology for importing and exporting data is established.

4.3.5 DWH (D2) ↔ ABM-e (D6)

AMB (D6) needs to obtain a Synthetic Population (SP) stored in DWH (D2) to simulate the proposed scenario. To this end, ABM (D6) will interact with DWH (D2) using the exporting tools provided within the DWH (D2), like Sqoop, Spark or Hive. Once the data has been imported into AMB (D6), this module would be able to use it to perform operations as needed. Once the simulation is finished, the complete results for all agents need to be saved into the DWH. Also, during the simulation, the ABM might require to save partial results, especially for simulations involving a large number of agents whose complete results could overflow the RAM. In any case, the ABM will interact with the DWH using the importing tools provided by the latter. The interaction between the two modules will be detailed in the future, once a common methodology for importing and exporting data is established.

4.3.6 ABM-e (D6)

```
// The simulation status checker service
service AMBstatus{
    // Get status of the simulation at a particular instant, or get its
percentage of completion.
    rpc getStatus () returns (ActualStatus) {}
    rpc getPercentage () returns (ActualPercentage) {}

// The response message containing the current status.
message ActualStatus{
    Status status= 1;
}

// Enum of possible simulation status
enum Status {
    IDLE= 0;
    RUNNING = 1;
    STOPPED= 2;
    FINISHED = 3;
}
// The response message containing the current progress level.
message ActualPercentage{
    int percentage= 1;
}
```

4.3.7 Land Market Module (D7)

```
// The Land Plot Market service definition
service Buyer {
    // Express the intention to buy up to certain aerea of Land
    rpc LandPlotDemand (LandDesire) returns (ActionResult) {}
    // Express the intention to sell up to certain aerea of Land
    rpc LandPlotSupply (PlotOffered) returns (ActionResult) {}

// The message containing the buyer's intentions data.
message LandDesire {
    int32 LandAreaDesired = 1;
    int32 AvailableBudget = 2;
    int32 BuyerLocation = 3;
    int32 minimumYield = 4;
    int32 maximumDistance2BuyerLocation = 5;
}

// The message containing the seller's intentions data.
message PlotOffered {
    int32 PlotAreaOffered = 1;
    int32 privateValuation = 2;
    int32 PlotLocation = 3;
    int32 Yield = 4;
}

// Enum of result
enum ACK{
    OK= 0;
    FAIL= 1;
}

// The response message containing the result of the user's 'shout' (received
and registered or failed)
message ActionResult {
    ACK result = 1;
}
```

4.3.8 Product Markets Module (D8)

```
// The product price predictor service
service MarketProduct{
    // Send a request for the product markets module to estimated the product
price.
    rpc CalculateMarketPrice (AgentsProduction) returns (EstimatedProductPrice)
{}

// The request message containing a list .
message AgentsProduction{
    repeated ProductQuantity productQuantity = 1;
}

// The request message containing the list of produced amounts.
message ProductQuantity {
    string ProductName= 1;
    float Quantity= 2;
}

// The response message containing a list of ProductPrice, with their names and
values.
message EstimatedProductPrice{
    repeated ProductPrice productPrice= 1;
}

// The message containing a product name with the estimated price
message ProductPrice {
    string ProductName= 1;
    float EstimatedPrice= 2;
}
```

4.3.9 Production Factors Markets Module (D9)

```
// The production factors price predictor service
service FactorsMarketProduct{
    // Send a request for the production factors markets module to estimated
the prod. factor price.
    rpc CalculateMarketPrice (AgentsProduction) returns (EstimatedMarketPrice)
{}
    rpc CalculateMarketPrice (AgentsDedicatedLand) returns
(EstimatedMarketPrice) {}

// The request message containing a list of produced quantities from which
required factors can be extrapolated.
message AgentsProduction{
    repeated ProductQuantity productQuantity = 1;
}

// The message containing the list of produced amounts.
message ProductQuantity {
    string ProductName= 1;
    float Quantity= 2;
}

// The request message containing a list of input factors amounts.
message AgentsDedicatedLand{
    repeated LandQuantity landQuantity = 1;
}

// The message containan individual pair input name-input quantity.
message LandQuantity {
    string ProductName= 1;
    float landha= 2;
}

// The message containing a product name with the estimated price
message EstimatedMarketPrice{
    float EstimatedFertilizerPrice= 1;
    float EstimatedRentMachinePrice= 2;
}
```

4.3.10 Environmental/Climate IAM (D10)

```
// The environmental impact assessment service
service EnvironmentalAssesser{
    // Send a request for the IAM to compute the global effect of the
aggregated agronomic actions of the agents.
    rpc CalculateAgriculturalEffect (AgentsInfluence) returns
(EnvironmentalImpact) {}

// The request message containing the list of aggregated values derived from
the sum of agronomic management actions by all simulated agents.
message AgentsInfluence{
    repeated PropertyValue propValue= 1;
}

// The message containing a pair element-value (i.e. fertilizer used in
Andalusia - 10e6 Tonnes)
message PropertyValue {
    string PropertyName= 1;
    float Value= 2;
}

// The message containing the list of predefined KPIs calculated using the
internal models of the IAM.
message EnvironmentalImpact{
    float estimatedGHGEmissionsVariation= 1;
    float estimatedWaterPollutionVariation= 2;
}
```

4.3.11 Socio-economic IAM (D11)

```
// The socio-economic impact assessment service
service SocioEconomicAssesser{
    // Send a request for the IAM to compute the global effect of the
aggregated agronomic actions of the agents.
    rpc CalculateAgriculturalEffect (AgentsInfluence) returns
(SocioEconomicImpact) {}

// The request message containing the list of aggregated values derived from
the sum of agronomic management actions by all simulated agents.
message AgentsInfluence{
    repeated PropertyValue propValue= 1;
}

// The message containing a pair element-value (i.e. Area of Olive Groves in
Andalusia in season 2019 - 5e6 ha ).
message PropertyValue {
    string PropertyName= 1;
    float Value= 2;
}

// The message containing the list of predefined KPIs calculated using the
internal models of the IAM.
message SocioEconomicImpact{
    float estimatedAWUVariation= 1; // influence in labour (Agricultural
Working Units)
    float estimatedProfitabilityVariation= 2;
}
```

4.3.12 Ecosystem Services IAM (D12)

```
// The ecosystem services impact assessment service
service EcosystemServicesAssesser{
    // Send a request for the IAM to compute the global effect of the
aggregated agronomic actions of the agents.
    rpc CalculateAgriculturalEffect (AgentsInfluence) returns
(EcosystemServicesImpact) {}

// The request message containing the list of aggregated values derived from
the sum of agronomic management actions by all simulated agents.
message AgentsInfluence{
    repeated PropertyValue propValue= 1;
}

// The message containing a pair element-value (i.e. Area of vineyards in
Andalusia in season 2019 - 5e6 ha ).
message PropertyValue {
    string PropertyName= 1;
    float Value= 2;
}

// The message containing the list of predefined KPIs calculated using the
internal models of the IAM.
message EcosystemServicesImpact{
    float estimatedTouristicIncomeVariation = 1; // influence in the number of
enologic tourists, for example
    float estimatedVariationOfWildlife = 2;
}
```

4.3.13 Climate Models Connectors (D13)

```
// The weather trajectory predictor
service WeatherForecaster{
    // Send a request for the climate model to generate a probable
trajectory(or trajectories) of the weather that (will) affect a particular
region or specific plot ubication.
    rpc PredictWeather (PlotContext) returns (ForecastWeather) {}

// The request message containing the location of the baricenter of the land
plot, and the base year for which the yearly weather trajectory wants to be
generated.
message PlotContext{
    int32 year= 1;
    Position location= 2;
}

// The message containing the location of the land plot.
message Position{
    float x= 1;
    float y= 2;
    float z= 3;
}

// The response message containing the forecasted weather trajectory compound
by a list of monthly averages.
message ForecastWeather{
    repeated WeatherForecast= 1;
}

// The message containing a single monthly forecast, made of average
Temperature, Rainfall and WindSpeed
message WeatherForecast{
    float avgMonthlyTemperature= 1;
    float avgMonthlyRainFall= 2;
    float avgMonthlyWindSpeed= 3;
}
```

4.3.14 Policy Environment Module (D14)

```
// The service that incorporates selected policies into the simulation
service PolicyIncorporator{
    // Send a request for the product markets module to estimated the product
price.
    rpc UpdateOF(PoliciesList) returns (ModifiedOF) {}

// The request message containing a list of policies whose impact want to be
simulated.
message PoliciesList{
    repeated string policies = 1;
}

// The request message containing a list .
message ModifiedOF{
    string of = 1;
}
```

4.3.15 Biophysical Models Connectors (D15)

```
// The soil evolution service
service SoilUpdater{
    // Send a request for the biophysical model to update the state of the soil
of a land plot.
    rpc UpdateSoil (SoilContext) returns (EvolvedSoilState) {}
    rpc UpdateSoil (SoilState) returns (EvolvedSoilState) {}

// The request message containing the soil identifier and characteristics, and
the agronomic management actions and/or (a)biotic disturbances that affected
it.
message SoilState{
    repeated SoilAttributteValue = 1;
    repeated AGactions = 2;
}

// The request message containing the soil identifier and characteristics, and
the agronomic management actions and/or (a)biotic disturbances that affected
it.
message SoilContext{
    int32 soilType = 1;
    repeated AGactions = 2;
}

// The message containing the list of Agronomic actions.
message AGactions{
    AGOperationType action = 1;
    int32 actionModifier = 2;
}

// Enum of allowed agronomic actions
enum AGOperationType {
    FERTILISATION = 0;
    IRRIGATION = 1;
    SANISATION = 2;
    CROPPING = 3;
}

// The response message containing the evolved soil state after the execution
of the call to the Biophysical Model.
message EvolvedSoilState{
    repeated SoilAttributteValue = 1;
}

// The message containing the list of Agronomic actions.
message SoilAttributteValue {
    string AttributeName = 1;
    string Value= 2;
}
```

4.3.16 AGRICORE Interface Module (D16)

The definition and functionalities of the different user-interfaces that allow human users to introduce data, configure and launch simulations, and visualise results will be covered in the dedicated deliverable 'D4.3 - Validated design for the AGRICORE interface'.

5 Development Monitoring

5.1 Open Source Approach

The execution of the software development in the AGRICORE project is coordinated so as to release all the implementations as open-source projects. The selection of this paradigm implies the need to homogenise the tools and repositories to be used, as well as to facilitate the work to the individual developments by taking care of the common preparations.

To that end, the following actions have been taken and/or are currently ongoing:

- Selection of GitLab as the open-source development platform for publishing the projects: The choice was made to use GitLab. GitLab is an open core company that develops software for the software development lifecycle. It provides an end-to-end open-source software development platform with built-in version control, issue tracking, code review and more. GitLab has different plans depending on the team and/or organisation that will use it. It has been possible to have the Gold package, which GitLab provides free of charge to open source projects such as AGRICORE, allowing them to use all the features they have in their platform. The GitLab Gold package covers all the needs of the project while ensuring that the project makes responsible use of resources and budget.
- Preparation of repository, management of permissions and access control.
- Selection of open-source licenses for publishing AGRICORE tools: This selection is being carried out as part of the T8.1 task by the different partners involved in the development of one or several custom modules. To this end, AXIA's guidance has been provided at the various IPR Workshops, but especially at the second one, which was specifically dedicated to the protection of software intellectual property.
- Supervision of license compatibility with reused and integrated libraries.
- Coordination with potential external contributors: This task will be carried out in later phases of the project, once the toolkit is sufficiently mature to allow the incorporation of external developments into the consortium.
- Definition of procedures for repository maintenance after AGRICORE completion.

5.2 Usage of GitLab

As mentioned before, GitLab was the chosen tool both to monitor the project technical advances so as to update the architecture and the associated documentation. Once the general AGRICORE repository was created, as many private projects were initialised as there are modules in the suite. Additionally, public repositories were created to release, after testing, all the developments as they are completed.

To manage all these projects jointly, an AGRICORE Group was also created in GitLab. The groups allow the joint management of permissions for all the projects/repositories included in the global repository. They also allow the creation and visualisation of all Epics, issues and merge requests, as well as collaborative management through communication between all members of the group at the same time. Finally, it is also possible to consult the analytics that show the activity of the group.

Through the epics and issues definition functionalities, requirements have been defined and associated with AGRICORE's public GitLab group. According to GitLab, an EPIC is a high-level technical task that can include other EPICs, as well as low-level technical tasks, which are called

issues.
- High-level requirements are registered via epics. With these epics, it would be possible to know the different functionalities required by the different modules of the AGRICORE group, to follow all related activities and to discuss them.

- Low-level requirements are also defined as epics, but they must be descendants of another epic (parent epic). In this way, the relationship between requirements of the same topic can be graphically observed. A low-level requirement can only have one parent.

- Technical tasks, also known as issues, are the set of implementation tasks (one or more) assigned once a requirement has been analysed, defined and registered, in order to complete the epic associated with that requirement.

6 Optimisation of the Agent-Based Simulation Approach

6.1 Current Applications of Parallel (HPC and/or GPU-based) Simulation of Agent-Based Systems

Central Processing Unit (CPU) is an essential hardware (HW) device in any computer as it can be considered the brain. It is composed of several cores optimised in the resolution of sequential tasks. Although each of these cores is heavyweight, with high clock speeds and latency-optimised, they are designed to run only one task at a time [16].

On the other hand, Graphics Processing Unit (GPU) is an optional HW device that was initially aimed to accelerate 3D graphics [17]. As opposed to a CPU, it is composed of thousands of lightweight cores optimised for dimensional matrix arithmetic and floating-point calculations [18].

Tasks that rely on linear algebra are well suited for GPUs and can be easily parallelised. This is the reason why GPUs can be used to solve artificial intelligence (AI) problems [19]. In fact, Nvidia also released recently RAPIDS, a suite of open-source libraries that integrates with data science libraries to speed up machine learning (ML) [20].

Piętak et al. [21] analysed the benefits of a hybrid system composed of a CPU (4 cores) and a GPU (1280 cores) for an evolutionary multi-agent system (EMAS). The objective is to study the evolution process of a population via reproduction and death operations. In the classic approach, each of these operations is computed by a single CPU. The new approach introduced, however, delegates the operations of evaluation and improvement, as well as the most expensive parts of the computations to the GPU (see Figure 12, extracted from [21]).

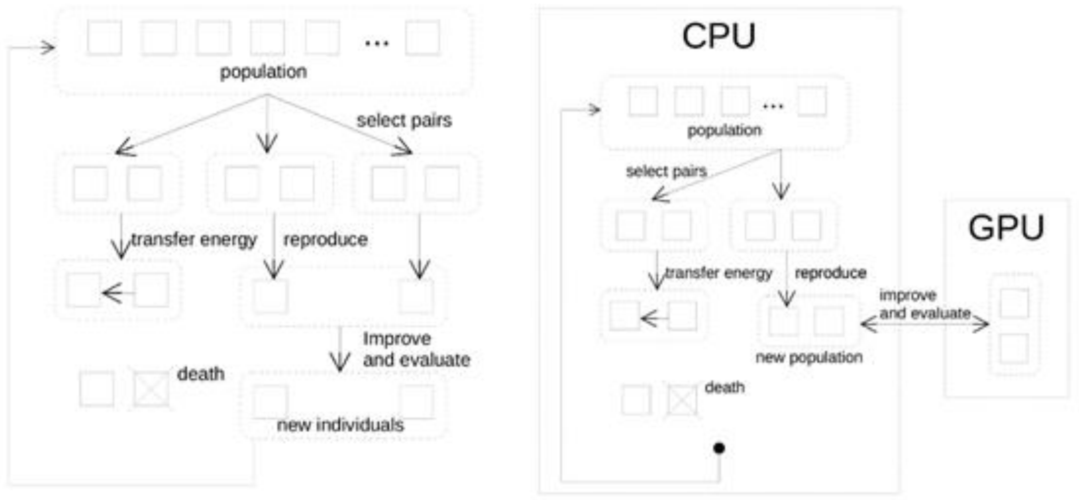


Figure 12 Memetic EMAS algorithm with classic (left) and hybrid (right) architecture.

This problem was addressed as a complex discrete system and solved using the low auto-correlation binary sequences (LABS). The results show that, although the number of steps is much higher, so is the speed-up obtained. Therefore, the hybrid architecture can generate solutions faster, and these turn out to be also of higher quality.

Another biological system was addressed by Konur et al. [22], where they simulated a membrane system, which aims to mimic the functioning and structure of biological cells and their computational counterparts, i.e. membranes. The platform used to perform the simulations of bacterial colonies is called FLAME (Flexible Large scale Agent-based Modelling Environment). This platform can simulate multi-agent models on parallel hardware architectures, including both HPCs and GPUs.

The results shown in Figure 13 (extracted from [22]) compare the performance of FLAME when using it with a pure CPU (only 4 cores) and a GPU (2688 cores). The massive difference in the number of cores has an essential impact on the time needed for the simulation. For populations of cells (number of agents) up to 1000, the GPU simulation time is practically constant and lower than 0.2 seconds, whereas, on a CPU, it increases from 1 to 200 seconds approximately. These times are relatively stable because the number of agents has not surpassed the GPU cores yet, so they all can be processed in parallel.

Once the number of agents is above 2000, the simulation times begin to increase due to full occupancy of the GPU, and some of its cores have to process multiple workloads. It is worth noticing that CPU FLAME cannot simulate a population of 100000 cells; meanwhile, GPU FLAME can solve in under 7 seconds. These significant differences in simulation times denote the improvement introduced by a GPU.

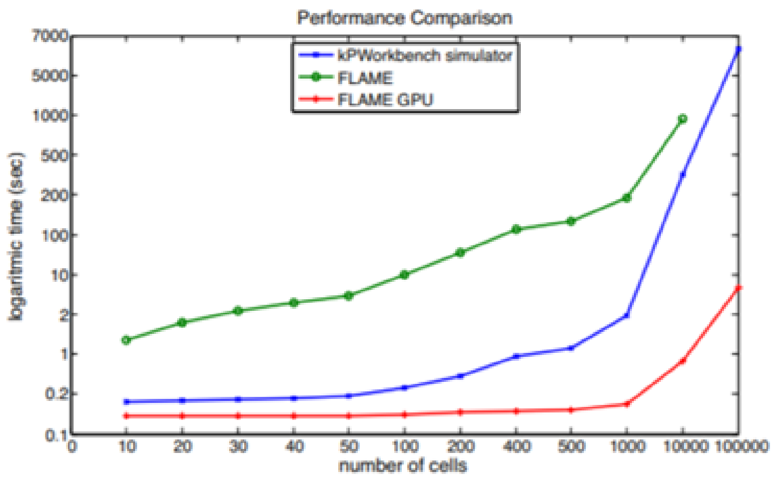


Figure 13 Simulation time using FLAME for bacterial colonies.

Nevertheless, biological systems are not the only systems modelled by agent-based models in the literature. In [23], Saprykin et al. proposed a data-driven approach to run large-scale agent-based traffic simulations on heterogeneous hardware. To this end, the authors developed and used GEMSim (GPU-Enhanced Mobility Simulator), a GPU-accelerated mobility simulator for large-scale and multi-modal agent-based scenarios. In another work [24], GEMSim was compared with MATSim (Multi-Agent Transport Simulation) [25], one of the most advanced existing agent-based simulators for transportation, and highlighted the increase in speed that their software tool can provide.

GEMSim is written in C++ and uses NVIDIA CUDA [26] to accelerate traffic simulations on GPU. The operations required are run by hundreds to thousands of GPU threads grouped in warps (32 threads together). Warps are executed in a Single Instruction Multiple Data (SIMD) manner to obtain parallelism. When using CPU hardware, each core processes a chunk of items instead of a single one as done by a GPU thread.

The results obtained for a scenario composed of 5.5 million agents (which can be classified into two types) are quite intriguing. On the one hand, the scalability of the CPU backend (with an Intel Xeon Gold 6150) depending on the number of cores and task size was evaluated (Figure 14 left, extracted from [24]). The results show that the higher number of cores, the higher performance can be achieved. However, using more than 20 cores only yields marginal improvements.

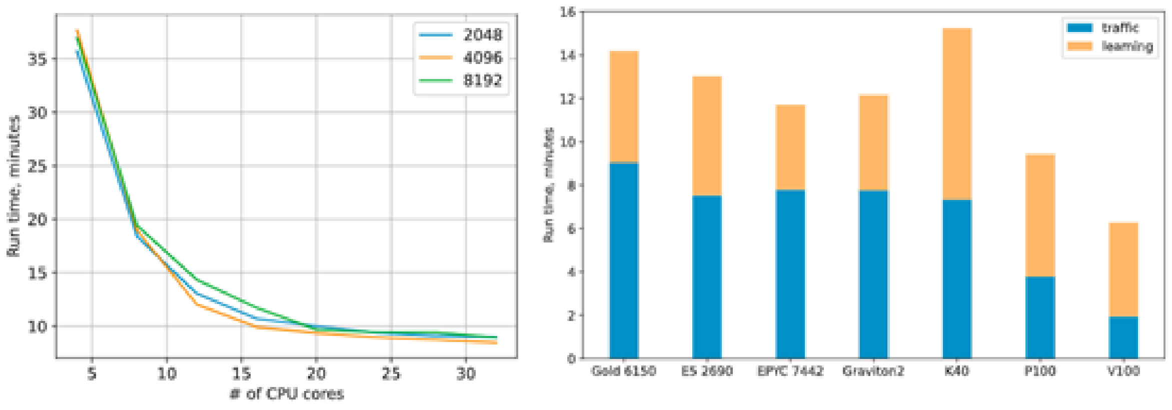


Figure 14 Runtime of CPU backend by number of cores (left) and comparison of CPU/GPU hardware (right).

On the other hand, a comparison of the run time for different hardware configurations is presented (Figure 14 right, Ibid.). Note that the labels "K40", "P100", and "V100" are the only ones that also account for a GPU (see the publication for further details). Several conclusions can be extracted from such a comparison:

- "K40" is the configuration that performs the worst given its longevity (released in 2013).
- Almost all modern CPUs have similar performance when it comes to the traffic propagation part.
- New generations of CPUs result in only marginal performance improvements.
- Each new generation of GPU doubles the run time performance of the traffic propagation model.
- For the CPU and CPU/GPU that achieved the best performance, the latter runs 3.89 times faster for the traffic model and 1.87 times for the complete iteration.

Another technique to increase the level of parallelism consists of using High-Performance Computing (HPC) [27]. This concept is quite simple as it entails connecting several nodes into a cluster. Each of these nodes is composed, in turn, of several CPUs. Therefore, by aggregating more and more processors, an increase in performance can be achieved for computationally intensive tasks. Notice that any desktop CPU presents a minimum of 4 cores nowadays, and high-end CPUs can have more than 16 cores.

Guo et al. [28] carried out a simple experiment to assess the performance impact of HPC when using Parallel Agent-Based Simulations (PABS). The system in question is the space target system, which contains different celestial bodies that move according to orbital kinetics rules. The implementation was done in Repast HPC [29], a toolkit designed explicitly for PABS, which is programmed in C++.

Parallelism was achieved by distributing the different agents into the available cores according to a rank. Such a rank allows identifying the branch of the program to execute depending on the agent. The results from Figure 15 (extracted from [28]) show that an increase in the number of nodes entails a reduction of the simulation time. However, the speed-up that can be obtained is reduced exponentially, aligning with the same trend previously presented in [23] for a single processor.

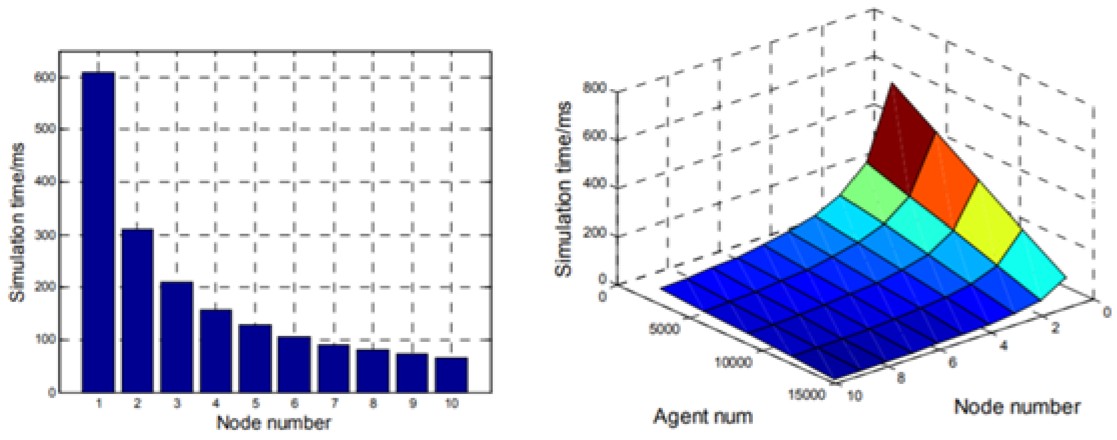


Figure 15 Simulation times for fixed (left) and variable (right) agents number.

Agent-based models have also been utilised to simulate emergency mass evacuations like those caused due to earthquakes or tsunamis. Wijerathne et al. [30] proposed an ABM with a high-resolution model of the environment to account for congestion, pedestrian-vehicle interaction, or fallen debris from damaged buildings. The agents present different behaviours depending on their condition (e.g. evacuee, volunteer, officials, etc.) and the number of such rises to 10 million.

In order to process such a computationally expensive problem, an HPC extension was developed. The hardware utilised is a K computer that can contain 2048 computing nodes or, in other words, 16384 CPU cores. Each of these cores is assigned a nearly equal workload, and parallelisation is achieved via OpenMP's tasks [31]. The following table encompasses the results obtained.

Number of nodes	Execution time (s)	Strong scalability (%)
512	3904.0	-
1024	2067.7	94.4
2048	1258.5	82.1

Table 1 Runtime and strong scalability with 512, 1024 and 2048 nodes in K computer.

As it can be observed, the execution time is reduced as more nodes are involved in the computation. However, such a reduction does not present a linear trend but exponential, which again aligns with the previous results. Ideally, doubling the number of cores would entail half the execution time. It is also denoted that the aforementioned results could have a margin of improvement because the system only uses parallelism for the execution of agents. Other operations such as migration and dynamic load balancing are executed on single threads.

It is undeniable that including mechanisms of parallelism such as GPU or HPC improve the performance of agent-based systems. However, when doing so, it is of capital importance to find the point from which the cost of adding more hardware does not translate into a significant performance improvement to avoid wasting resources. The addition of a GPU is suitable when the hardware is already available, particularly if the problem can be described with linear algebra equations. Furthermore, a toolkit to achieve parallelism is provided directly by the manufacturer. The inclusion of more CPUs into a cluster is generally more expensive, especially if high-end processors are considered.

6.2 Current Suitability for the AGRICORE Approach

From the literature review presented in the previous section, it can be concluded that there are multiple methods to achieve parallelism. Rather than straightforwardly applying the same implementation, each author analysed their problem and presented a solution according to their needs. Therefore, this section will study the different implementations for the particular case of the AGRICORE project to determine whether they are suitable or not.

Piętak et al. [21] decided to use a hybrid architecture composed of both a CPU and a GPU. The GPU was responsible for a series of operations that the agents had to perform, which were the most computationally expensive. However, this case is relatively simple as there is only one type of agent, and each of them needs to perform the same operations.

The approach taken by Konur et al. [22] is quite similar. They made use of a GPU in their agent-based system in which each of the cores represented one agent of the population. The same core performed all the operations, and the bottleneck takes place once the number of agents exceeds the GPU cores available.

The previous implementations are particularly suitable when there is low complexity, i.e. there is only one kind of agent, and they need to perform the same operations. However, should the complexity of the problem get increased either by different types of agents, their operations or a combination of both, the aforementioned solutions are no longer viable. New approaches are needed to distribute the computational load into the multiple cores available (from GPU or HPC).

There are several methods to address agent-based systems that account for more than one type of agent. In [24], the parallel operations required were run by hundreds or thousands of GPU threads, depending on their complexity. Such threads were joined together in groups of 32 to form the so-called warps. This approach can only be given when the hardware available includes a GPU.

In the case of HPCs, two similar solutions were found to solve the complexity increase of the system. The first of them consists of assigning one specific rank to every agent. Such a rank defines the operations the agent needs to perform, and each core can run only agents belonging to a single rank [28]. On the other hand, Wijerathne et al. [30] developed a slightly different approach. Instead of assigning a type of agent per core, they computed the execution time of each one of them. Each core is assigned a nearly equal workload, which is intrinsically related to the aforementioned execution time.

Attending now to the AGRICORE project, each agricultural producer can be considered a separate agent. However, they can be grouped up depending on the farming in which they work. For instance, all olive producers can be regarded as individual agents, but they need to carry out the same operations to farm olives. This can also be expanded if stock producers were to be considered.

As it can be observed, the first two approaches discussed in the current section cannot be applied to the AGRICORE project due to the number of agent types. Furthermore, the implementation is highly dependent on the architecture to be used, i.e. GPU or HPC. It is essential to highlight that the integration of a GPU is generally more straightforward and less expensive than using high-end processors with multiple cores.

Should a GPU be used, the approach taken in [24] can also be applied for the AGRICORE project. Groups of warps are responsible for performing the operations of each agent. The number of warps and execution time depends on the complexity of such operations. According to the authors, the improvements brought by brand new hardware generations are more significant for GPUs than CPUs.

In case GPUs could not be used for whatever reason, and HPC was preferred, parallelisation benefits can also be achieved. Any of the implementations discussed ([28] and [30]) could also be applied in the AGRICORE project. However, under the premise of seeking optimisation, distributing an equal workload between cores is much preferred.

Should an agricultural agent require fewer operations for his farming than another (as permanent crop farmers compared to livestock farmers), a core might be idle while others are still computing. This might even entail a waste of resources depending on the number of agent types (one core per agent is introduced to the system) if their operations are not computationally expensive. Therefore, should an HPC architecture be used in the AGRICORE project, this last approach would be ideal for optimal use of resources.

7 Conclusions

This document is the first version of the definition of the architecture and interfaces of the AGRICORE suite. As was done in the introduction, it should be stressed that this is a living document, which may change as the different modules that make up the structure of the final tool, which started from scratch at the beginning of the project, are implemented. Nevertheless, in this first half of the project, the consortium has managed to redefine the AGRICORE architecture from the initial 11 modules to the current 16 modules presented in this document. Some of these changes are the consequence of a functional reformulation of some of the modules; others are the result of a deepening of the modularisation of the design; finally, some are imposed by the need to build interfaces to make the project independent of the external modules to which it needs to be connected.

For each module, its main objective and its inputs and outputs have been established. Although each module has a different level of implementation, practically all of them have their implementation and encapsulation strategy defined.

As for the interfaces, the task is even more complicated because the final implementation of the modules that interact with each other is not yet available. However, tentative definitions, coded using Protocol Buffers, of such interfaces as gRPCs have been presented in the document. Also specified in this document are the completed and ongoing activities aimed at ensuring the open-source nature of the AGRICORE project, as well as the procedures implemented, using GitLab, for the collaborative development of the software within the Consortium. Finally, a brief exploratory analysis is included on how the parallelisation and power capabilities offered by GPUs and HPCs can be used to ensure that the future final version of the AGRICORE tool can be up-scaled while maintaining minimum performance and speed requirements.

8 References

[1] ^D. L. Parnas, P. C. Clements, and D. M. Weiss, "The Modular Structure of Complex Systems," IEEE Transactions on Software Engineering, vol. SE-11, no. 3, pp. 259–266, 1985, doi: 10.1109/TSE.1985.232209.

[2] ^W. Britz et al., "A design for a generic and modular bio-economic farm model," Agricultural Systems, vol. 191, p. 103133, 2021, doi: https://doi.org/10.1016/j.agsy.2021.103133.

[3] ^P. Reidsma, S. Janssen, J. Jansen, and Martin K. van Ittersum, "On the development and use of farm models for policy impact assessment in the European Union – A review," Agricultural Systems, vol. 159, pp. 111–125, 2018, doi: https://doi.org/10.1016/j.agsy.2017.10.012.

[4] ^The Open Group Website. [Online]. Available: http://www.opengroup.org/

[5] ^N. Dragoni et al., Microservices: yesterday, today, and tomorrow. 2017.

[6] ^A. Balalaie, A. Heydarnoori, and P. Jamshidi, "Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture," IEEE Software, vol. 33, no. 3, pp. 42–52, 2016, doi: 10.1109/MS.2016.64.

[7] ^DAPR, Introduction to the Distributed Application Runtime. [Online]. Available: https://docs.dapr.io/concepts/overview/

[8] ^H. Sun, W. Ha, P.-L. Teh, and J. Huang, "A Case Study on Implementing Modularity in Software Development," Journal of Computer Information Systems, Jul. 2015, doi: 10.1080/08874417.2016.1183430.

[9] ^K. Sullivan, W. Griswold, Y. Cai, and B. Hallen, "The Structure and Value of Modularity in Software Design," ACM SIGSOFT Software Engineering Notes, vol. 26, Aug. 2001, doi: 10.1145/503271.503224.

[10] ^M. Kaszubowski, The benefits of modular software design. 2020. [Online]. Available: https://mkaszubowski.com/2020/06/02/modular-software-design-benefits.html

[11] ^M. Tsagris, The FEDHC Bayesian network learning algorithm. 2021.

[12] ^The PCHC and PCTABU Bayesian network learning algorithms in pchc. [Online]. Available: https://rdrr.io/cran/pchc/man/pchc.html

[13] ^Electron Build cross-platform desktop apps with JavaScript, HTML, and CSS. [Online]. Available: https://www.electronjs.org/

[14] ^Best Practice Software Engineering - Interface. [Online]. Available: http://best-practice-software-engineering.ifs.tuwien.ac.at/patterns/interface.html

[15] ^L. S. Wheatcraft, "9.2.2 Everything you wanted to know about interfaces, but were afraid to ask," INCOSE International Symposium, vol. 20, no. 1, pp. 1132–1149, Jul. 2010, doi: 10.1002/j.2334-5837.2010.tb01130.x.

[16] ^Levinas, Mantas, GPU vs CPU: What Are The Key Differences? — Cherry Servers. 2020. [Online]. Available: https://blog.cherryservers.com/gpu-vs-cpu-what-are-the-key-differences

[17] ^What is a GPU? — Intel. [Online]. Available: https://www.intel.la/content/www/xl/es/products/docs/processors/what-is-a-gpu.html

[18] ^Caulfield, Brian, What's the difference between a CPU and a GPU? — NVIDIA. 2009. [Online]. Available: https://blogs.nvidia.com/blog/2009/12/16/whats-the-difference-between-a-cpu-and-a-gpu/

[19] ^Shabarok, German, Why deep learning uses GPUs? — Towards Data Science. 2020. [Online]. Available: https://towardsdatascience.com/why-deep-learning-uses-gpus-c61b399e93a0

[20] ^NVIDIA Introduces RAPIDS Open-Source GPU-Acceleration Platform for Large-Scale Data Analytics and Machine Learning — NVIDIA NEWSROOM. 2018. [Online]. Available: https://nvidianews.nvidia.com/news/nvidia-introduces-rapids-open-source-gpu-acceleration-platform-for-large-scale-data-analytics-and-machine-learning

[21] ^1 2 3 K. Piętak, D. Żurek, M. Pietroń, A. Dymara, and M. Kisiel-Dorohinicki, "Striving for performance of discrete optimisation via memetic agent-based systems in a hybrid CPU/GPU environment," Journal of Computational Science, vol. 31, pp. 151–162, 2019, doi: https://doi.org/10.1016/j.jocs.2019.01.007.

[22] ^1 2 3 S. Konur, M. Kiran, M. Gheorghe, M. Burkitt, and F. Ipate, "Agent-Based High-Performance Simulation of Biological Systems on the GPU," in 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems, 2015, pp. 84–89. doi: 10.1109/HPCC-CSS-ICESS.2015.253.

[23] ^1 2 A. Saprykin, N. Chokani, and R. S. Abhari, "A data-driven approach to run agent-based multi-modal traffic simulations on heterogeneous CPU-GPU hardware," Procedia Computer Science, vol. 184, pp. 720–727, 2021, doi: https://doi.org/10.1016/j.procs.2021.04.021.

[24] ^1 2 3 4 A. Saprykin, N. Chokani, and R. S. Abhari, "GEMSim: A GPU-accelerated multi-modal mobility simulator for large-scale scenarios," Simulation Modelling Practice and Theory, vol. 94, pp. 199–214, 2019, doi: https://doi.org/10.1016/j.simpat.2019.03.002.

[25] ^K. W. Axhausen, A. Horni, and K. Nagel, The Multi-Agent Transport Simulation (MATSim). London: Ubiquity Press, 2016.

[26] ^Cuda Toolkit — NVIDIA DEVELOPER. [Online]. Available: https://developer.nvidia.com/cuda-toolkit

[27] ^What is HPC? — Intel. [Online]. Available: https://www.intel.la/content/www/xl/es/high-performance-computing/what-is-hpc.html

[28] ^1 2 3 4 C. Guo and W. Xiong, "Parallel agent-based simulation of complex system based on repast HPC," in 2013 2nd International Symposium on Instrumentation and Measurement, Sensor Network and Automation (IMSNA), 2013, pp. 808–812. doi: 10.1109/IMSNA.2013.6743400.

[29] ^Collier, Nick, Repast HPC Manual. 2013. [Online]. Available: https://repast.github.io/docs/repast_hpc.pdf

[30] ^1 2 3 L. Wijerathne, W. Petprakob, L. Aguilar, M. Hori, and T. Ichmura, "Scalable HPC Enhanced Agent Based System for Simulating Mixed Mode Evacuation of Large Urban Areas," Transportation Research Procedia, vol. 34, pp. 275–282, 2018, doi: https://doi.org/10.1016/j.trpro.2018.11.042.

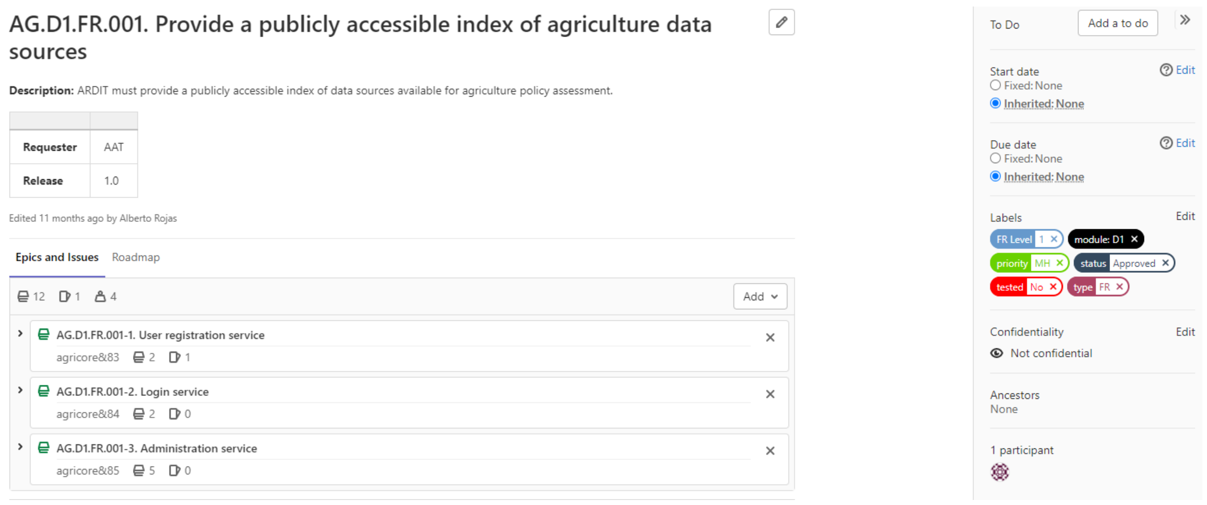
[31] ^OpenMP. [Online]. Available: https://www.openmp.org

For preparing this report, besides the previous references, the following deliverables have been taken into consideration:

Deliverable Number	**Deliverable Title**	**Lead beneficiary**	**Type**	**Dissemination Level**	**Due date**
D4.1	AGRICORE requirements and project management platform	AAT	Report	Public	M12 (August 2020)
D6.6	Software Quality Assurance measures for AGRICORE	AAT	Report	Public	M15 (November 2020)

Annex A: List of Requirements (Gitlab Epics)

Module D1: ARDIT



AG.D1.FR.001. Provide a publicly accessible index of agriculture data sources

Description: ARDIT must provide a publicly accessible index of data sources available for agriculture policy assessment.

Requester	AAT
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

12 1 4 Add

AG.D1.FR.001-1. User registration service
agricore&83 2 1

AG.D1.FR.001-2. Login service
agricore&84 2 0

AG.D1.FR.001-3. Administration service
agricore&85 5 0

To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

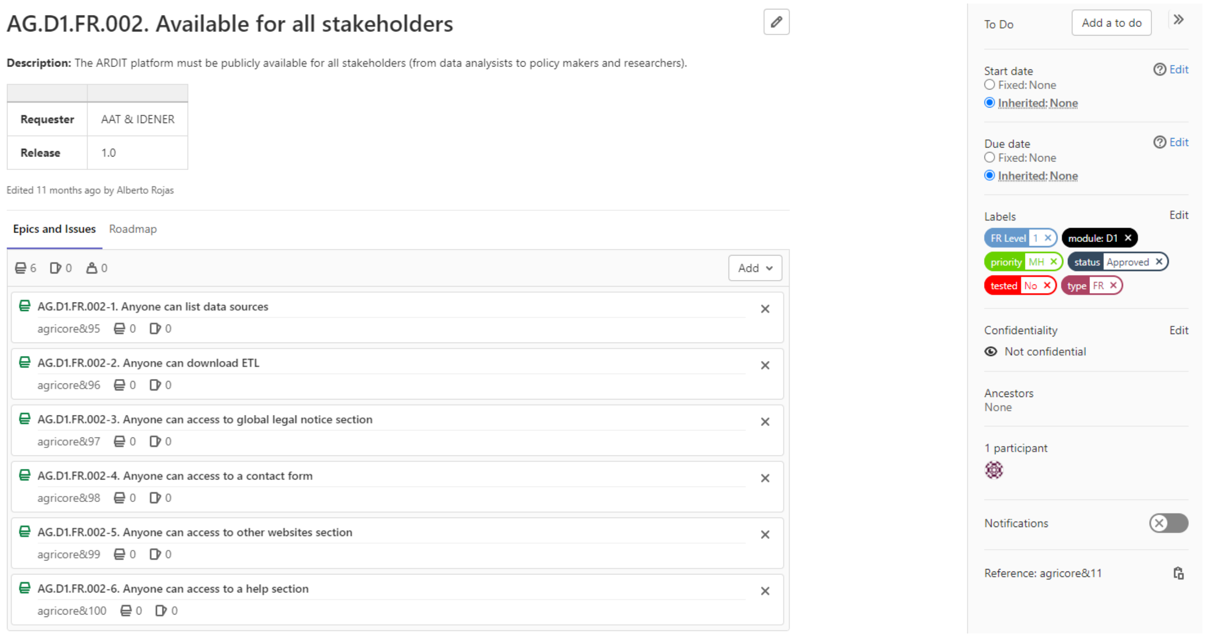
Labels Edit
FR Level 1 | module: D1 | priority MH | status Approved | tested No | type FR

Confidentiality Edit
Not confidential

Ancestors
None

1 participant

Figure 16 AG.D1.FR.001. Provide a publicly accessible index of agriculture data sources



AG.D1.FR.002. Available for all stakeholders

Description: The ARDIT platform must be publicly available for all stakeholders (from data analysists to policy makers and researchers).

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

6 0 0 Add

AG.D1.FR.002-1. Anyone can list data sources
agricore&95 0 0

AG.D1.FR.002-2. Anyone can download ETL
agricore&96 0 0

AG.D1.FR.002-3. Anyone can access to global legal notice section
agricore&97 0 0

AG.D1.FR.002-4. Anyone can access to a contact form
agricore&98 0 0

AG.D1.FR.002-5. Anyone can access to other websites section
agricore&99 0 0

AG.D1.FR.002-6. Anyone can access to a help section
agricore&100 0 0

To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 1 | module: D1 | priority MH | status Approved | tested No | type FR

Confidentiality Edit
Not confidential

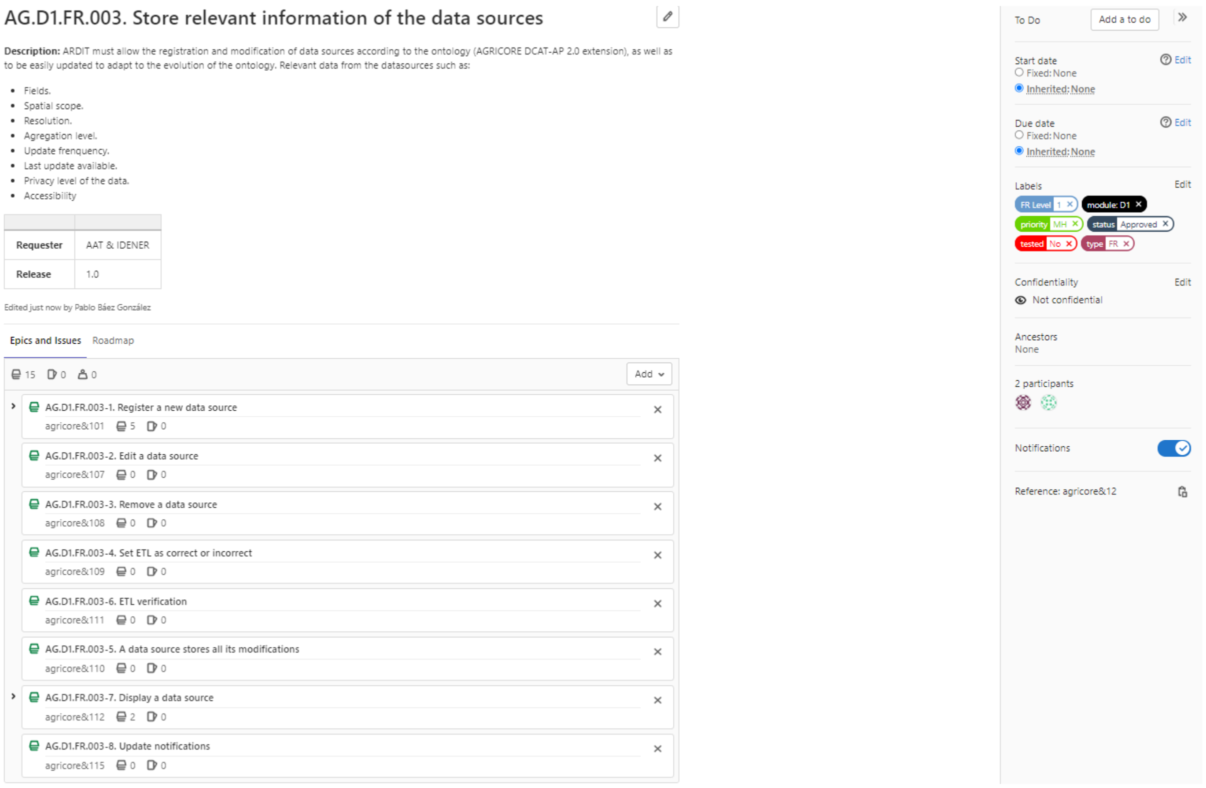
Ancestors
None

1 participant

Notifications

Reference: agricore&11

Figure 17 AG.D1.FR.002. Available for all stakeholders



AG.D1.FR.003. Store relevant information of the data sources

Description: ARDIT must allow the registration and modification of data sources according to the ontology (AGRICORE DCAT-AP 2.0 extension), as well as to be easily updated to adapt to the evolution of the ontology. Relevant data from the datasources such as:

- Fields.
- Spatial scope.
- Resolution.
- Agregation level.
- Update frenquency.
- Last update available.
- Privacy level of the data.
- Accessibility

Requester	AAT & IDENER
Release	1.0

Edited just now by Pablo Báez González

Epics and Issues Roadmap

15 0 0 Add

AG.D1.FR.003-1. Register a new data source
agricore&101 5 0

AG.D1.FR.003-2. Edit a data source
agricore&107 0 0

AG.D1.FR.003-3. Remove a data source
agricore&108 0 0

AG.D1.FR.003-4. Set ETL as correct or incorrect
agricore&109 0 0

AG.D1.FR.003-6. ETL verification
agricore&111 0 0

AG.D1.FR.003-5. A data source stores all its modifications
agricore&110 0 0

AG.D1.FR.003-7. Display a data source
agricore&112 2 0

AG.D1.FR.003-8. Update notifications
agricore&115 0 0

To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 1 | module: D1 | priority MH | status Approved | tested No | type FR

Confidentiality Edit
Not confidential

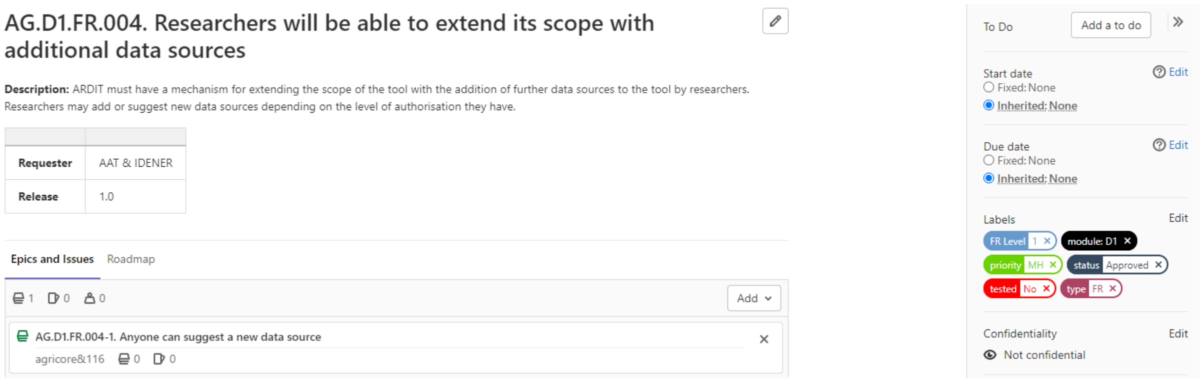
Ancestors
None

2 participants

Notifications

Reference: agricore&12

Figure 18 AG.D1.FR.003. Store relevant information of the data sources



AG.D1.FR.004. Researchers will be able to extend its scope with additional data sources

Description: ARDIT must have a mechanism for extending the scope of the tool with the addition of further data sources to the tool by researchers. Researchers may add or suggest new data sources depending on the level of authorisation they have.

Requester	AAT & IDENER
Release	1.0

Epics and Issues Roadmap

1 0 0 Add

AG.D1.FR.004-1. Anyone can suggest a new data source
agricore&116 0 0

To Do Add a to do

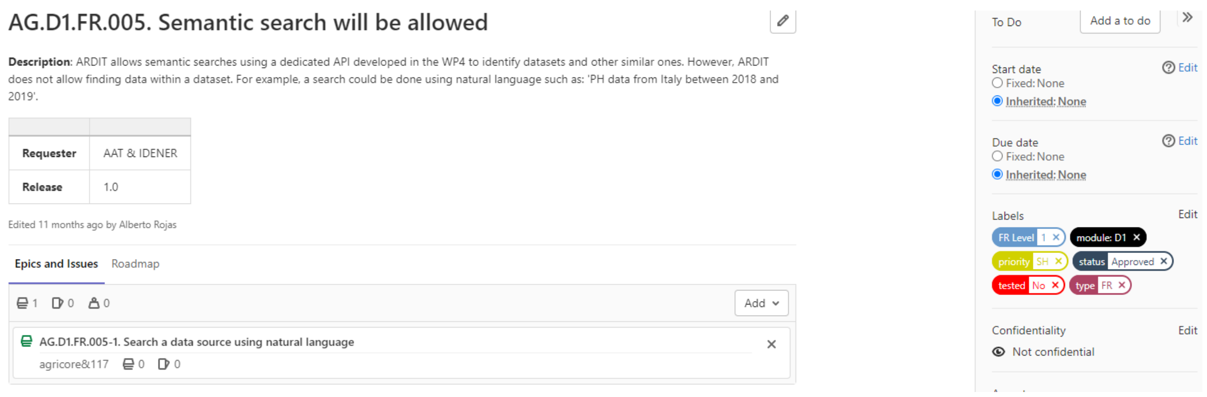
Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 1 | module: D1 | priority MH | status Approved | tested No | type FR

Confidentiality Edit
Not confidential

Figure 19 AG.D1.FR.004. Researchers will be able to extend its scope with additional data sources



AG.D1.FR.005. Semantic search will be allowed

Description: ARDIT allows semantic searches using a dedicated API developed in the WP4 to identify datasets and other similar ones. However, ARDIT does not allow finding data within a dataset. For example, a search could be done using natural language such as: 'PH data from Italy between 2018 and 2019'.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

1 0 0 Add

AG.D1.FR.005-1. Search a data source using natural language
agricore&117 0 0

To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

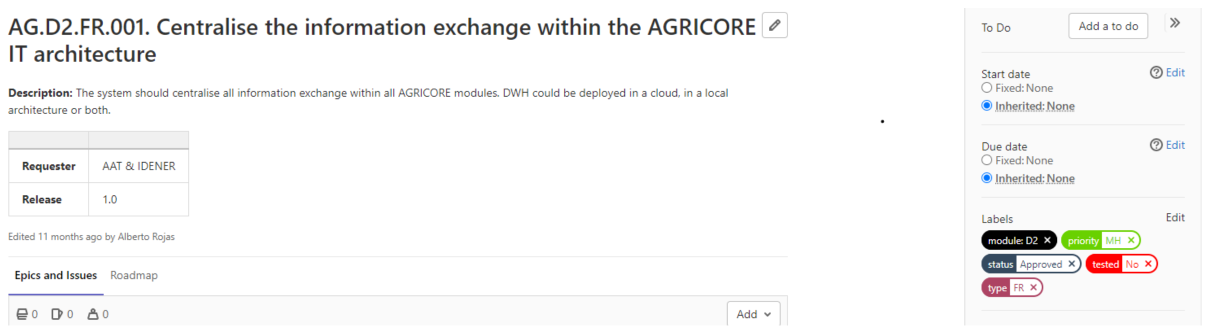
Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 1, module: D1, priority SH, status Approved, tested No, type FR

Confidentiality Edit
Not confidential

Figure 20 AG.D1.FR.005. Semantic search will be allowed

Module D2: DWH



AG.D2.FR.001. Centralise the information exchange within the AGRICORE IT architecture

Description: The system should centralise all information exchange within all AGRICORE modules. DWH could be deployed in a cloud, in a local architecture or both.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

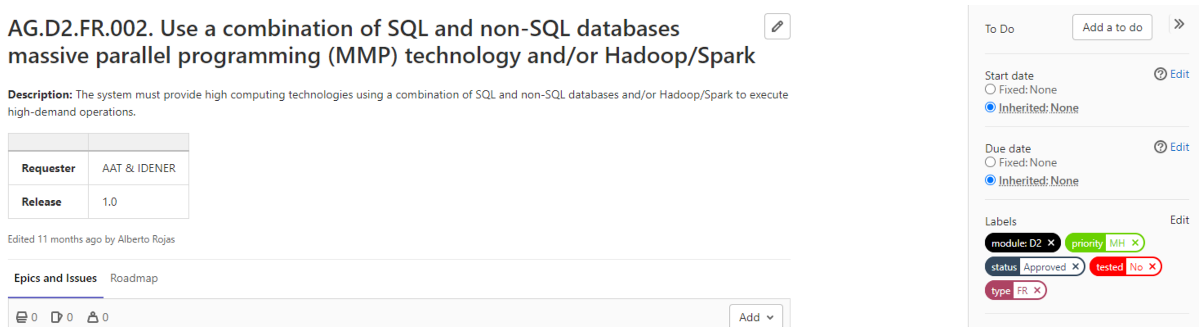
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D2, priority MH, status Approved, tested No, type FR

Figure 21 AG.D2.FR.001. Centralise the information exchange within the AGRICORE IT architecture



AG.D2.FR.002. Use a combination of SQL and non-SQL databases massive parallel programming (MMP) technology and/or Hadoop/Spark

Description: The system must provide high computing technologies using a combination of SQL and non-SQL databases and/or Hadoop/Spark to execute high-demand operations.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

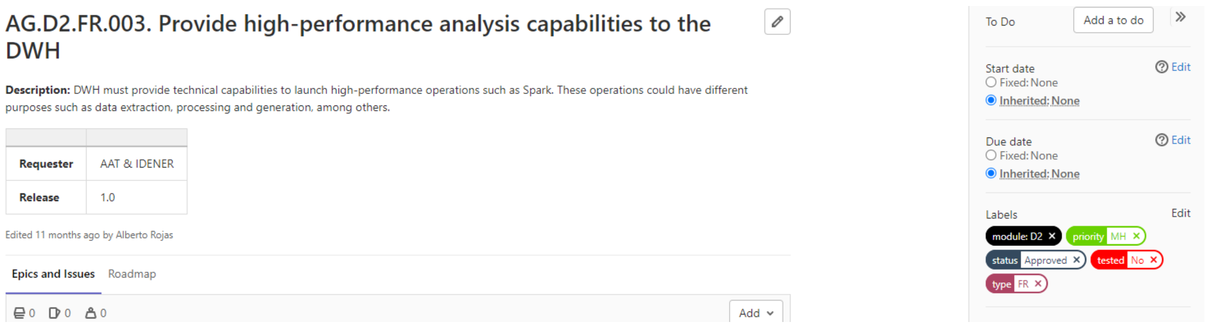
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D2, priority MH, status Approved, tested No, type FR

Figure 22 AG.D2.FR.002. Use a combination of SQL and non-SQL databases massive parallel programming (MMP) technology and/or Hadoop/Spark


AG.D2.FR.003. Provide high-performance analysis capabilities to the DWH

Description: DWH must provide technical capabilities to launch high-performance operations such as Spark. These operations could have different purposes such as data extraction, processing and generation, among others.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

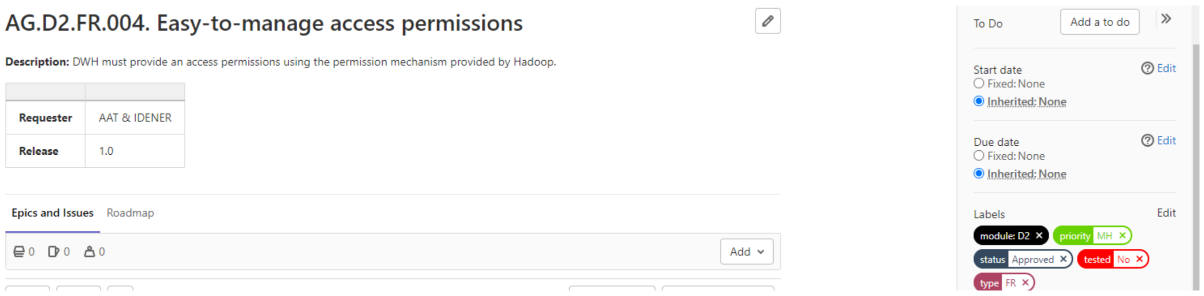
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D2 | priority MH | status Approved | tested No | type FR

Figure 23 AG.D2.FR.003. Provide high-performance analysis capabilities to the DWH


AG.D2.FR.004. Easy-to-manage access permissions

Description: DWH must provide an access permissions using the permission mechanism provided by Hadoop.

Requester	AAT & IDENER
Release	1.0

Epics and Issues Roadmap

0 0 0 Add

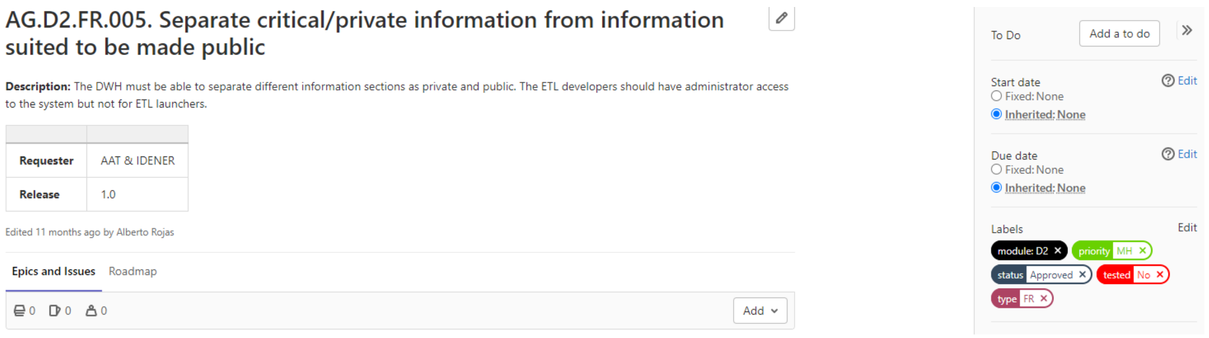
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D2 | priority MH | status Approved | tested No | type FR

Figure 24 AG.D2.FR.004. Easy-to-manage access permissions


AG.D2.FR.005. Separate critical/private information from information suited to be made public

Description: The DWH must be able to separate different information sections as private and public. The ETL developers should have administrator access to the system but not for ETL launchers.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

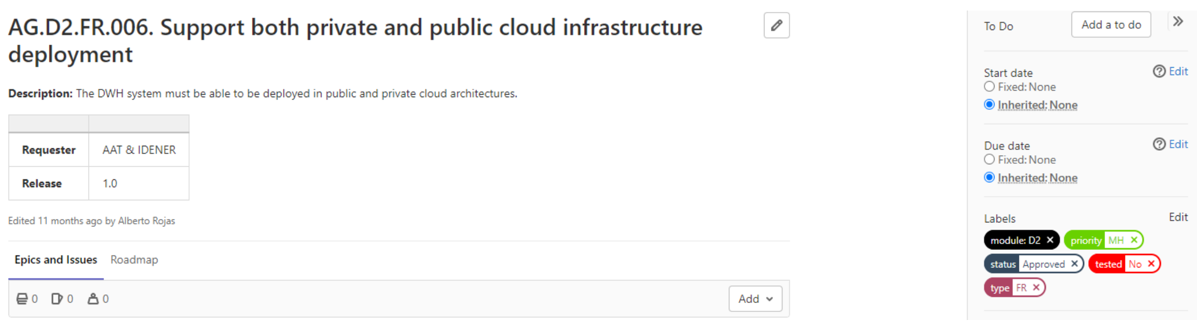
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D2 | priority MH | status Approved | tested No | type FR

Figure 25 AG.D2.FR.005. Separate critical/private information from information suited to be made public



AG.D2.FR.006. Support both private and public cloud infrastructure deployment

Description: The DWH system must be able to be deployed in public and private cloud architectures.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

To Do Add a to do

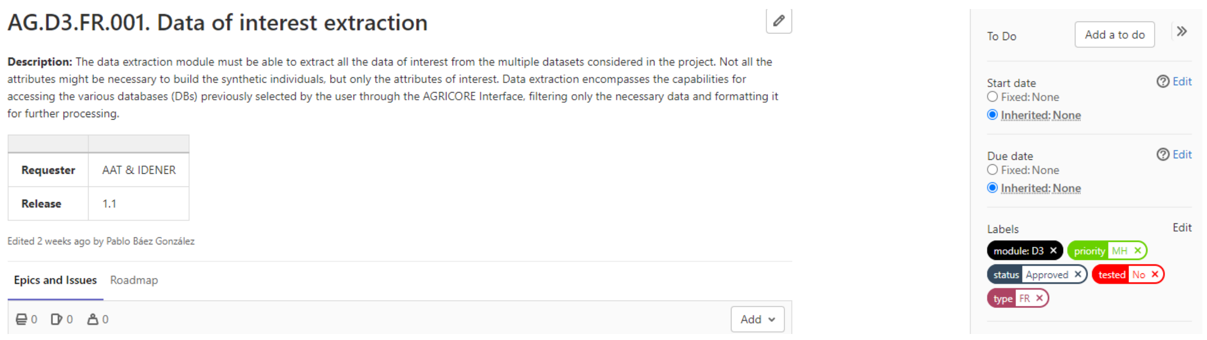
Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D2 | priority MH | status Approved | tested No | type FR

Figure 26 AG.D2.FR.006. Support both private and public cloud infrastructure deployment

Module D3: DEM



AG.D3.FR.001. Data of interest extraction

Description: The data extraction module must be able to extract all the data of interest from the multiple datasets considered in the project. Not all the attributes might be necessary to build the synthetic individuals, but only the attributes of interest. Data extraction encompasses the capabilities for accessing the various databases (DBs) previously selected by the user through the AGRICORE Interface, filtering only the necessary data and formatting it for further processing.

Requester	AAT & IDENER
Release	1.1

Edited 2 weeks ago by Pablo Báez González

Epics and Issues Roadmap

0 0 0 Add

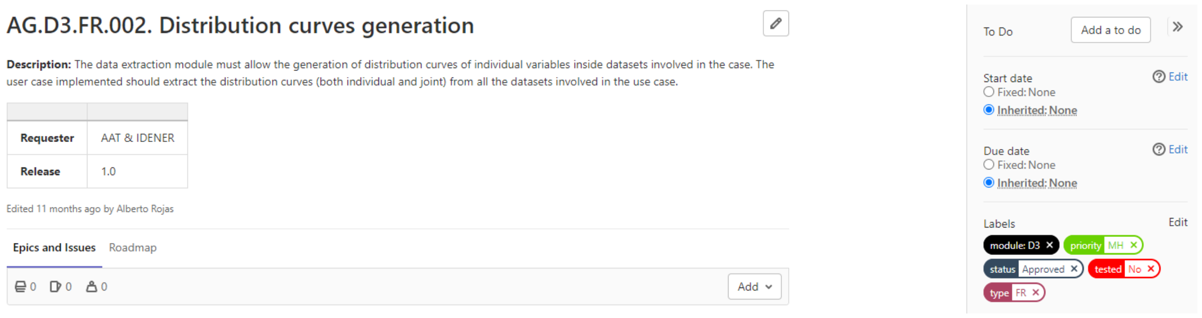
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D3 | priority MH | status Approved | tested No | type FR

Figure 27 AG.D3.FR.001. Data of interest extraction



AG.D3.FR.002. Distribution curves generation

Description: The data extraction module must allow the generation of distribution curves of individual variables inside datasets involved in the case. The user case implemented should extract the distribution curves (both individual and joint) from all the datasets involved in the use case.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

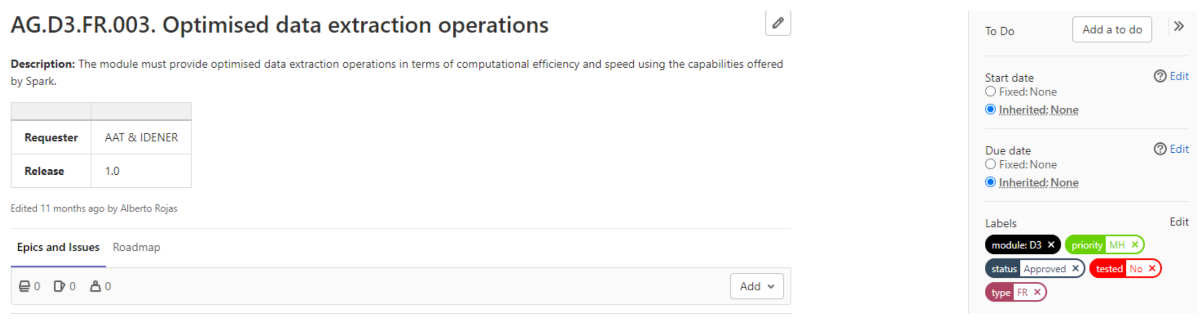
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D3 | priority MH | status Approved | tested No | type FR

Figure 28 AG.D3.FR.002. Distribution curves generation



AG.D3.FR.003. Optimised data extraction operations

Description: The module must provide optimised data extraction operations in terms of computational efficiency and speed using the capabilities offered by Spark.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

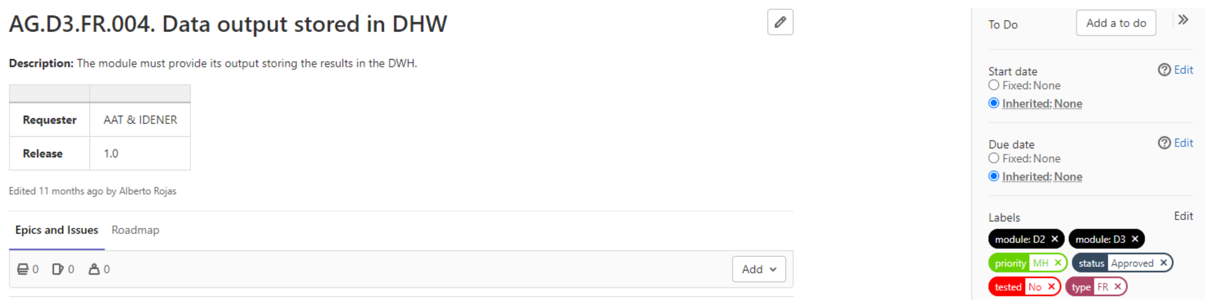
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D3 · priority MH · status Approved · tested No · type FR

Figure 29 AG.D3.FR.003. Optimised data extraction operations



AG.D3.FR.004. Data output stored in DHW

Description: The module must provide its output storing the results in the DWH.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

To Do Add a to do

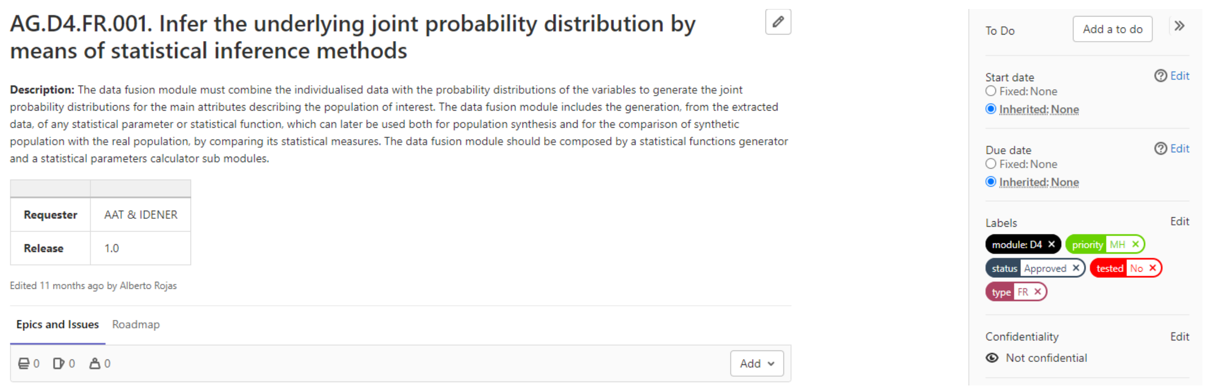
Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D2 · module: D3 · priority MH · status Approved · tested No · type FR

Figure 30 AG.D3.FR.004. Data output stored in DHW

Module D4: DFM



AG.D4.FR.001. Infer the underlying joint probability distribution by means of statistical inference methods

Description: The data fusion module must combine the individualised data with the probability distributions of the variables to generate the joint probability distributions for the main attributes describing the population of interest. The data fusion module includes the generation, from the extracted data, of any statistical parameter or statistical function, which can later be used both for population synthesis and for the comparison of synthetic population with the real population, by comparing its statistical measures. The data fusion module should be composed by a statistical functions generator and a statistical parameters calculator sub modules.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

To Do Add a to do

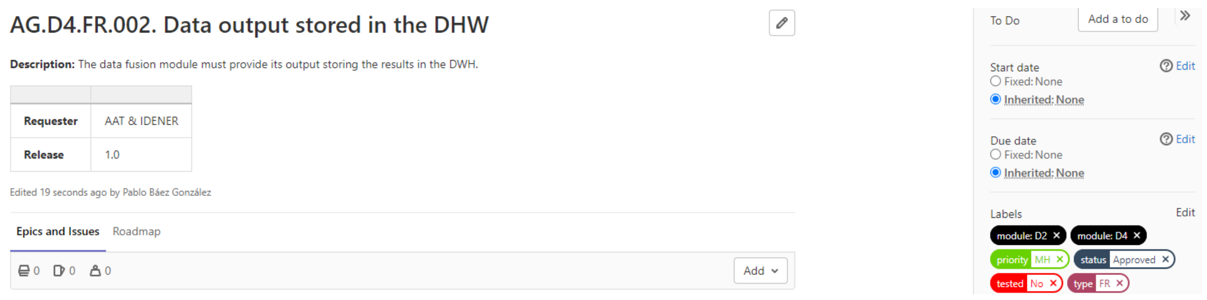
Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D4 · priority MH · status Approved · tested No · type FR

Confidentiality Edit
Not confidential

Figure 31 AG.D4.FR.001. Infer the underlying joint probability distribution by means of statistical inference methods



AG.D4.FR.002. Data output stored in the DHW

Description: The data fusion module must provide its output storing the results in the DWH.

Requester	AAT & IDENER
Release	1.0

Edited 19 seconds ago by Pablo Báez González

Epics and Issues Roadmap

0 0 0 Add

To Do Add a to do

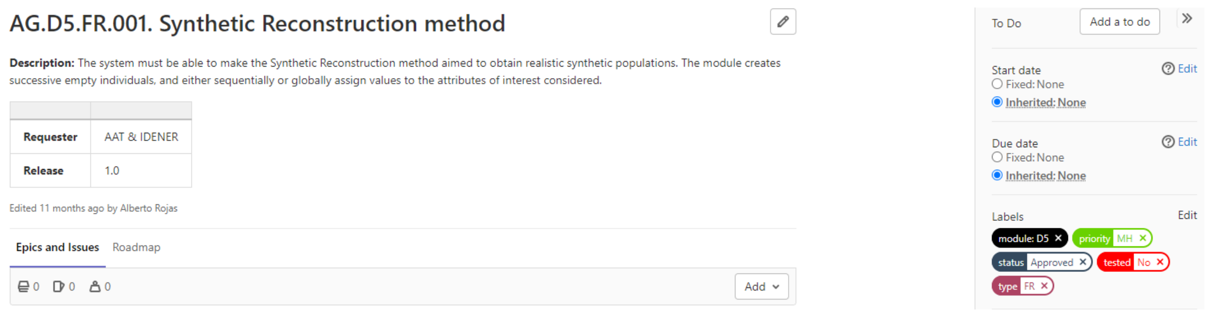
Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D2 module: D4 priority MH status Approved tested No type FR

Figure 32 AG.D4.FR.004. Data output stored in DHW

Module D5: SPG



AG.D5.FR.001. Synthetic Reconstruction method

Description: The system must be able to make the Synthetic Reconstruction method aimed to obtain realistic synthetic populations. The module creates successive empty individuals, and either sequentially or globally assign values to the attributes of interest considered.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

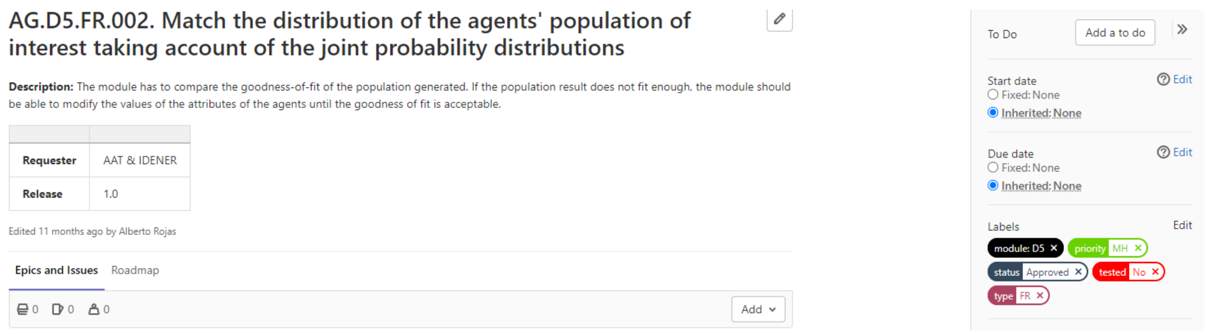
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D5 priority MH status Approved tested No type FR

Figure 33 AG.D5.FR.001. Synthetic Reconstruction method



AG.D5.FR.002. Match the distribution of the agents' population of interest taking account of the joint probability distributions

Description: The module has to compare the goodness-of-fit of the population generated. If the population result does not fit enough, the module should be able to modify the values of the attributes of the agents until the goodness of fit is acceptable.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

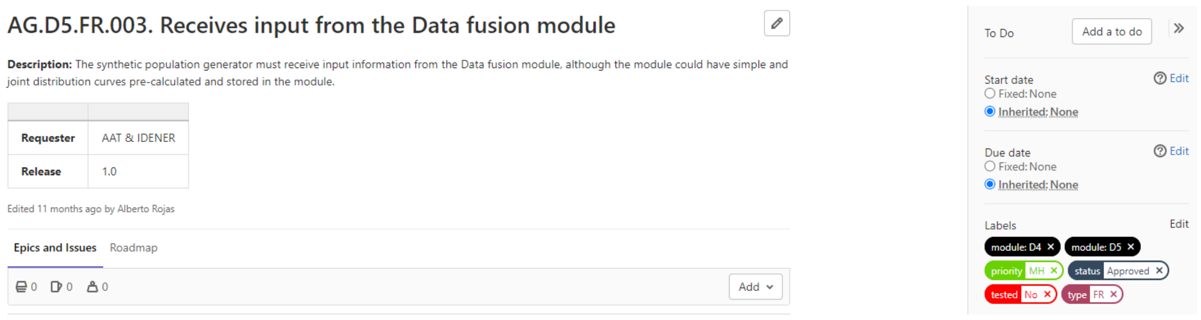
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D5 priority MH status Approved tested No type FR

Figure 34 AG.D5.FR.002. Match the distribution of the agents' population of interest taking account of the joint probability distributions



AG.D5.FR.003. Receives input from the Data fusion module

Description: The synthetic population generator must receive input information from the Data fusion module, although the module could have simple and joint distribution curves pre-calculated and stored in the module.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

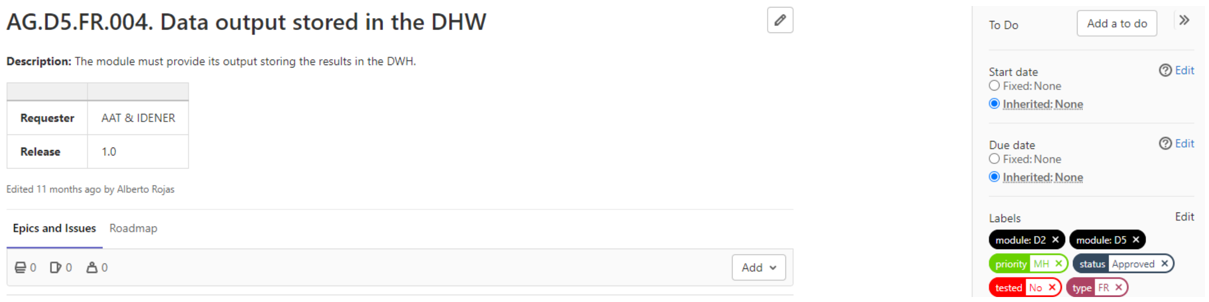
To Do Add a to do

Start date
Fixed: None
Inherited: None
Edit

Due date
Fixed: None
Inherited: None
Edit

Labels Edit
module: D4 module: D5 priority MH status Approved tested No type FR

Figure 35 AG.D5.FR.003. Receives input from the Data fusion module



AG.D5.FR.004. Data output stored in the DHW

Description: The module must provide its output storing the results in the DWH.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

To Do Add a to do

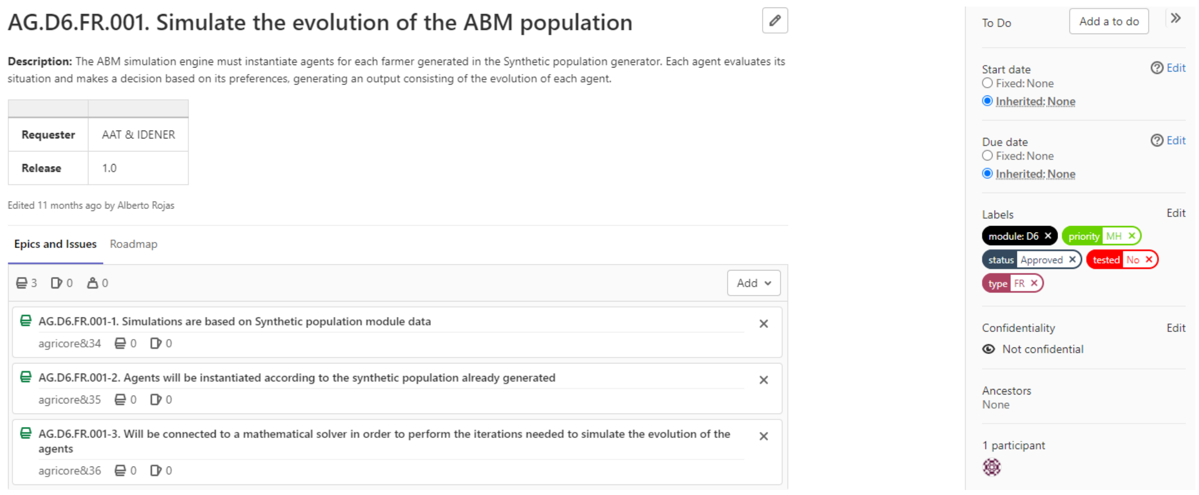
Start date
Fixed: None
Inherited: None
Edit

Due date
Fixed: None
Inherited: None
Edit

Labels Edit
module: D2 module: D5 priority MH status Approved tested No type FR

Figure 36 AG.D5.FR.004. Data output stored in the DHW

Module D6: ABM-e



AG.D6.FR.001. Simulate the evolution of the ABM population

Description: The ABM simulation engine must instantiate agents for each farmer generated in the Synthetic population generator. Each agent evaluates its situation and makes a decision based on its preferences, generating an output consisting of the evolution of each agent.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

3 0 0 Add

AG.D6.FR.001-1. Simulations are based on Synthetic population module data
agricore&34 0 0

AG.D6.FR.001-2. Agents will be instantiated according to the synthetic population already generated
agricore&35 0 0

AG.D6.FR.001-3. Will be connected to a mathematical solver in order to perform the iterations needed to simulate the evolution of the agents
agricore&36 0 0

To Do Add a to do

Start date
Fixed: None
Inherited: None
Edit

Due date
Fixed: None
Inherited: None
Edit

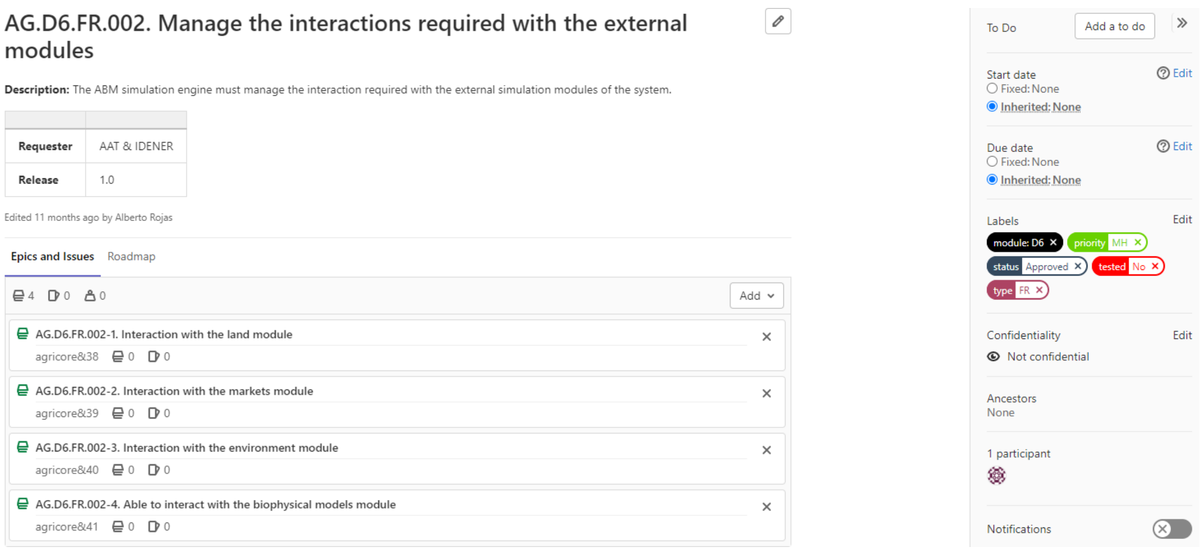
Labels Edit
module: D6 priority MH status Approved tested No type FR

Confidentiality Edit
Not confidential

Ancestors
None

1 participant

Figure 37 AG.D6.FR.001. Simulate the evolution of the ABM population



AG.D6.FR.002. Manage the interactions required with the external modules

Description: The ABM simulation engine must manage the interaction required with the external simulation modules of the system.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

4 0 0 Add

AG.D6.FR.002-1. Interaction with the land module
agricore&38 0 0

AG.D6.FR.002-2. Interaction with the markets module
agricore&39 0 0

AG.D6.FR.002-3. Interaction with the environment module
agricore&40 0 0

AG.D6.FR.002-4. Able to interact with the biophysical models module
agricore&41 0 0

To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D6 priority MH status Approved tested No type FR

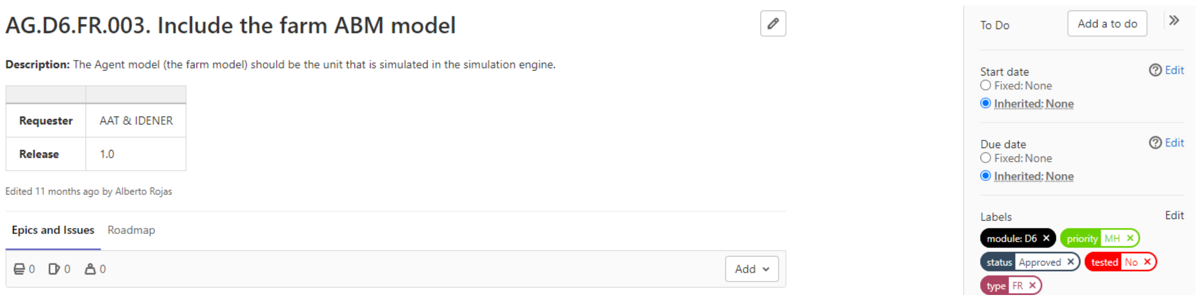
Confidentiality Edit
Not confidential

Ancestors
None

1 participant

Notifications

Figure 38 AG.D6.FR.002. Manage the interactions required with the external modules



AG.D6.FR.003. Include the farm ABM model

Description: The Agent model (the farm model) should be the unit that is simulated in the simulation engine.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

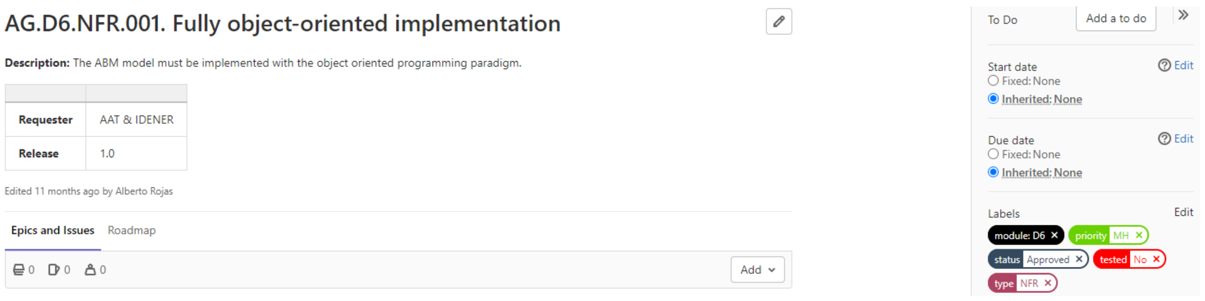
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D6 priority MH status Approved tested No type FR

Figure 39 AG.D6.FR.003. Include the farm ABM model



AG.D6.NFR.001. Fully object-oriented implementation

Description: The ABM model must be implemented with the object oriented programming paradigm.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

0 0 0 Add

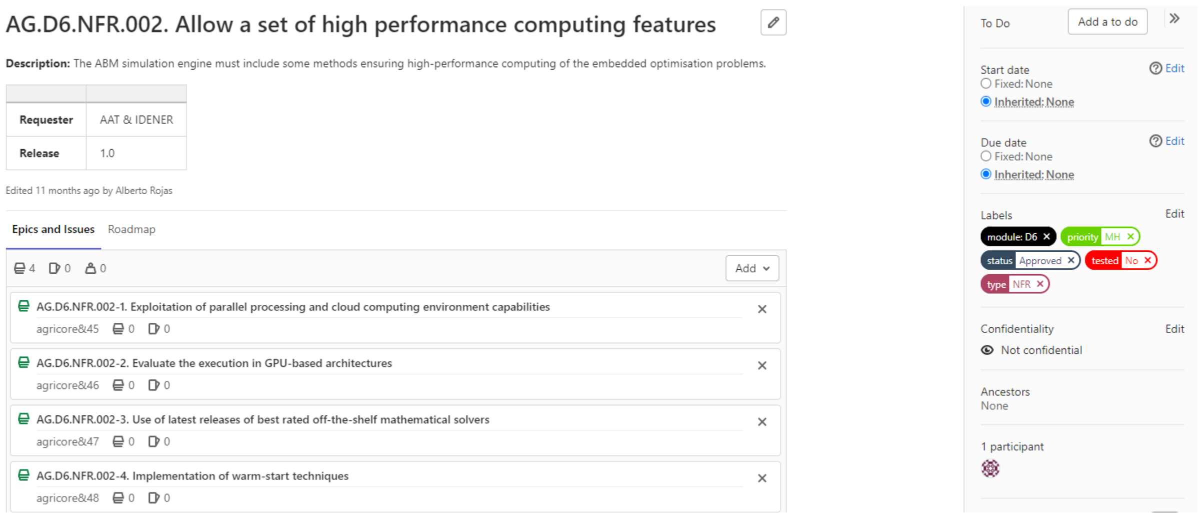
To Do Add a to do

Start date Edit
Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
module: D6 priority MH status Approved tested No type NFR

Figure 40 AG.D6.NFR.001. Fully object-oriented implementation



AG.D6.NFR.002. Allow a set of high performance computing features

Description: The ABM simulation engine must include some methods ensuring high-performance computing of the embedded optimisation problems.

Requester	AAT & IDENER
Release	1.0

Edited 11 months ago by Alberto Rojas

Epics and Issues Roadmap

4 0 0 Add

- AG.D6.NFR.002-1. Exploitation of parallel processing and cloud computing environment capabilities — agricore&45 0 0
- AG.D6.NFR.002-2. Evaluate the execution in GPU-based architectures — agricore&46 0 0
- AG.D6.NFR.002-3. Use of latest releases of best rated off-the-shelf mathematical solvers — agricore&47 0 0
- AG.D6.NFR.002-4. Implementation of warm-start techniques — agricore&48 0 0

To Do — Add a to do

Start date: Fixed: None; Inherited: None

Due date: Fixed: None; Inherited: None

Labels: module: D6; priority MH; status Approved; tested No; type NFR

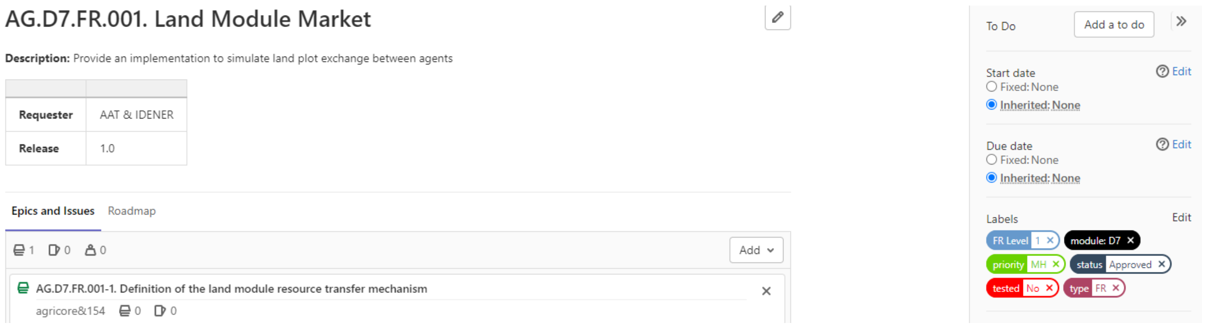
Confidentiality: Not confidential

Ancestors: None

1 participant

Figure 41 AG.D6.NFR.002. Allow a set of high performance computing features

Module D7: Land Market Module



AG.D7.FR.001. Land Module Market

Description: Provide an implementation to simulate land plot exchange between agents

Requester	AAT & IDENER
Release	1.0

Epics and Issues Roadmap

1 0 0 Add

- AG.D7.FR.001-1. Definition of the land module resource transfer mechanism — agricore&154 0 0

To Do — Add a to do

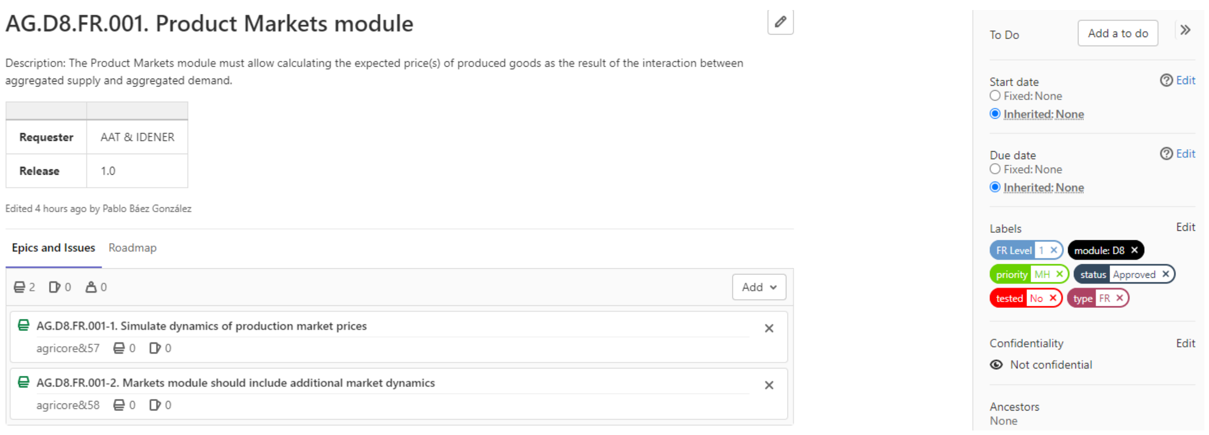
Start date: Fixed: None; Inherited: None

Due date: Fixed: None; Inherited: None

Labels: FR Level 1; module: D7; priority MH; status Approved; tested No; type FR

Figure 42 AG.D7.FR.001. Land Module Market

Module D8: Product Market Module



AG.D8.FR.001. Product Markets module

Description: The Product Markets module must allow calculating the expected price(s) of produced goods as the result of the interaction between aggregated supply and aggregated demand.

Requester	AAT & IDENER
Release	1.0

Edited 4 hours ago by Pablo Báez González

Epics and Issues · Roadmap

2 · 0 · 0

- AG.D8.FR.001-1. Simulate dynamics of production market prices — agricore&57 · 0 · 0
- AG.D8.FR.001-2. Markets module should include additional market dynamics — agricore&58 · 0 · 0

Start date: Fixed: None; Inherited: None

Due date: Fixed: None; Inherited: None

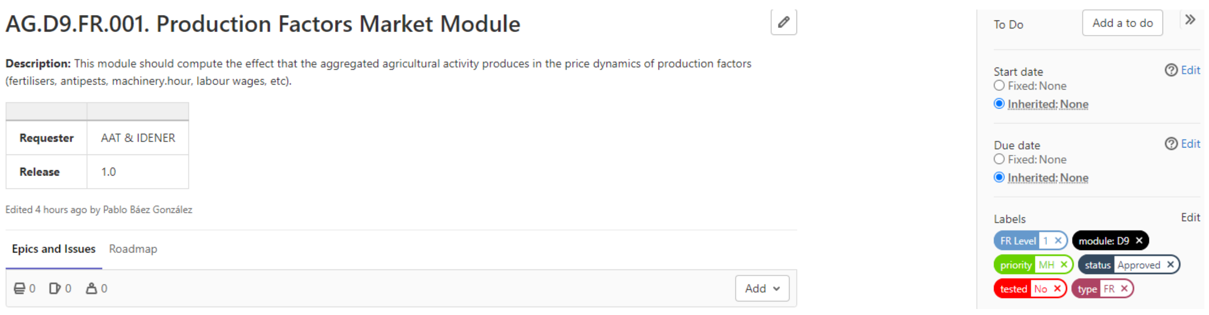
Labels: FR Level 1 · module: D8 · priority MH · status Approved · tested No · type FR

Confidentiality: Not confidential

Ancestors: None

Figure 43 AG.D8.FR.001. Product Markets module

Module D9: Production Factors Market Module



AG.D9.FR.001. Production Factors Market Module

Description: This module should compute the effect that the aggregated agricultural activity produces in the price dynamics of production factors (fertilisers, antipests, machinery.hour, labour wages, etc).

Requester	AAT & IDENER
Release	1.0

Edited 4 hours ago by Pablo Báez González

Epics and Issues · Roadmap

0 · 0 · 0

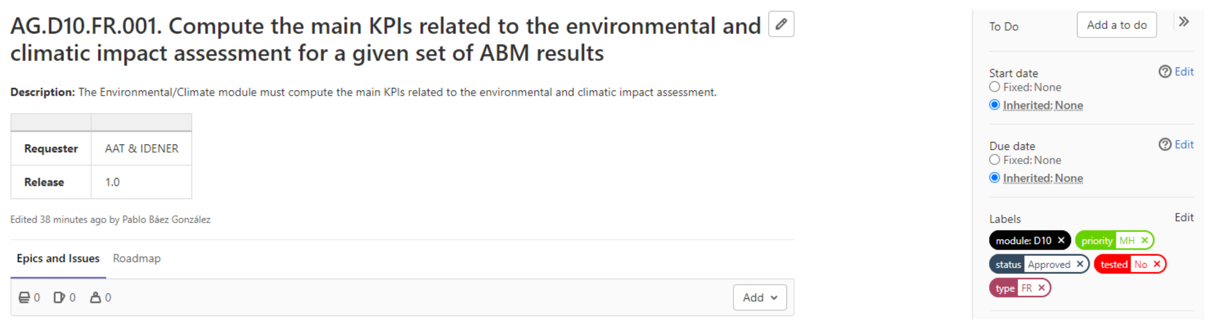
Start date: Fixed: None; Inherited: None

Due date: Fixed: None; Inherited: None

Labels: FR Level 1 · module: D9 · priority MH · status Approved · tested No · type FR

Figure 44 AG.D9.FR.001. Production Factors Market Module

Module D10: Environmental/Climate IAM



AG.D10.FR.001. Compute the main KPIs related to the environmental and climatic impact assessment for a given set of ABM results

Description: The Environmental/Climate module must compute the main KPIs related to the environmental and climatic impact assessment.

Requester	AAT & IDENER
Release	1.0

Edited 38 minutes ago by Pablo Báez González

Epics and Issues · Roadmap

0 · 0 · 0

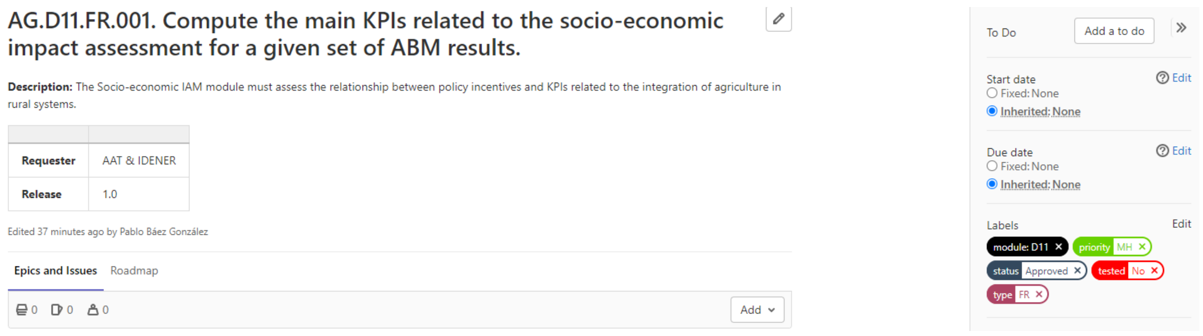
Start date: Fixed: None; Inherited: None

Due date: Fixed: None; Inherited: None

Labels: module: D10 · priority MH · status Approved · tested No · type FR

Figure 45 AG.D10.FR.001. Compute the main KPIs related to the environmental and climatic impact assessment for a given set of ABM results

Module D11: Socio-economic IAM



AG.D11.FR.001. Compute the main KPIs related to the socio-economic impact assessment for a given set of ABM results.

Description: The Socio-economic IAM module must assess the relationship between policy incentives and KPIs related to the integration of agriculture in rural systems.

Requester	AAT & IDENER
Release	1.0

Edited 37 minutes ago by Pablo Báez González

To Do

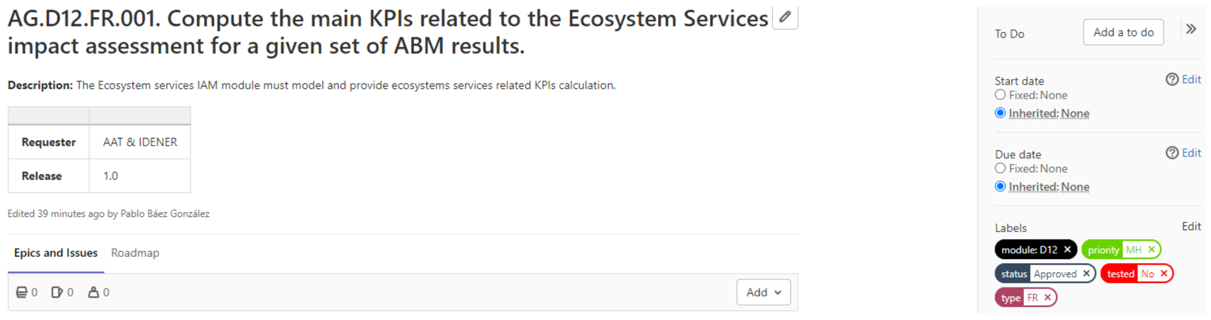
Start date: Fixed: None; Inherited: None

Due date: Fixed: None; Inherited: None

Labels: module: D11; priority MH; status Approved; tested No; type FR

Figure 46 AG.D11.FR.001. Compute the main KPIs related to the socio-economic impact assessment for a given set of ABM results.

Module D12: Ecosystem Services IAM



AG.D12.FR.001. Compute the main KPIs related to the Ecosystem Services impact assessment for a given set of ABM results.

Description: The Ecosystem services IAM module must model and provide ecosystems services related KPIs calculation.

Requester	AAT & IDENER
Release	1.0

Edited 39 minutes ago by Pablo Báez González

To Do

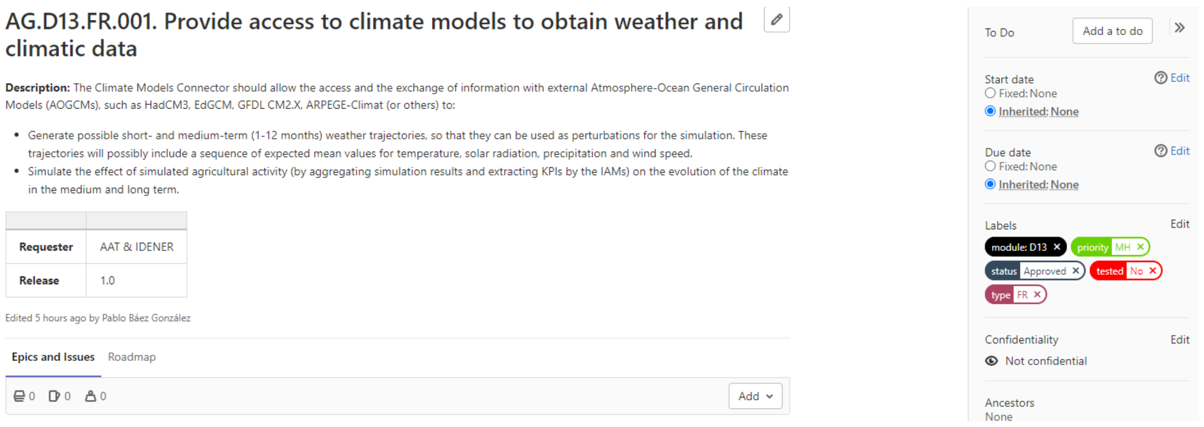
Start date: Fixed: None; Inherited: None

Due date: Fixed: None; Inherited: None

Labels: module: D12; priority MH; status Approved; tested No; type FR

Figure 47 AG.D12.FR.001. Compute the main KPIs related to the Ecosystem Services impact assessment for a given set of ABM results.

Module D13: Climate Models Connectors



AG.D13.FR.001. Provide access to climate models to obtain weather and climatic data

Description: The Climate Models Connector should allow the access and the exchange of information with external Atmosphere-Ocean General Circulation Models (AOGCMs), such as HadCM3, EdGCM, GFDL CM2.X, ARPEGE-Climat (or others) to:

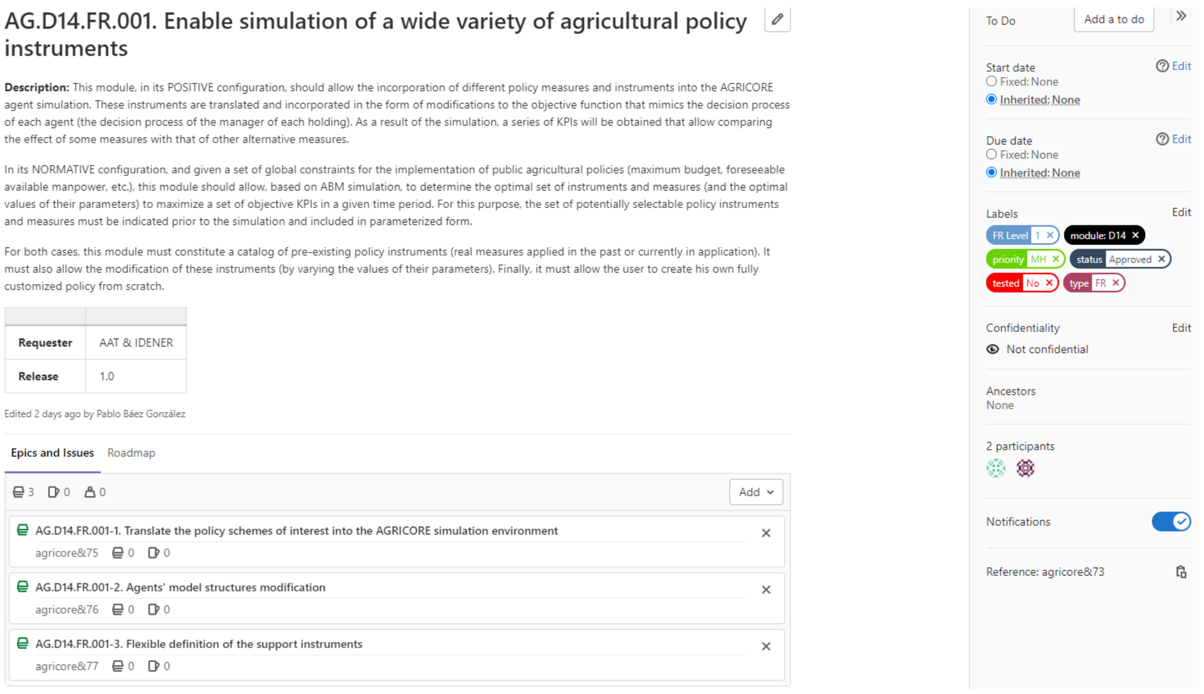
- Generate possible short- and medium-term (1-12 months) weather trajectories, so that they can be used as perturbations for the simulation. These trajectories will possibly include a sequence of expected mean values for temperature, solar radiation, precipitation and wind speed.
- Simulate the effect of simulated agricultural activity (by aggregating simulation results and extracting KPIs by the IAMs) on the evolution of the climate in the medium and long term.

Requester	AAT & IDENER
Release	1.0

Edited 5 hours ago by Pablo Báez González

Figure 48 AG.D13.FR.001. Provide access to climate models to obtain weather and climatic data

Module D14: Policy Environment Module



AG.D14.FR.001. Enable simulation of a wide variety of agricultural policy instruments

Description: This module, in its POSITIVE configuration, should allow the incorporation of different policy measures and instruments into the AGRICORE agent simulation. These instruments are translated and incorporated in the form of modifications to the objective function that mimics the decision process of each agent (the decision process of the manager of each holding). As a result of the simulation, a series of KPIs will be obtained that allow comparing the effect of some measures with that of other alternative measures.

In its NORMATIVE configuration, and given a set of global constraints for the implementation of public agricultural policies (maximum budget, foreseeable available manpower, etc.), this module should allow, based on ABM simulation, to determine the optimal set of instruments and measures (and the optimal values of their parameters) to maximize a set of objective KPIs in a given time period. For this purpose, the set of potentially selectable policy instruments and measures must be indicated prior to the simulation and included in parameterized form.

For both cases, this module must constitute a catalog of pre-existing policy instruments (real measures applied in the past or currently in application). It must also allow the modification of these instruments (by varying the values of their parameters). Finally, it must allow the user to create his own fully customized policy from scratch.

Requester	AAT & IDENER
Release	1.0

Edited 2 days ago by Pablo Báez González

AG.D14.FR.001-1. Translate the policy schemes of interest into the AGRICORE simulation environment
agricore&75 0 0

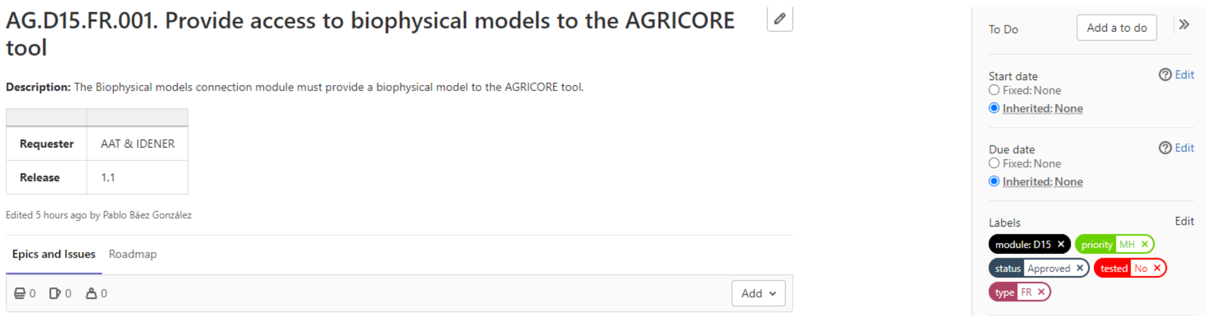
AG.D14.FR.001-2. Agents' model structures modification
agricore&76 0 0

AG.D14.FR.001-3. Flexible definition of the support instruments
agricore&77 0 0

Reference: agricore&73

Figure 49 AG.D14.FR.001. Enable simulation of a wide variety of agricultural policy instruments

Module D15: Biophysical Models Connectors



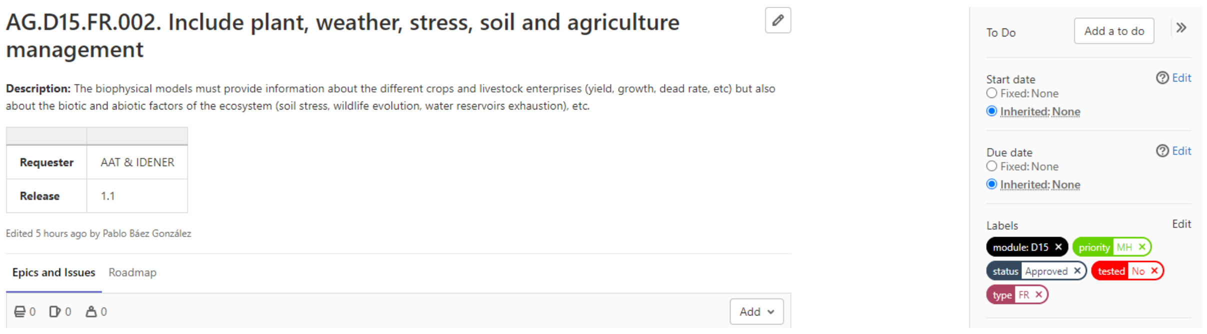
AG.D15.FR.001. Provide access to biophysical models to the AGRICORE tool

Description: The Biophysical models connection module must provide a biophysical model to the AGRICORE tool.

Requester	AAT & IDENER
Release	1.1

Edited 5 hours ago by Pablo Báez González

Figure 50 AG.D15.FR.001. Provide access to biophysical models to the AGRICORE tool



AG.D15.FR.002. Include plant, weather, stress, soil and agriculture management

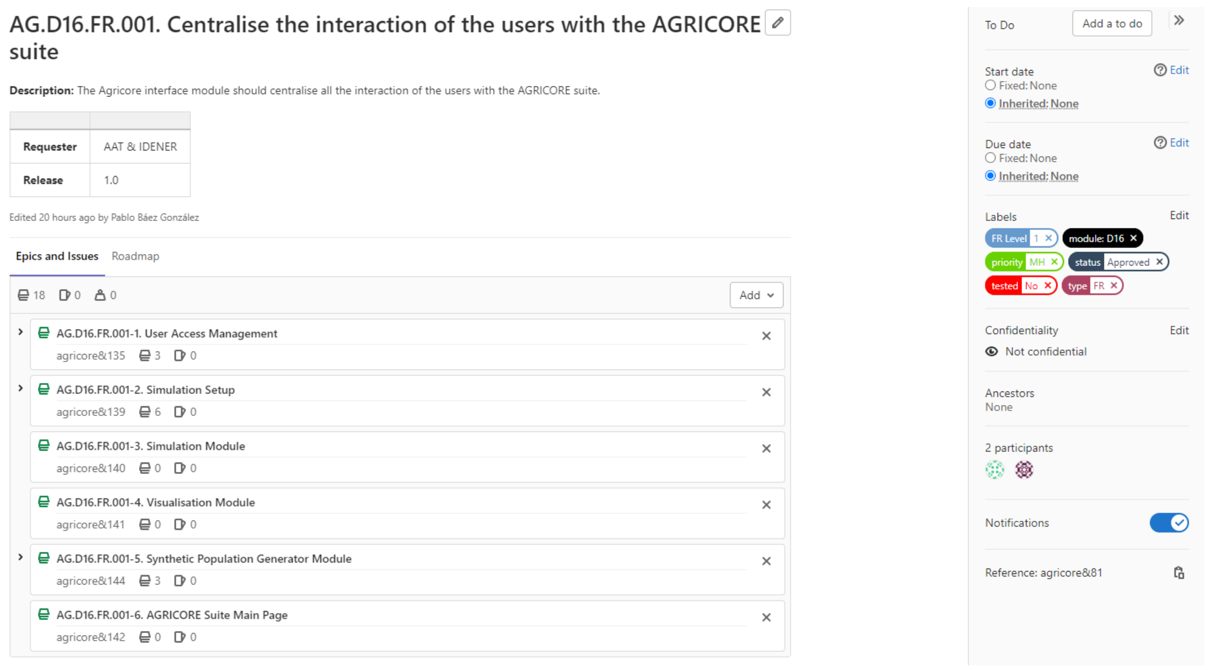
Description: The biophysical models must provide information about the different crops and livestock enterprises (yield, growth, dead rate, etc) but also about the biotic and abiotic factors of the ecosystem (soil stress, wildlife evolution, water reservoirs exhaustion), etc.

Requester	AAT & IDENER
Release	1.1

Edited 5 hours ago by Pablo Báez González

Figure 51 AG.D15.FR.002. Include plant, weather, stress, soil and agriculture management

Module D16: AGRICORE Interface Module



AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite

Description: The Agricore interface module should centralise all the interaction of the users with the AGRICORE suite.

Requester	AAT & IDENER
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues   Roadmap

18 0 0   Add

- AG.D16.FR.001-1. User Access Management — agricore&135 3 0
- AG.D16.FR.001-2. Simulation Setup — agricore&139 6 0
- AG.D16.FR.001-3. Simulation Module — agricore&140 0 0
- AG.D16.FR.001-4. Visualisation Module — agricore&141 0 0
- AG.D16.FR.001-5. Synthetic Population Generator Module — agricore&144 3 0
- AG.D16.FR.001-6. AGRICORE Suite Main Page — agricore&142 0 0

To Do   Add a to do

Start date   Edit
Fixed: None
Inherited: None

Due date   Edit
Fixed: None
Inherited: None

Labels   Edit
FR Level 1   module: D16   priority MH   status Approved   tested No   type FR

Confidentiality   Edit
Not confidential

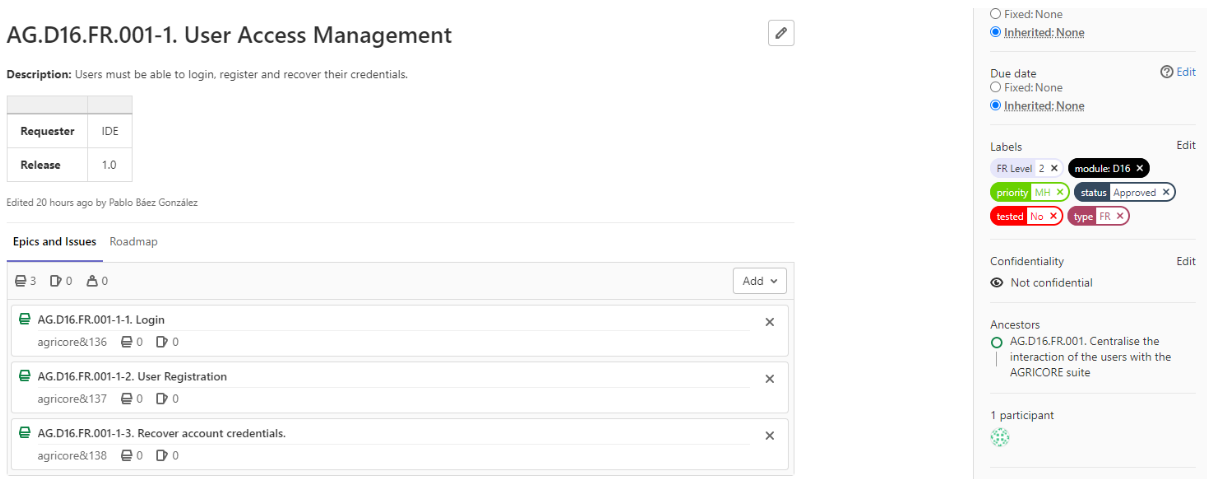
Ancestors
None

2 participants

Notifications

Reference: agricore&81

Figure 52 AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite



AG.D16.FR.001-1. User Access Management

Description: Users must be able to login, register and recover their credentials.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues   Roadmap

3 0 0   Add

- AG.D16.FR.001-1-1. Login — agricore&136 0 0
- AG.D16.FR.001-1-2. User Registration — agricore&137 0 0
- AG.D16.FR.001-1-3. Recover account credentials. — agricore&138 0 0

Fixed: None
Inherited: None

Due date   Edit
Fixed: None
Inherited: None

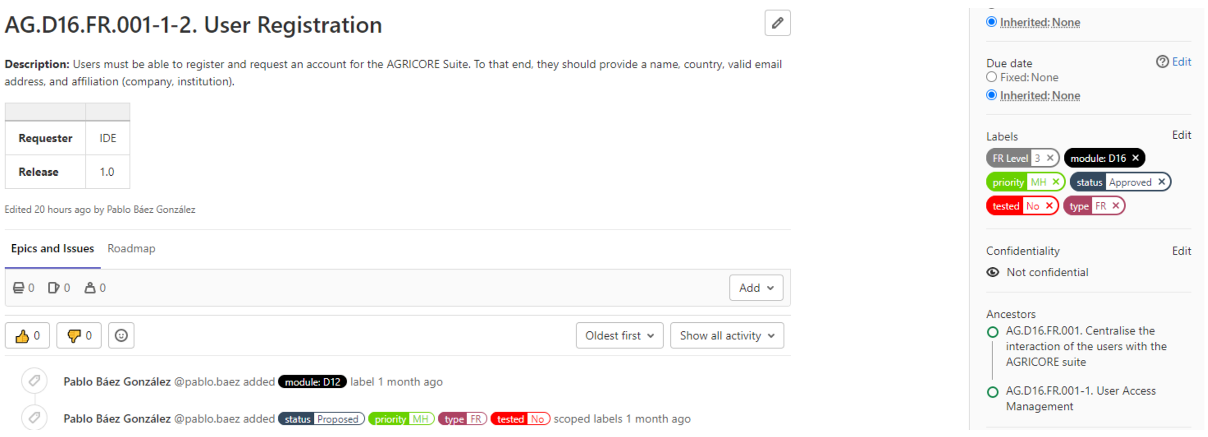
Labels   Edit
FR Level 2   module: D16   priority MH   status Approved   tested No   type FR

Confidentiality   Edit
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite

1 participant

Figure 53 AG.D16.FR.001-1. User Access Management



AG.D16.FR.001-1-2. User Registration

Description: Users must be able to register and request an account for the AGRICORE Suite. To that end, they should provide a name, country, valid email address, and affiliation (company, institution).

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues · Roadmap

0 0 0 · Add

0 0 · Oldest first · Show all activity

Pablo Báez González @pablo.baez added module: D12 label 1 month ago

Pablo Báez González @pablo.baez added status Proposed priority MH type FR tested No scoped labels 1 month ago

Inherited: None

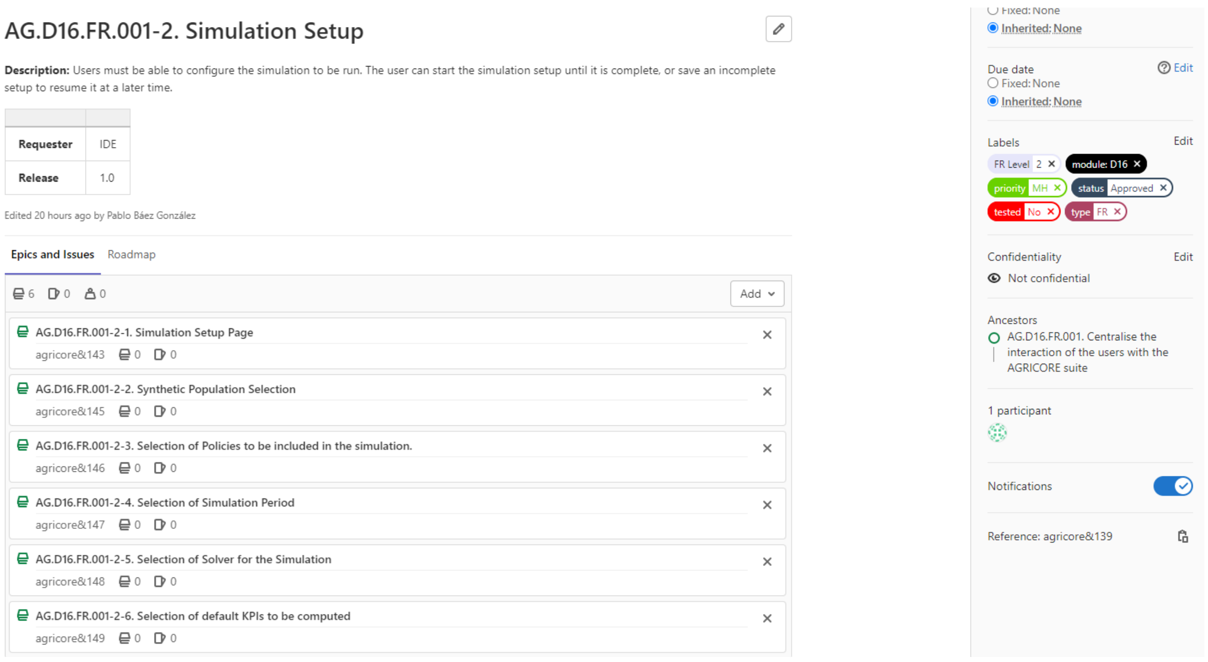
Due date · Edit
Fixed: None
Inherited: None

Labels · Edit
FR Level 3 · module: D16 · priority MH · status Approved · tested No · type FR

Confidentiality · Edit
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-1. User Access Management

Figure 54 AG.D16.FR.001-1-2. User Registration



AG.D16.FR.001-2. Simulation Setup

Description: Users must be able to configure the simulation to be run. The user can start the simulation setup until it is complete, or save an incomplete setup to resume it at a later time.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues · Roadmap

6 0 0 · Add

- AG.D16.FR.001-2-1. Simulation Setup Page — agricore&143 0 0
- AG.D16.FR.001-2-2. Synthetic Population Selection — agricore&145 0 0
- AG.D16.FR.001-2-3. Selection of Policies to be included in the simulation. — agricore&146 0 0
- AG.D16.FR.001-2-4. Selection of Simulation Period — agricore&147 0 0
- AG.D16.FR.001-2-5. Selection of Solver for the Simulation — agricore&148 0 0
- AG.D16.FR.001-2-6. Selection of default KPIs to be computed — agricore&149 0 0

Fixed: None
Inherited: None

Due date · Edit
Fixed: None
Inherited: None

Labels · Edit
FR Level 2 · module: D16 · priority MH · status Approved · tested No · type FR

Confidentiality · Edit
Not confidential

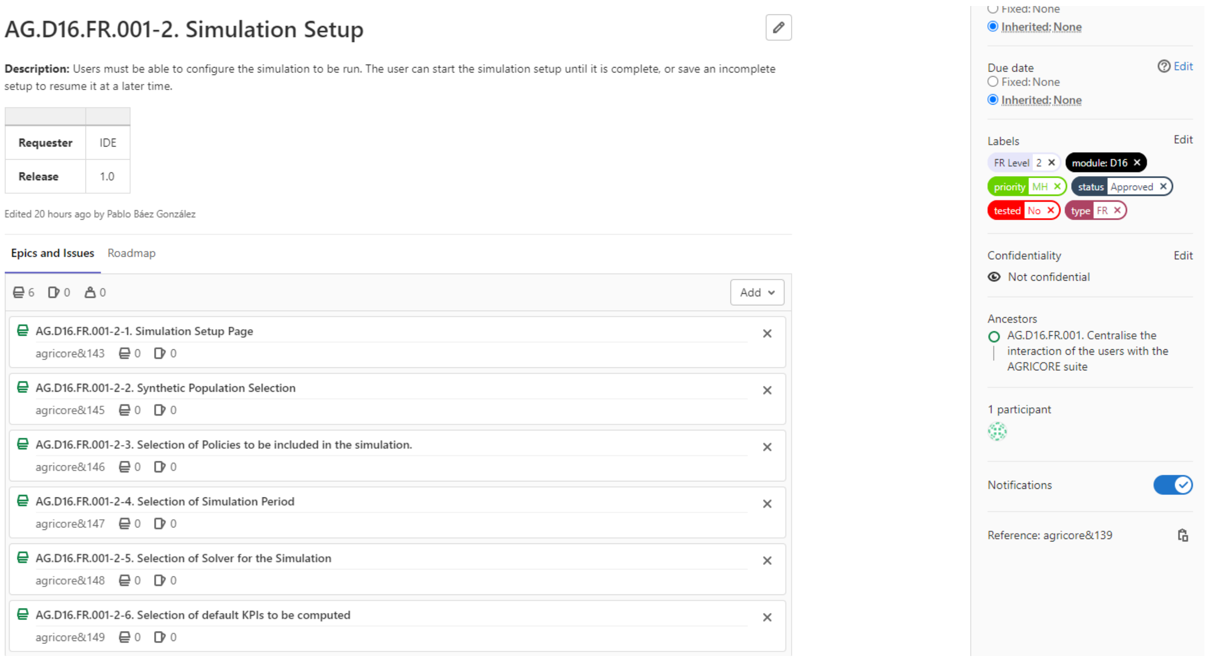
Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite

1 participant

Notifications

Reference: agricore&139

Figure 55 AG.D16.FR.001-2. Simulation Setup



AG.D16.FR.001-2. Simulation Setup

Description: Users must be able to configure the simulation to be run. The user can start the simulation setup until it is complete, or save an incomplete setup to resume it at a later time.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues Roadmap

6 0 0 Add

AG.D16.FR.001-2-1. Simulation Setup Page
agricore&143 0 0

AG.D16.FR.001-2-2. Synthetic Population Selection
agricore&145 0 0

AG.D16.FR.001-2-3. Selection of Policies to be included in the simulation.
agricore&146 0 0

AG.D16.FR.001-2-4. Selection of Simulation Period
agricore&147 0 0

AG.D16.FR.001-2-5. Selection of Solver for the Simulation
agricore&148 0 0

AG.D16.FR.001-2-6. Selection of default KPIs to be computed
agricore&149 0 0

Fixed: None
Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 2 · module: D16 · priority MH · status Approved · tested No · type FR

Confidentiality Edit
Not confidential

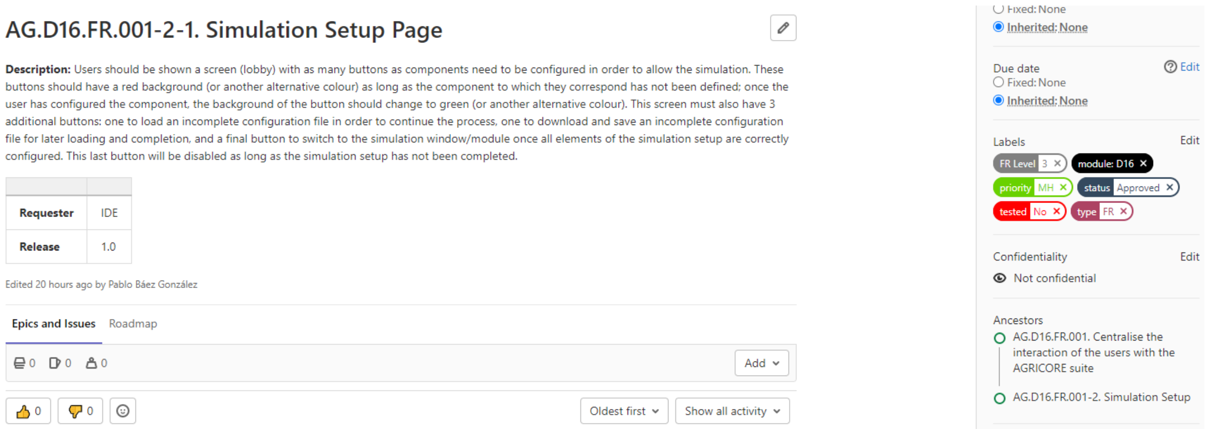
Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite

1 participant

Notifications

Reference: agricore&139

Figure 56 AG.D16.FR.001-2. Simulation Setup



AG.D16.FR.001-2-1. Simulation Setup Page

Description: Users should be shown a screen (lobby) with as many buttons as components need to be configured in order to allow the simulation. These buttons should have a red background (or another alternative colour) as long as the component to which they correspond has not been defined; once the user has configured the component, the background of the button should change to green (or another alternative colour). This screen must also have 3 additional buttons: one to load an incomplete configuration file in order to continue the process, one to download and save an incomplete configuration file for later loading and completion, and a final button to switch to the simulation window/module once all elements of the simulation setup are correctly configured. This last button will be disabled as long as the simulation setup has not been completed.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues Roadmap

0 0 0 Add

0 0 Oldest first Show all activity

Fixed: None
Inherited: None

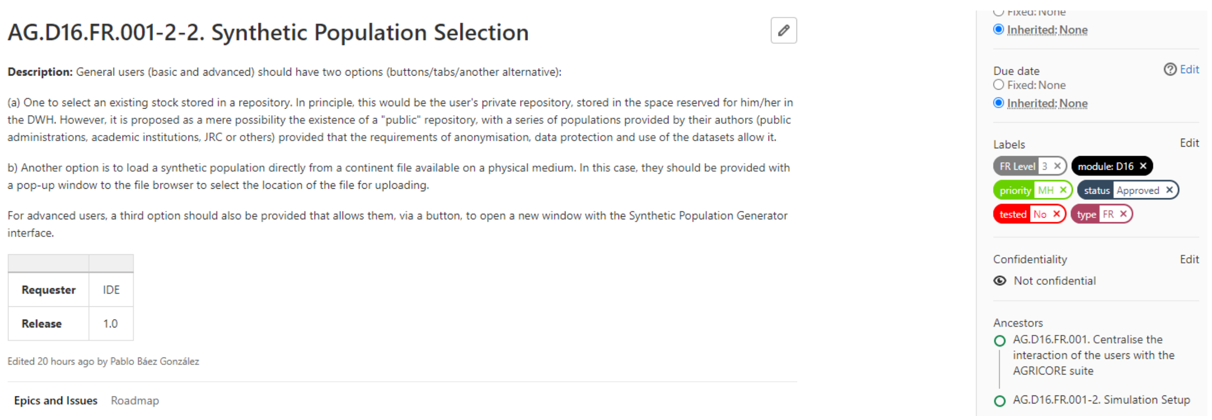
Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 3 · module: D16 · priority MH · status Approved · tested No · type FR

Confidentiality Edit
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-2. Simulation Setup

Figure 57 AG.D16.FR.001-2-1. Simulation Setup Page



AG.D16.FR.001-2-2. Synthetic Population Selection

Description: General users (basic and advanced) should have two options (buttons/tabs/another alternative):

(a) One to select an existing stock stored in a repository. In principle, this would be the user's private repository, stored in the space reserved for him/her in the DWH. However, it is proposed as a mere possibility the existence of a "public" repository, with a series of populations provided by their authors (public administrations, academic institutions, JRC or others) provided that the requirements of anonymisation, data protection and use of the datasets allow it.

b) Another option is to load a synthetic population directly from a continent file available on a physical medium. In this case, they should be provided with a pop-up window to the file browser to select the location of the file for uploading.

For advanced users, a third option should also be provided that allows them, via a button, to open a new window with the Synthetic Population Generator interface.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues Roadmap

Fixed: None
Inherited: None

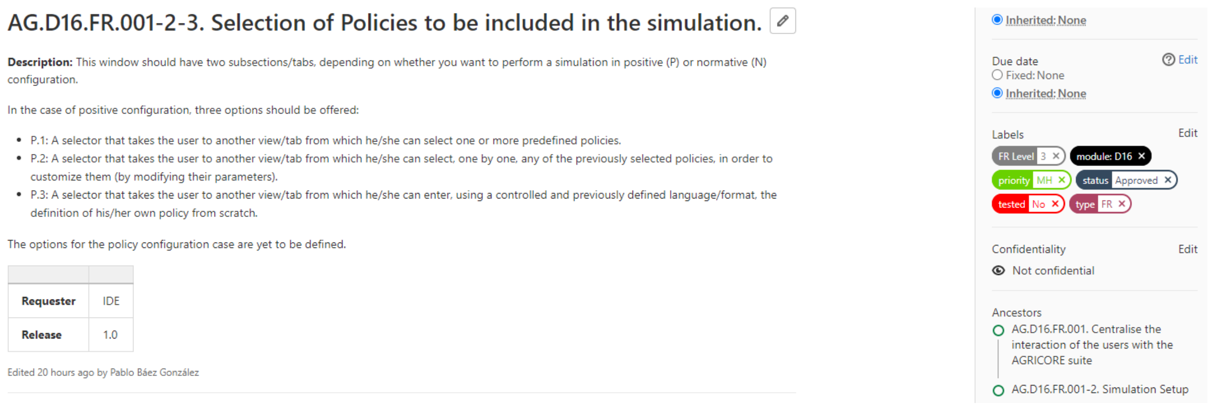
Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 3 module: D16 priority MH status Approved tested No type FR

Confidentiality Edit
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-2. Simulation Setup

Figure 58 AG.D16.FR.001-2-2. Synthetic Population Selection



AG.D16.FR.001-2-3. Selection of Policies to be included in the simulation.

Description: This window should have two subsections/tabs, depending on whether you want to perform a simulation in positive (P) or normative (N) configuration.

In the case of positive configuration, three options should be offered:

- P.1: A selector that takes the user to another view/tab from which he/she can select one or more predefined policies.
- P.2: A selector that takes the user to another view/tab from which he/she can select, one by one, any of the previously selected policies, in order to customize them (by modifying their parameters).
- P.3: A selector that takes the user to another view/tab from which he/she can enter, using a controlled and previously defined language/format, the definition of his/her own policy from scratch.

The options for the policy configuration case are yet to be defined.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Inherited: None

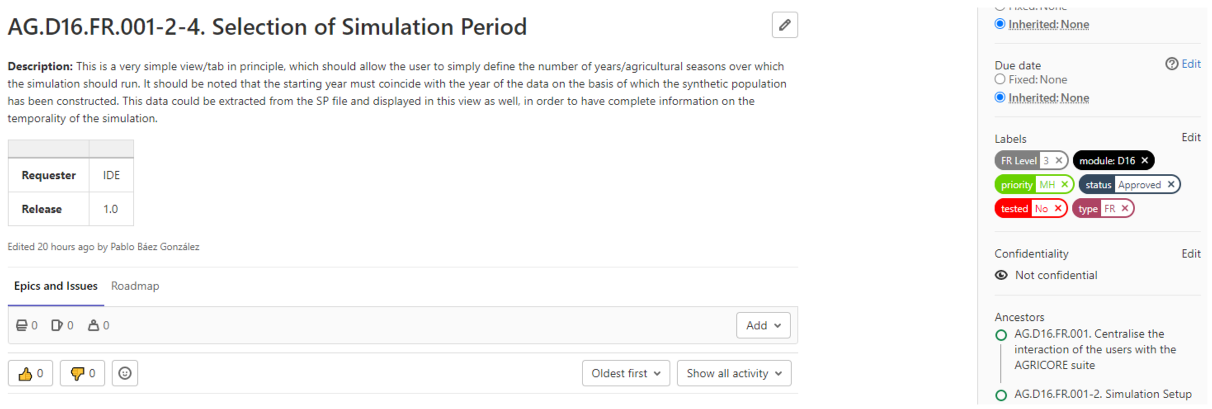
Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 3 module: D16 priority MH status Approved tested No type FR

Confidentiality Edit
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-2. Simulation Setup

Figure 59 AG.D16.FR.001-2-3. Selection of Policies to be included in the simulation.



AG.D16.FR.001-2-4. Selection of Simulation Period

Description: This is a very simple view/tab in principle, which should allow the user to simply define the number of years/agricultural seasons over which the simulation should run. It should be noted that the starting year must coincide with the year of the data on the basis of which the synthetic population has been constructed. This data could be extracted from the SP file and displayed in this view as well, in order to have complete information on the temporality of the simulation.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues Roadmap

0 0 0 Add

0 0 Oldest first Show all activity

Fixed: None
Inherited: None

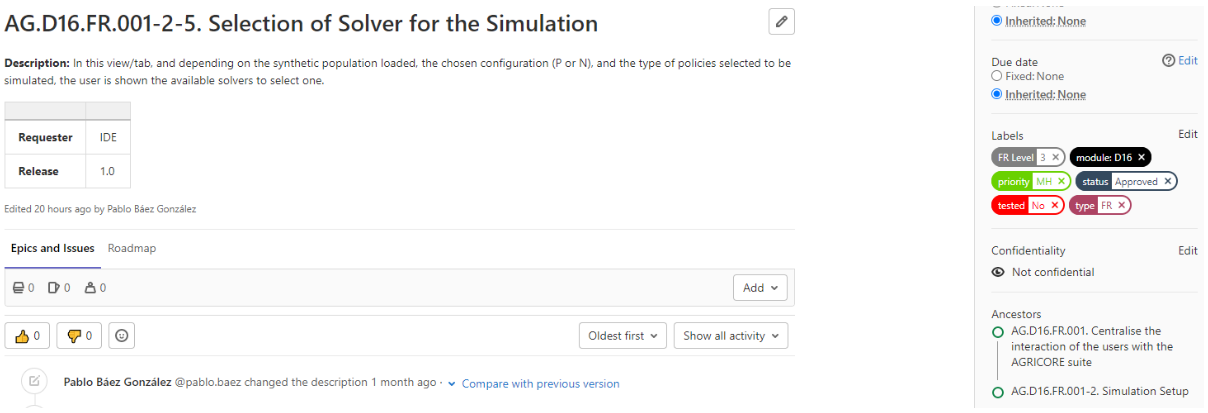
Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 3 module: D16 priority MH status Approved tested No type FR

Confidentiality Edit
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-2. Simulation Setup

Figure 60 AG.D16.FR.001-2-4. Selection of Simulation Period



AG.D16.FR.001-2-5. Selection of Solver for the Simulation

Description: In this view/tab, and depending on the synthetic population loaded, the chosen configuration (P or N), and the type of policies selected to be simulated, the user is shown the available solvers to select one.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Pablo Báez González @pablo.baez changed the description 1 month ago · Compare with previous version

Inherited: None

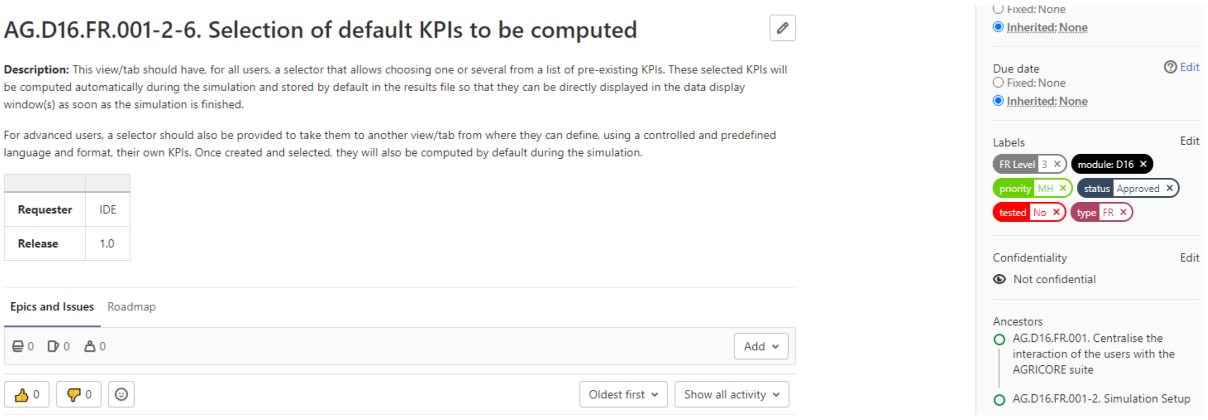
Due date
Fixed: None
Inherited: None

Labels: FR Level 3, module: D16, priority MH, status Approved, tested No, type FR

Confidentiality: Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-2. Simulation Setup

Figure 61 AG.D16.FR.001-2-5. Selection of Solver for the Simulation



AG.D16.FR.001-2-6. Selection of default KPIs to be computed

Description: This view/tab should have, for all users, a selector that allows choosing one or several from a list of pre-existing KPIs. These selected KPIs will be computed automatically during the simulation and stored by default in the results file so that they can be directly displayed in the data display window(s) as soon as the simulation is finished.

For advanced users, a selector should also be provided to take them to another view/tab from where they can define, using a controlled and predefined language and format, their own KPIs. Once created and selected, they will also be computed by default during the simulation.

Requester	IDE
Release	1.0

Fixed: None
Inherited: None

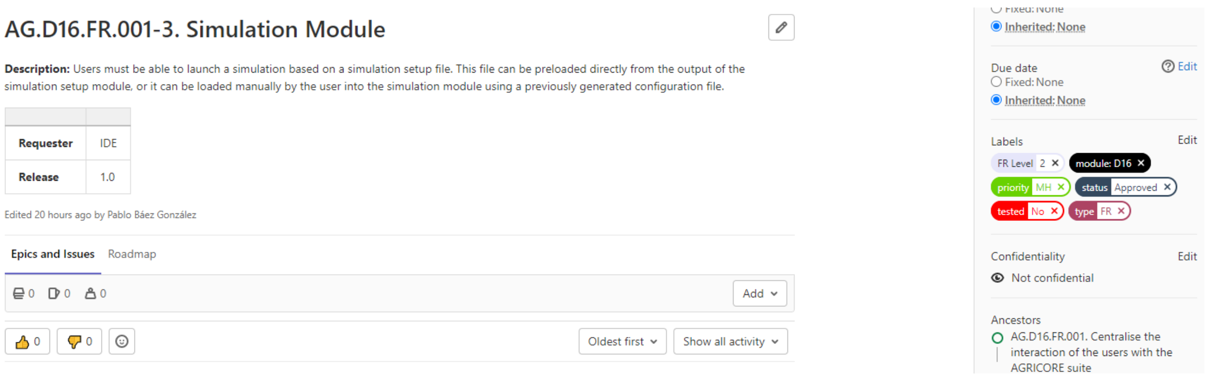
Due date
Fixed: None
Inherited: None

Labels: FR Level 3, module: D16, priority MH, status Approved, tested No, type FR

Confidentiality: Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-2. Simulation Setup

Figure 62 AG.D16.FR.001-2-6. Selection of default KPIs to be computed



AG.D16.FR.001-3. Simulation Module

Description: Users must be able to launch a simulation based on a simulation setup file. This file can be preloaded directly from the output of the simulation setup module, or it can be loaded manually by the user into the simulation module using a previously generated configuration file.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Fixed: None
Inherited: None

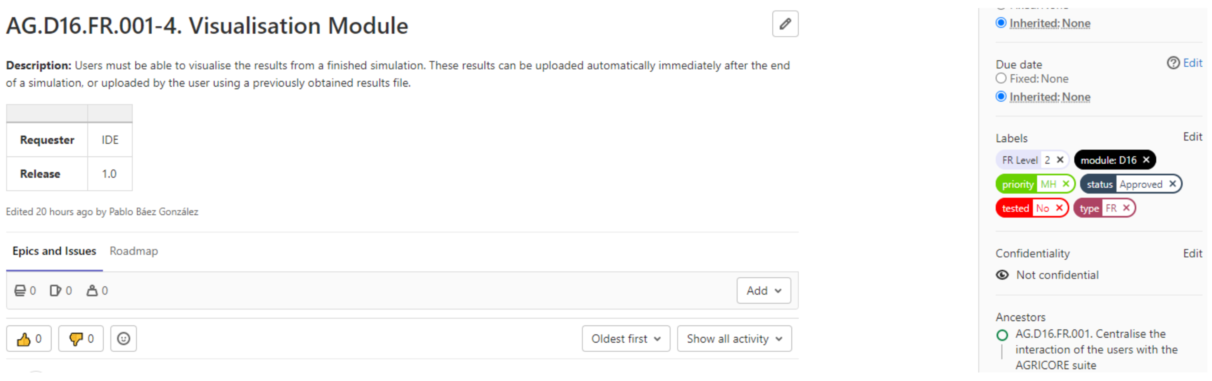
Due date
Fixed: None
Inherited: None

Labels: FR Level 2, module: D16, priority MH, status Approved, tested No, type FR

Confidentiality: Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite

Figure 63 AG.D16.FR.001-3. Simulation Module



AG.D16.FR.001-4. Visualisation Module

Description: Users must be able to visualise the results from a finished simulation. These results can be uploaded automatically immediately after the end of a simulation, or uploaded by the user using a previously obtained results file.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues Roadmap

0 0 0 Add

0 0 Oldest first Show all activity

Inherited: None

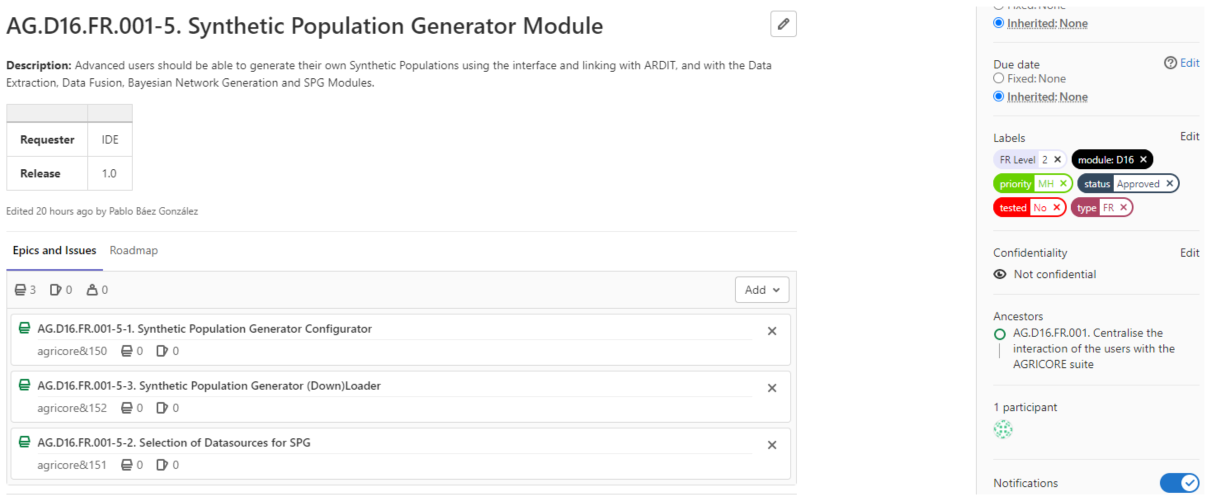
Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 2 | module: D16 | priority MH | status Approved | tested No | type FR

Confidentiality Edit
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite

Figure 64 AG.D16.FR.001-4. Visualisation Module



AG.D16.FR.001-5. Synthetic Population Generator Module

Description: Advanced users should be able to generate their own Synthetic Populations using the interface and linking with ARDIT, and with the Data Extraction, Data Fusion, Bayesian Network Generation and SPG Modules.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Epics and Issues Roadmap

3 0 0 Add

AG.D16.FR.001-5-1. Synthetic Population Generator Configurator
agricore&150 0 0

AG.D16.FR.001-5-3. Synthetic Population Generator (Down)Loader
agricore&152 0 0

AG.D16.FR.001-5-2. Selection of Datasources for SPG
agricore&151 0 0

Inherited: None

Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 2 | module: D16 | priority MH | status Approved | tested No | type FR

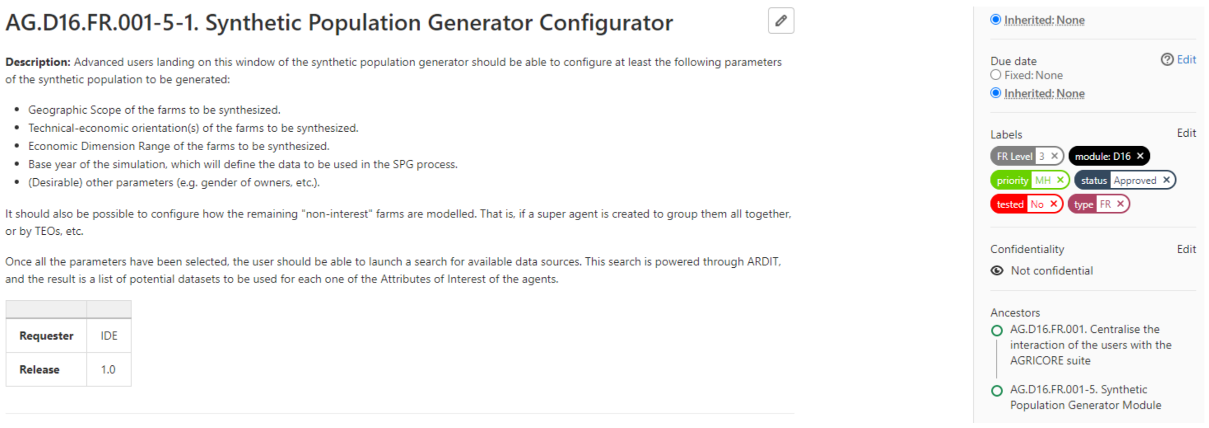
Confidentiality Edit
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite

1 participant

Notifications

Figure 65 AG.D16.FR.001-5. Visualisation Module



AG.D16.FR.001-5-1. Synthetic Population Generator Configurator

Description: Advanced users landing on this window of the synthetic population generator should be able to configure at least the following parameters of the synthetic population to be generated:

- Geographic Scope of the farms to be synthesized.
- Technical-economic orientation(s) of the farms to be synthesized.
- Economic Dimension Range of the farms to be synthesized.
- Base year of the simulation, which will define the data to be used in the SPG process.
- (Desirable) other parameters (e.g. gender of owners, etc.).

It should also be possible to configure how the remaining "non-interest" farms are modelled. That is, if a super agent is created to group them all together, or by TEOs, etc.

Once all the parameters have been selected, the user should be able to launch a search for available data sources. This search is powered through ARDIT, and the result is a list of potential datasets to be used for each one of the Attributes of Interest of the agents.

Requester	IDE
Release	1.0

Inherited: None

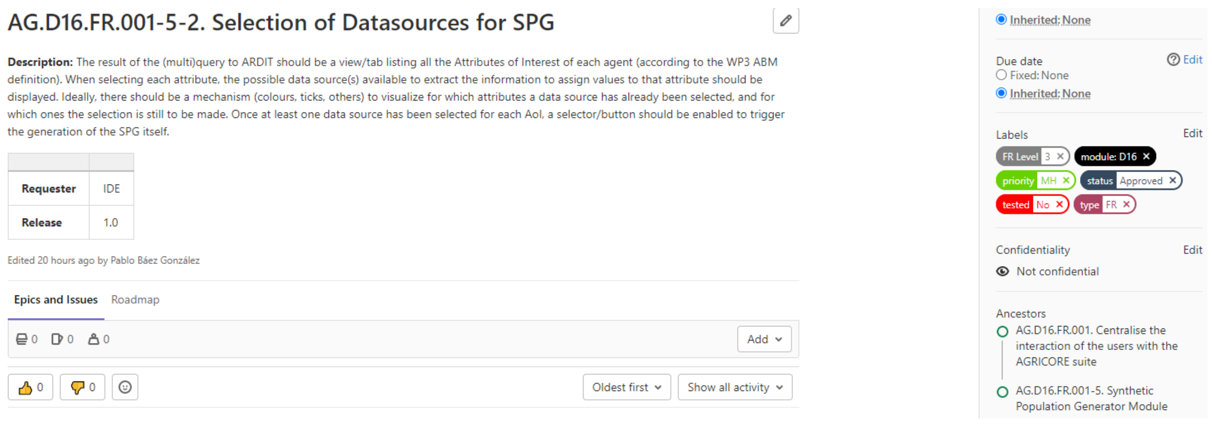
Due date Edit
Fixed: None
Inherited: None

Labels Edit
FR Level 3 | module: D16 | priority MH | status Approved | tested No | type FR

Confidentiality Edit
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-5. Synthetic Population Generator Module

Figure 66 AG.D16.FR.001-5-1. Synthetic Population Generator Configurator



AG.D16.FR.001-5-2. Selection of Datasources for SPG

Description: The result of the (multi)query to ARDIT should be a view/tab listing all the Attributes of Interest of each agent (according to the WP3 ABM definition). When selecting each attribute, the possible data source(s) available to extract the information to assign values to that attribute should be displayed. Ideally, there should be a mechanism (colours, ticks, others) to visualize for which attributes a data source has already been selected, and for which ones the selection is still to be made. Once at least one data source has been selected for each AoI, a selector/button should be enabled to trigger the generation of the SPG itself.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Inherited: None

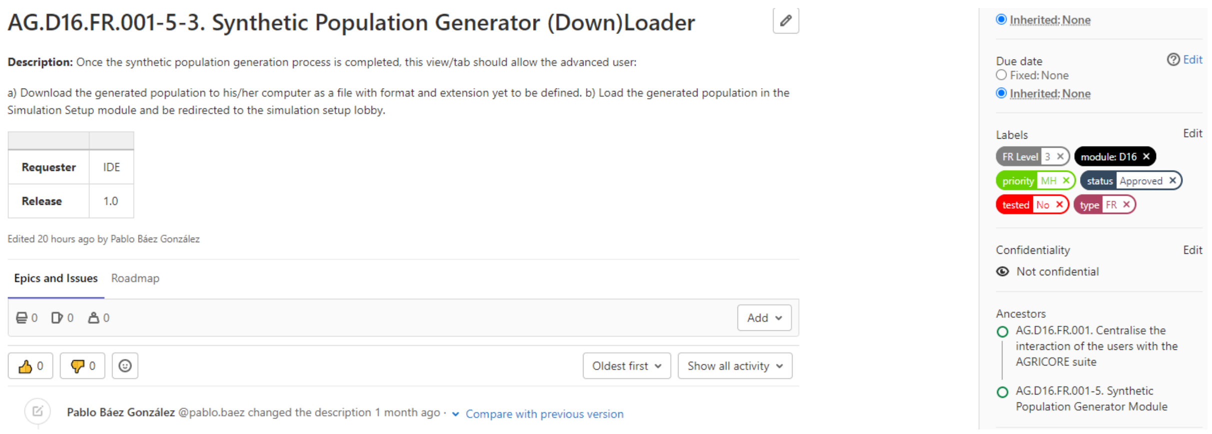
Due date
Fixed: None
Inherited: None

Labels
FR Level 3 · module: D16 · priority MH · status Approved · tested No · type FR

Confidentiality
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-5. Synthetic Population Generator Module

Figure 67 AG.D16.FR.001-5-2. Selection of Datasources for SPG



AG.D16.FR.001-5-3. Synthetic Population Generator (Down)Loader

Description: Once the synthetic population generation process is completed, this view/tab should allow the advanced user:

a) Download the generated population to his/her computer as a file with format and extension yet to be defined. b) Load the generated population in the Simulation Setup module and be redirected to the simulation setup lobby.

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Pablo Báez González @pablo.baez changed the description 1 month ago · Compare with previous version

Inherited: None

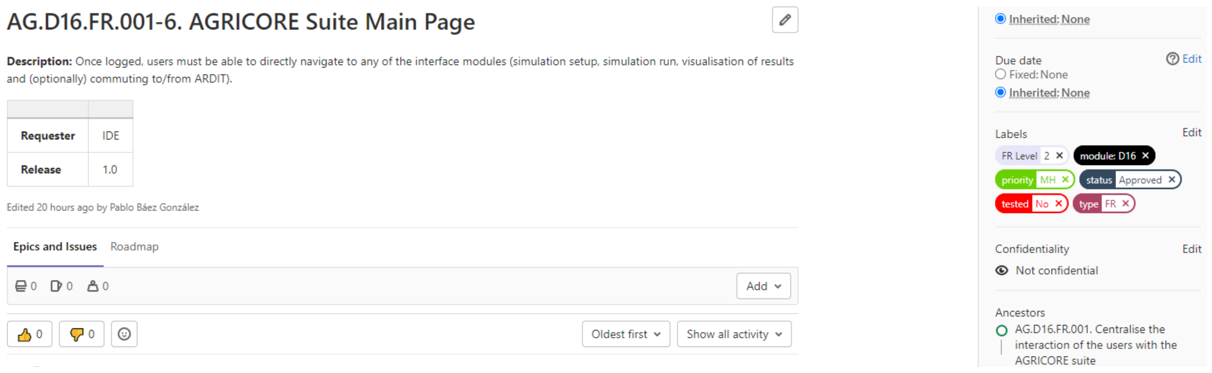
Due date
Fixed: None
Inherited: None

Labels
FR Level 3 · module: D16 · priority MH · status Approved · tested No · type FR

Confidentiality
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite
AG.D16.FR.001-5. Synthetic Population Generator Module

Figure 68 AG.D16.FR.001-5-3. Synthetic Population Generator (Down)Loader



AG.D16.FR.001-6. AGRICORE Suite Main Page

Description: Once logged, users must be able to directly navigate to any of the interface modules (simulation setup, simulation run, visualisation of results and (optionally) commuting to/from ARDIT).

Requester	IDE
Release	1.0

Edited 20 hours ago by Pablo Báez González

Inherited: None

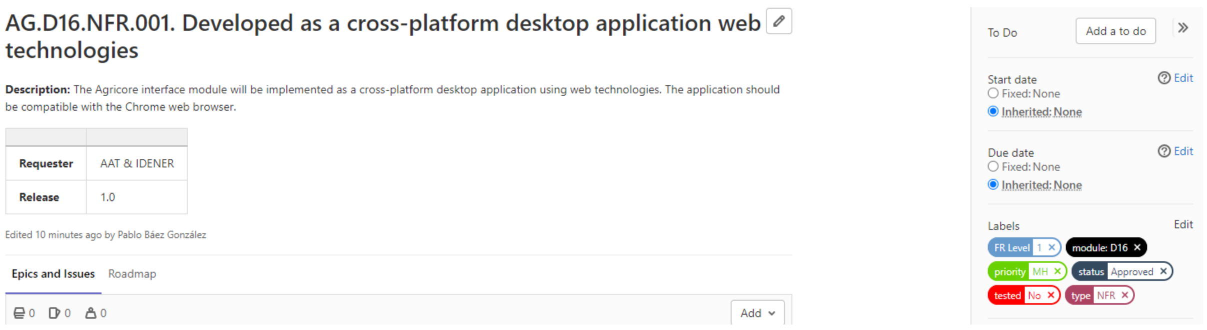
Due date
Fixed: None
Inherited: None

Labels
FR Level 2 · module: D16 · priority MH · status Approved · tested No · type FR

Confidentiality
Not confidential

Ancestors
AG.D16.FR.001. Centralise the interaction of the users with the AGRICORE suite

Figure 69 AG.D16.FR.001-6. AGRICORE Suite Main Page



AG.D16.NFR.001. Developed as a cross-platform desktop application web technologies

Description: The Agricore interface module will be implemented as a cross-platform desktop application using web technologies. The application should be compatible with the Chrome web browser.

Requester	AAT & IDENER
Release	1.0

Edited 10 minutes ago by Pablo Báez González

Figure 70 AG.D16.NFR.001. Developed as a cross-platform desktop application web technologies